Agilent
PNA Series Network
Analyzer

Printed Version of PNA Help
User's and Programming Guide

Supports Firmware A.08.00
March 12, 2008
Warranty Statement

THE MATERIAL CONTAINED IN THIS DOCUMENT IS PROVIDED “AS IS,” AND IS SUBJECT TO BEING CHANGED, WITHOUT NOTICE, IN FUTURE EDITIONS. FURTHER, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, AGILENT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED WITH REGARD TO THIS MANUAL AND ANY INFORMATION CONTAINED HEREIN, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. AGILENT SHALL NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR ANY INFORMATION CONTAINED HEREIN. SHOULD AGILENT AND THE USER HAVE A SEPARATE WRITTEN AGREEMENT WITH WARRANTY TERMS COVERING THE MATERIAL IN THIS DOCUMENT THAT CONFLICT WITH THESE TERMS, THE WARRANTY TERMS IN THE SEPARATE AGREEMENT WILL CONTROL.

DFARS/Restricted Rights Notice

If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as “Commercial computer software” as defined in DFAR 252.227-7014 (June 1995), or as a “commercial item” as defined in FAR 2.101(a) or as “Restricted computer software” as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies' standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June 1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Certification

Agilent Technologies, Inc. certifies that this product met its published specifications at the time of shipment from the factory. Agilent Technologies, Inc. further certifies that its calibration measurements are traceable to the United States National Institute of Standards and Technology, to the extent allowed by the Institute's calibration facility, and to the calibration facilities of other International Standards Organization members.

Contacting Agilent

Assistance with test and measurements needs and information on finding a local Agilent office are available on the Web at: http://www.agilent.com/find/assist

If you do not have access to the Internet, please contact your Agilent field engineer.

In any correspondence or telephone conversation, refer to the Agilent product by its model number and full serial number. With this information, the Agilent representative can determine whether your product is still within its warranty period.

Printed Version of PNA Help User's and Programming Guide
# TABLE OF CONTENTS

**Whats New**
- 41

**Administrative Tasks**
- PNA User Accounts and Passwords 46
- Computer Properties 49
- Error-check and Disk Defragmenter 52
- Operating System Recovery 53
- Windows Considerations 54

**Applications**
- Gain Compression Application (Opt 086) 56
- Noise Figure (Opt 029) 77
- Pulsed Application (Opt H08) 91
- WideBand Pulsed App 105
- Frequency Offset (Opt 080)
  - Frequency Offset (Option 080) 106
  - Frequency Converting Device Measurements 116
  - Frequency Offset Calibration 117
  - Conversion Loss 119
  - Conversion Compression 123
  - Isolation 125
  - Harmonic Distortion 128
  - Return Loss and VSWR 130
- Frequency Converter Application (Opt 083)
  - Known Issues 132
  - Overview 133
  - Using the Frequency Converter Application 135
  - Calibrations 145
  - Configure a Mixer 166
  - Configure an External LO Source 175
  - How to make a VMC Fixed Output Measurement 183
  - How to make an SMC Fixed Output Measurement 192
  - Embedded LO (Opt 084) 200
Characterize Adaptor Macro
SMC with a Booster Amp
Time Domain (Opt 010)

Quick Start
Front Panel Tour (E836x and PNA-L)
Front Panel Tour (PNA-X)
Rear Panel Tour (E836x and PNA-L)
Rear Panel Tour (PNA-X)
Powering the PNA ON and OFF
PNA User Interface (E836x and PNA-L)
Traces, Channels, and Windows on the PNA
Basic Measurement Sequence
Frequency Blanking
Internal Second Source
Connectivity Guide
PNA Preferences
VNC
Using Help

1. Set Up a Measurement
Preset the PNA
Measurement Parameters
Measurement Classes
Frequency Range
Power Level
Sweep Settings
Trigger Setup
External Triggering
Trigger Model Animation
Data Format and Scale
Preconfigured Measurement Setups
Path Configurator
Customize Your Analyzer Screen
Copy Channels
ADC Measurements 376
System Impedance 379

2. Optimize a Measurement
   Dynamic Range 381
   Dynamic Range-Jumpers 383
   Dynamic Range-Test Set Option 384
   Number of Data Points 385
   Phase Accuracy 387
   Electrically Long Devices 391
   Reflection Accuracy 393
   Measurement Stability 396
   Noise Reduction Techniques 398
   Crosstalk 406
   Effects of Accessories 407
   Fastest Sweep 408
   Multiple State Measurements 410
   Fastest Data Transfer 413
   Using Macros 414

3. Calibrate a Measurement
   Calibration Overview 418
   Calibration Standards 420
   Calibration Wizard 424
   Select a Calibration 439
   Using Cal Sets 443
   Error Correction and Interpolation 451
   Calibration Thru Methods 455
   Accurate Calibrations 461
   Validity of a Cal 464
   ECal 469
   ECal User Characterization 477
   TRL Calibration 489
   Calibration Preferences 493
   Measurement Errors 497
Modify Cal Kits
Power Calibration
Fixture Compensation
Port Extensions
Characterize Adaptor Macro
Delta Match Calibration

4. Analyze Data
Locate Data Using Markers
Math & Memory Operations
Equation Editor
Use Limits to Test Devices

5. Output Data
Save and Recall Data
Drive Mapping
Print

Tutorials
App Notes
Network Analyzer Basics
Connector Care
ESD Protection

Measurements
Absolute Output Power
AM-PM Conversion
Amplifier Measurements
Antenna Measurements
Balanced Measurements
Complex Impedance
Comparing the PNA Delay Functions
Deviation from Linear Phase
External Source Control
Gain and Flatness
Gain Compression
Group Delay
<table>
<thead>
<tr>
<th>Class/Interface Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExternalTestsets Collection</td>
<td>754</td>
</tr>
<tr>
<td>Fixturing Object</td>
<td>756</td>
</tr>
<tr>
<td>FOM Collection</td>
<td>761</td>
</tr>
<tr>
<td>FOMRange Object</td>
<td>763</td>
</tr>
<tr>
<td>GainCompression Object</td>
<td>766</td>
</tr>
<tr>
<td>GainCompressionCal Object</td>
<td>769</td>
</tr>
<tr>
<td>Gating Object</td>
<td>771</td>
</tr>
<tr>
<td>GuidedCalibration Object</td>
<td>773</td>
</tr>
<tr>
<td>HWAuxIO Object</td>
<td>776</td>
</tr>
<tr>
<td>HWExternalTestSetIO Object</td>
<td>779</td>
</tr>
<tr>
<td>HWMaterialHandlerIO Object</td>
<td>781</td>
</tr>
<tr>
<td>IIFConfiguration Object</td>
<td>783</td>
</tr>
<tr>
<td>IMixer Interface</td>
<td>786</td>
</tr>
<tr>
<td>InterfaceControl Object</td>
<td>789</td>
</tr>
<tr>
<td>Limit Test Collection</td>
<td>790</td>
</tr>
<tr>
<td>LimitSegment Object</td>
<td>792</td>
</tr>
<tr>
<td>Marker Object</td>
<td>794</td>
</tr>
<tr>
<td>Measurement Object</td>
<td>797</td>
</tr>
<tr>
<td>Measurements Collection</td>
<td>804</td>
</tr>
<tr>
<td>NAWindow Object</td>
<td>805</td>
</tr>
<tr>
<td>NAWindows Collection</td>
<td>807</td>
</tr>
<tr>
<td>NoiseCal Object</td>
<td>808</td>
</tr>
<tr>
<td>NoiseFigure Object</td>
<td>810</td>
</tr>
<tr>
<td>PathConfiguration Object</td>
<td>812</td>
</tr>
<tr>
<td>PathConfigurationManager Object</td>
<td>814</td>
</tr>
<tr>
<td>PathElement Object</td>
<td>816</td>
</tr>
<tr>
<td>Port Extension Object</td>
<td>818</td>
</tr>
<tr>
<td>PowerLossSegment Object</td>
<td>820</td>
</tr>
<tr>
<td>PowerLossSegments Collection</td>
<td>822</td>
</tr>
<tr>
<td>PowerMeterInterface Object</td>
<td>823</td>
</tr>
<tr>
<td>PowerMeterInterfaces Collection</td>
<td>825</td>
</tr>
<tr>
<td>PowerSensor Object</td>
<td>827</td>
</tr>
<tr>
<td>PowerSensorCalFactorSegment Object</td>
<td>829</td>
</tr>
</tbody>
</table>
PowerSensors Collection 830
Preferences Object 831
PulseGenerator Object 833
SCPIStringParser Object 835
Segment Object 837
Segments Collection 839
SignalProcessingModuleFour Object 841
SMC Type Object 844
SourcePowerCalibrator Object 846
TestsetControl Object 849
Trace Object 851
Traces Collection 853
Transform Object 854
TriggerSetup Object 856
VMC Type Object 858

Properties

AcceptTriggerBeforeArmed 861
AcquisitionDirection 862
AcquisitionMode 863
Active Cal Kit 864
Active Channel 865
Active Marker 866
Active Measurement 867
Active NAWindow 868
ActiveXAxisRange 869
ADCCaptureMode 870
ALCLevelingMode 871
Active Trace 872
AllowArbitrarySegments 873
Alternate Sweep 874
AmbientTemperature 875
Application 876
Arrange Windows 877
<table>
<thead>
<tr>
<th>Attenuator Mode</th>
<th>878</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attenuator</td>
<td>879</td>
</tr>
<tr>
<td>AutoIFBWAdjustment</td>
<td>881</td>
</tr>
<tr>
<td>AutoOrient</td>
<td>882</td>
</tr>
<tr>
<td>AutoPortExtConfig</td>
<td>883</td>
</tr>
<tr>
<td>AutoPortExtDCOffset</td>
<td>884</td>
</tr>
<tr>
<td>AutoPortExtLoss</td>
<td>885</td>
</tr>
<tr>
<td>AutoPortExtSearchStart</td>
<td>886</td>
</tr>
<tr>
<td>AutoPortExtSearchStop</td>
<td>887</td>
</tr>
<tr>
<td>AutoPortExtState</td>
<td>888</td>
</tr>
<tr>
<td>AuxiliaryTriggerCount</td>
<td>889</td>
</tr>
<tr>
<td>AuxTriggerScopelsGlobal</td>
<td>890</td>
</tr>
<tr>
<td>Averaging Count</td>
<td>892</td>
</tr>
<tr>
<td>Averaging Factor</td>
<td>893</td>
</tr>
<tr>
<td>Averaging ON/OFF</td>
<td>894</td>
</tr>
<tr>
<td>AvoidSpurs</td>
<td>895</td>
</tr>
<tr>
<td>BalancedMode</td>
<td>896</td>
</tr>
<tr>
<td>Bandwidth Target</td>
<td>897</td>
</tr>
<tr>
<td>Bandwidth Tracking</td>
<td>898</td>
</tr>
<tr>
<td>BB_BalPort1Negative</td>
<td>899</td>
</tr>
<tr>
<td>BB_BalPort1Positive</td>
<td>900</td>
</tr>
<tr>
<td>BB_BalPort2Negative</td>
<td>901</td>
</tr>
<tr>
<td>BB_BalPort2Positive</td>
<td>902</td>
</tr>
<tr>
<td>BBalMeasurementProperty</td>
<td>903</td>
</tr>
<tr>
<td>Begin Response</td>
<td>904</td>
</tr>
<tr>
<td>Begin Stimulus</td>
<td>905</td>
</tr>
<tr>
<td>BroadbandTuningSpan</td>
<td>906</td>
</tr>
<tr>
<td>Bucket Number</td>
<td>907</td>
</tr>
<tr>
<td>C0</td>
<td>908</td>
</tr>
<tr>
<td>C1</td>
<td>909</td>
</tr>
<tr>
<td>C2</td>
<td>910</td>
</tr>
<tr>
<td>C3</td>
<td>911</td>
</tr>
<tr>
<td>Cal Factor</td>
<td>912</td>
</tr>
<tr>
<td>ENRFile</td>
<td>1022</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>ENRID</td>
<td>1023</td>
</tr>
<tr>
<td>ENRSN</td>
<td>1024</td>
</tr>
<tr>
<td>Error Correction</td>
<td>1025</td>
</tr>
<tr>
<td>ErrorCorrection(Channel)</td>
<td>1026</td>
</tr>
<tr>
<td>External ALC</td>
<td>1027</td>
</tr>
<tr>
<td>ExternalTriggerConnectionBehavior</td>
<td>1028</td>
</tr>
<tr>
<td>ExternalTriggerDelay</td>
<td>1030</td>
</tr>
<tr>
<td>Filter BW</td>
<td>1031</td>
</tr>
<tr>
<td>Filter CF</td>
<td>1032</td>
</tr>
<tr>
<td>Filter Loss</td>
<td>1033</td>
</tr>
<tr>
<td>Filter Q</td>
<td>1034</td>
</tr>
<tr>
<td>FilterErrors</td>
<td>1035</td>
</tr>
<tr>
<td>FilterMode</td>
<td>1037</td>
</tr>
<tr>
<td>FirmwareMajorRevision</td>
<td>1038</td>
</tr>
<tr>
<td>FirmwareMinorRevision</td>
<td>1039</td>
</tr>
<tr>
<td>FirmwareSeries</td>
<td>1040</td>
</tr>
<tr>
<td>FixturingState</td>
<td>1041</td>
</tr>
<tr>
<td>FootSwitch</td>
<td>1042</td>
</tr>
<tr>
<td>Footswitch Mode</td>
<td>1043</td>
</tr>
<tr>
<td>Format (Marker)</td>
<td>1044</td>
</tr>
<tr>
<td>Format</td>
<td>978</td>
</tr>
<tr>
<td>Frequency</td>
<td>1046</td>
</tr>
<tr>
<td>Frequency Span</td>
<td>1047</td>
</tr>
<tr>
<td>FrequencyList</td>
<td>1048</td>
</tr>
<tr>
<td>Frequency Offset Divisor</td>
<td>1049</td>
</tr>
<tr>
<td>Frequency Offset Frequency</td>
<td>1050</td>
</tr>
<tr>
<td>Frequency Offset Multiplier</td>
<td>1051</td>
</tr>
<tr>
<td>Frequency Offset Override To CW</td>
<td>1052</td>
</tr>
<tr>
<td>Frequency Offset State</td>
<td>1053</td>
</tr>
<tr>
<td>Gate Shape</td>
<td>1054</td>
</tr>
<tr>
<td>Gate Type</td>
<td>1055</td>
</tr>
<tr>
<td>GPIBAddress</td>
<td>1056</td>
</tr>
</tbody>
</table>
GPIB Mode

GPIB Port Count

Handshake Enable

ID

ID String

IF Bandwidth Option

IF Bandwidth

IF Denominator

IFFilter Sample Count

IFFilter Sample Period

IFFilter Sample Period List

IFFilter Sample Period Mode

IFFilter Source Property

IF Gain Level

IF Gain Mode

IF Gate Enable

IFFrequency

IFFrequency Mode

IF Numerator

IF Sideband

IF Source Path

IF Start Frequency

IF Stop Frequency

Impedance States

Impulse Width

Index State

Input A

Input B

Input C

Input Denominator

Input Fixed Frequency

Input Is Greater Than LO

Input Numerator
InputPower 1090
InputRangeMode 1091
InputStartFrequency 1092
InputStopFrequency 1093
InternalTestsetPortCount 1094
Interpolate Correction 1095
Interpolated 1096
InterpolateNormalization 1097
Interrupt 1098
IsContinuous 1099
IsECALModuleFoundEx 1100
IsFrequencyOffsetPresent 1101
IsHold 1102
IsMarkerOn 1103
IsolationAveragingIncrement 1104
IsOn 1105
IsReceiverStepAttenuatorPresent 1106
IsReferenceBypassSwitchPresent 1107
IsSParameter 1108
IterationsTolerance 1109
Kaiser Beta 1110
L0 1111
L1 1112
L2 1113
L3 1114
Label 1115
Label Testset 1116
LastModified 1117
LimitTestFailed 1118
Limit Line Begin Stimulus 905
Limit Line End Stimulus 1119
Limit Line Begin Response 904
Limit Line End Response 1120
<table>
<thead>
<tr>
<th>Marker Type</th>
<th>1152</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marker X-axis Value</td>
<td>1153</td>
</tr>
<tr>
<td>Marker Y-axis Value</td>
<td>1154</td>
</tr>
<tr>
<td>Maximum Frequency</td>
<td>1156</td>
</tr>
<tr>
<td>Maximum Frequency (capabilities)</td>
<td>1157</td>
</tr>
<tr>
<td>Maximum Frequency (sourceCal)</td>
<td>1158</td>
</tr>
<tr>
<td>MaximumIFFilterSampleCount</td>
<td>1159</td>
</tr>
<tr>
<td>MaximumIFFrequency</td>
<td>1160</td>
</tr>
<tr>
<td>MaximumIterationsPerPoint</td>
<td>1161</td>
</tr>
<tr>
<td>MaximumNumberOfChannels</td>
<td>1162</td>
</tr>
<tr>
<td>MaximumNumberOfTracesPerWindow</td>
<td>1163</td>
</tr>
<tr>
<td>MaximumNumberOfWindows</td>
<td>1164</td>
</tr>
<tr>
<td>MaximumNumberOfPoints</td>
<td>1165</td>
</tr>
<tr>
<td>MaximumReceiverStepAttenuator</td>
<td>1166</td>
</tr>
<tr>
<td>MaximumSourceALCPower</td>
<td>1167</td>
</tr>
<tr>
<td>MaximumSourceStepAttenuator</td>
<td>1168</td>
</tr>
<tr>
<td>MaxPreciseTuningfilterations</td>
<td>1169</td>
</tr>
<tr>
<td>Mean</td>
<td>1170</td>
</tr>
<tr>
<td>Medium</td>
<td>1171</td>
</tr>
<tr>
<td>Minimum Frequency</td>
<td>1172</td>
</tr>
<tr>
<td>MinimumFrequency (capabilities)</td>
<td>1173</td>
</tr>
<tr>
<td>MinimumFrequency (sourceCal)</td>
<td>1174</td>
</tr>
<tr>
<td>MinimumIFFilterSampleCount</td>
<td>1175</td>
</tr>
<tr>
<td>MinimumIFFrequency Property</td>
<td>1176</td>
</tr>
<tr>
<td>MinimumNumberOfPoints</td>
<td>1177</td>
</tr>
<tr>
<td>MinimumReceiverStepAttenuator</td>
<td>1178</td>
</tr>
<tr>
<td>MinimumSourceALCPower</td>
<td>1179</td>
</tr>
<tr>
<td>Mode</td>
<td>1180</td>
</tr>
<tr>
<td>Multiplier</td>
<td>1181</td>
</tr>
<tr>
<td>Name (Calset)</td>
<td>1182</td>
</tr>
<tr>
<td>Name (CalKit object)</td>
<td>1183</td>
</tr>
<tr>
<td>Name config</td>
<td>1184</td>
</tr>
<tr>
<td>Name element</td>
<td>1185</td>
</tr>
</tbody>
</table>
OrientECALModule 1219
OutputFixedFrequency 1221
OutputPort 1222
OutputPorts calset 1224
OutputPorts 1225
OutputRangeMode 1226
OutputSideband 1227
OutputStartFrequency 1228
OutputStopFrequency 1229
Parameter 1230
Parameter_elo 1231
Parent 1232
PassFailLogic 1233
PassFailMode 1234
PassFailPolicy 1235
PassFailScope 1236
PassFailStatus 1237
Path Property 1238
PathCalMethod 1239
PathConfiguration 1241
PathThruMethod 1242
Peak Excursion 1244
Peak Threshold 1245
PeakToPeak 1246
Period 1247
Phase Offset 1248
Port 1 1249
Port 2 1250
Port 3 1251
Port2PdeembedCktModel 1252
Port2PdeembedState 1253
PortArbzImag 1254
PortArbzReal 1255
PortArbzState
PortArbzZ0
PortCatalog
PortCLogic
PortCMode
PortDelay
PortExtState
PortExtUse1
PortExtUse2
PortFreq1
PortFreq2
Port Label
PortLogic
PortLoss1
PortLoss2
PortLossDC
PortMatching_C
PortMatching_G
PortMatching_L
PortMatching_R
PortMatchingCktModel
PortMatchingState
PortMode
PortsNeedingDeltaMatch
Power Slope
Power Acquisition Device
PowerLevel
Power Meter Channel
Power Meter GPIBAddress
PowerOnDuringRetraceMode
PowerSweepRetracePowerMode
PreciseTuningTolerance
PreferInternalTriggerOnChannelSingle
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PreferInternalTriggerOnUnguidedCal</td>
<td>1290</td>
</tr>
<tr>
<td>RangeCount</td>
<td>1291</td>
</tr>
<tr>
<td>rangeNumber</td>
<td>1292</td>
</tr>
<tr>
<td>R1 Input Path</td>
<td>1293</td>
</tr>
<tr>
<td>Readings Per Point</td>
<td>1294</td>
</tr>
<tr>
<td>ReadingsTolerance</td>
<td>1295</td>
</tr>
<tr>
<td>ReadyForTriggerState</td>
<td>1296</td>
</tr>
<tr>
<td>Receiver Attenuator</td>
<td>1297</td>
</tr>
<tr>
<td>ReceiverCount</td>
<td>1298</td>
</tr>
<tr>
<td>ReceiverStepAttenuatorStepSize</td>
<td>1299</td>
</tr>
<tr>
<td>Receive Port</td>
<td>1300</td>
</tr>
<tr>
<td>ReduceIFBandwidth</td>
<td>1301</td>
</tr>
<tr>
<td>ReferenceCalFactor</td>
<td>1302</td>
</tr>
<tr>
<td>Reference Marker State</td>
<td>1303</td>
</tr>
<tr>
<td>Reference Level</td>
<td>1304</td>
</tr>
<tr>
<td>Reference Position</td>
<td>1305</td>
</tr>
<tr>
<td>RemoteCalStoragePreference</td>
<td>1306</td>
</tr>
<tr>
<td>ReverseLinearPowerLevel</td>
<td>1307</td>
</tr>
<tr>
<td>SafeSweepCoarsePowerAdjustment</td>
<td>1308</td>
</tr>
<tr>
<td>SafeSweepEnable</td>
<td>1309</td>
</tr>
<tr>
<td>SafeSweepFinePowerAdjustment</td>
<td>1310</td>
</tr>
<tr>
<td>SafeSweepFineThreshold</td>
<td>1311</td>
</tr>
<tr>
<td>SB_BalPortNegative</td>
<td>1312</td>
</tr>
<tr>
<td>SB_BalPortPositive</td>
<td>1313</td>
</tr>
<tr>
<td>SB_SEPrt</td>
<td>1314</td>
</tr>
<tr>
<td>SBalMeasurement</td>
<td>1315</td>
</tr>
<tr>
<td>Scope</td>
<td>1316</td>
</tr>
<tr>
<td>Search Function</td>
<td>1317</td>
</tr>
<tr>
<td>SecurityLevel</td>
<td>1318</td>
</tr>
<tr>
<td>Segment Number</td>
<td>1319</td>
</tr>
<tr>
<td>SelectPort</td>
<td>1320</td>
</tr>
<tr>
<td>Show Statistics</td>
<td>1321</td>
</tr>
<tr>
<td>ShowProperties</td>
<td>1322</td>
</tr>
<tr>
<td>Parameter</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>SICL</td>
<td>1323</td>
</tr>
<tr>
<td>SICLAddress</td>
<td>1324</td>
</tr>
<tr>
<td>Simultaneous2PortAcquisition</td>
<td>1325</td>
</tr>
<tr>
<td>SmartSweepMaximumIterations</td>
<td>1326</td>
</tr>
<tr>
<td>SmartSweepSettlingTime</td>
<td>1327</td>
</tr>
<tr>
<td>SmartSweepShowIterations</td>
<td>1328</td>
</tr>
<tr>
<td>SmartSweepTolerance</td>
<td>1329</td>
</tr>
<tr>
<td>Smoothing Aperture</td>
<td>1330</td>
</tr>
<tr>
<td>Smoothing ON/OFF</td>
<td>1331</td>
</tr>
<tr>
<td>SnPFormat</td>
<td>1332</td>
</tr>
<tr>
<td>Sound On Fail</td>
<td>1333</td>
</tr>
<tr>
<td>SourceCount</td>
<td>1334</td>
</tr>
<tr>
<td>Source Port</td>
<td>1335</td>
</tr>
<tr>
<td>SourcePortCount</td>
<td>1336</td>
</tr>
<tr>
<td>SourcePortMode</td>
<td>1337</td>
</tr>
<tr>
<td>SourcePortNames</td>
<td>1338</td>
</tr>
<tr>
<td>Source</td>
<td>1339</td>
</tr>
<tr>
<td>SourcePowerCalPowerOffset</td>
<td>1340</td>
</tr>
<tr>
<td>Source Power Correction</td>
<td>1341</td>
</tr>
<tr>
<td>Source Power Option</td>
<td>1342</td>
</tr>
<tr>
<td>Source Power State</td>
<td>1343</td>
</tr>
<tr>
<td>Span</td>
<td>1344</td>
</tr>
<tr>
<td>Span (Meas)</td>
<td>1345</td>
</tr>
<tr>
<td>SSB_BalPortNegative</td>
<td>1346</td>
</tr>
<tr>
<td>SSB_BalPortPositive</td>
<td>1347</td>
</tr>
<tr>
<td>SSB_SEPort1</td>
<td>1348</td>
</tr>
<tr>
<td>SSB_SEPort2</td>
<td>1349</td>
</tr>
<tr>
<td>SSBMeasurement</td>
<td>1350</td>
</tr>
<tr>
<td>Stage1Coefficients</td>
<td>1351</td>
</tr>
<tr>
<td>Stage1Frequency</td>
<td>1352</td>
</tr>
<tr>
<td>Stage1MaximumCoefficient</td>
<td>1353</td>
</tr>
<tr>
<td>Stage1MaximumCoefficientCount</td>
<td>1354</td>
</tr>
<tr>
<td>Stage1MaximumCoefficientSum</td>
<td>1355</td>
</tr>
</tbody>
</table>
strPortMatch_S2PFile
SweepEndMode
SweepHoldOff
Sweep Generation Mode
Sweep Time
SweepTimeOption
Sweep Type
System Impedance Z0
SystemName
Target Value
Test Port Power
TestSetType
Text
ThruCalMethod (FCA)
ThruCalMethod
ThruPortList
Title
Title State
TotalNumberOfPoints
Touchscreen
Trace Math
TraceTitle
TraceTitleState
Tracking
Transform Mode
Trigger Delay
TriggerInPolarity
TriggerInType
TriggerOutDuration
TriggerOutInterval
TriggerOutPolarity
TriggerOutPosition
TriggerOutputEnabled
<table>
<thead>
<tr>
<th>Feature</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TuningIFBW</td>
<td>1427</td>
</tr>
<tr>
<td>TuningMode</td>
<td>1428</td>
</tr>
<tr>
<td>TuningSweepInterval</td>
<td>1429</td>
</tr>
<tr>
<td>Trigger Mode</td>
<td>1430</td>
</tr>
<tr>
<td>Trigger Signal</td>
<td>1431</td>
</tr>
<tr>
<td>Trigger Type</td>
<td>1432</td>
</tr>
<tr>
<td>Type (calstd)</td>
<td>1433</td>
</tr>
<tr>
<td>Type_ts</td>
<td>1434</td>
</tr>
<tr>
<td>TZImag Property</td>
<td>1435</td>
</tr>
<tr>
<td>TZReal Property</td>
<td>1436</td>
</tr>
<tr>
<td>UnusedChannelNumbers</td>
<td>1437</td>
</tr>
<tr>
<td>USBPowerMeterCatalog</td>
<td>1438</td>
</tr>
<tr>
<td>UseCalWindow</td>
<td>1439</td>
</tr>
<tr>
<td>UsedChannelNumbers</td>
<td>1440</td>
</tr>
<tr>
<td>Use Power Loss Segments</td>
<td>1441</td>
</tr>
<tr>
<td>Use Power Sensor Frequency Limits</td>
<td>1442</td>
</tr>
<tr>
<td>User Range</td>
<td>1443</td>
</tr>
<tr>
<td>User Range Max</td>
<td>1444</td>
</tr>
<tr>
<td>User Range Min</td>
<td>1445</td>
</tr>
<tr>
<td>UserPresetEnable</td>
<td>1446</td>
</tr>
<tr>
<td>Valid</td>
<td>1447</td>
</tr>
<tr>
<td>ValidConnectorType</td>
<td>1448</td>
</tr>
<tr>
<td>Value element</td>
<td>1449</td>
</tr>
<tr>
<td>Values</td>
<td>1450</td>
</tr>
<tr>
<td>Velocity Factor</td>
<td>1451</td>
</tr>
<tr>
<td>View</td>
<td>1452</td>
</tr>
<tr>
<td>Visible</td>
<td>1453</td>
</tr>
<tr>
<td>WGCutoffFreq</td>
<td>1454</td>
</tr>
<tr>
<td>Width</td>
<td>1455</td>
</tr>
<tr>
<td>Window Number</td>
<td>1456</td>
</tr>
<tr>
<td>Window State</td>
<td>1457</td>
</tr>
<tr>
<td>XAxisAnnotation</td>
<td>1458</td>
</tr>
<tr>
<td>XAxis Point Spacing</td>
<td>1459</td>
</tr>
</tbody>
</table>
Methods

Abort 1465
AbortPowerAcquisition 1466
Acquire Cal Standard 1467
Acquire Cal Standard2 1469
AcquireCalConfidenceCheckECALEx 1471
AcquirePowerReadingsEx 1472
AcquireStep 1474
Activate 1475
Activate Marker 1476
Activate Window 1477
Add (channels) 1478
Add (measurement) 1479
Add (naWindows) 1482
Add (PowerLossSegment) 1483
Add (PowerSensorCalFactorSegment) 1484
Add (segments) 1485
Add Testset 1486
Allow All Events 1487
AllowChannelToSweepDuringCalAcquisition 1488
Allow Event Category 1489
Allow Event Message 1490
Allow Event Severity 1491
Apply 1492
ApplyDeltaMatchFromCalSet 1493
ApplyPowerCorrectionValuesEx 1494
ApplySourcePowerCorrectionTo 1495
AutoPortExtMeasure 1496
Delete ShortCut
Delta Marker
Disallow All Events
DisplayNAWindowDuringCalAcquisition
DisplayOnlyCalWindowDuringCalAcquisition
Do Print
DoECAL1PortEx
DoECAL2PortEx
DoneCalConfidenceCheckECAL
DoReceiverPowerCal
EnumerateCalSets
Execute
Execute Shortcut
GenerateGlobalDeltaMatchSequence
GenerateErrorTerms
GenerateSteps
Get AuxIO
Get Cal Standard
Get CalManager
Get CalSetByGUID
Get CalSetCatalog
Get CalSetUsageInfo
Get Cal Types
Get Complex
Get DataByString
Get Data
Get ECALModuleInfoEx
Get ErrorCorrection
Get Error Term
Get Error Term2
Get Error Term By String
Get Error Term Complex
Get Error Term Complex2
Get Error Term Complex By String
Get Error Term List
Get Error Term List2
Get ExtendedCallInterface
Get ExternalTestSetIO
Get Filter Statistics
Get Guid
get InputVoltageEX
Get Input1
Get IsolationPaths
Get MaterialHandlerIO
Get NAComplex
Get NumberOfGroups
Get Output
Get Output Voltage
Get OutputVoltage Mode
Get Paired Data
Get Port
Get PortC Data
Get PortNumber
GetRaw2DData
GetRaw2DDataIm
GetRaw2DDataRe
Get Reference Marker
Get Required Eterm Names
Get Scalar
Get Shortcut
Get SnPData
Get SnpDataWithSpecifiedPorts
Get SourcePowerCalDataEx
Get SourcePowerCalDataScalarEx
Get Standard
Get Standard By String
Get Standard Complex
Get Standard Complex By String
Get StandardsList
Get Standard List2
Get StandardsForClass
Get StepDescription
Get SupportedALCModes
Get Test Result
Get Trace Statistics
Get X-Axis Values
Get XAxisValues (Meas)
Get X-axis Values Variant
Has CalType
Hold
Hold (All Chans)
Initialize
Item
LaunchCalWizard
Launch Dialog
LaunchPowerMeterSettingsDialog
Load Configuration
LoadENRFile
LoadFile
Manual Trigger
MessageText
Next IF Bandwidth
Number of Groups
Open CalSet
Parse
Preset (app and chan)
Previous IF Bandwidth
Print To File
Put Complex
Put Data Complex 1670
PutENRData 1672
Put ErrorTerm 1673
Put ErrorTerm2 1675
Put Error Term By String 1676
Put ErrorTerm Complex 1677
Put ErrorTerm Complex2 1679
Put Error Term Complex By String 1681
Put Formatted Scalar Data 1682
Put NAComplex 1684
Put Output 1686
Put Output Voltage 1687
Put Output Voltage Mode 1688
Put Port 1689
Put PortCData 1691
Put Scalar 1692
Put Shortcut 1694
Put SourcePowerCalDataEx 1695
Put SourcePowerCalDataScalarEx 1696
Put Standard 1697
Put Standard By String 1699
Put Standard Complex 1700
Put Standard Complex By String 1702
Quit 1703
Read Data 1704
Read Raw 1705
Recall 1707
Recall Kits 1708
Remove 1709
Reset 1711
ResetLOFrequency 1712
ResetTuningParameters 1713
Restore Cal Kit Defaults 1714
<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restore Cal Kit Defaults All</td>
<td>1715</td>
</tr>
<tr>
<td>Resume</td>
<td>1716</td>
</tr>
<tr>
<td>Save</td>
<td>1717</td>
</tr>
<tr>
<td>Save (CalSet)</td>
<td>1719</td>
</tr>
<tr>
<td>Save CalSets</td>
<td>1720</td>
</tr>
<tr>
<td>SaveCitiDataData</td>
<td>1721</td>
</tr>
<tr>
<td>SaveCitiFormattedData</td>
<td>1722</td>
</tr>
<tr>
<td>SaveENRFile</td>
<td>1723</td>
</tr>
<tr>
<td>Save File</td>
<td>1724</td>
</tr>
<tr>
<td>Save Kits</td>
<td>1725</td>
</tr>
<tr>
<td>Search Filter Bandwidth</td>
<td>1726</td>
</tr>
<tr>
<td>Search Max</td>
<td>1727</td>
</tr>
<tr>
<td>Search Min</td>
<td>1728</td>
</tr>
<tr>
<td>Search Next Peak</td>
<td>1729</td>
</tr>
<tr>
<td>Search Peak Left</td>
<td>1730</td>
</tr>
<tr>
<td>Search Peak Right</td>
<td>1731</td>
</tr>
<tr>
<td>Search Target</td>
<td>1732</td>
</tr>
<tr>
<td>Search Target Left</td>
<td>1733</td>
</tr>
<tr>
<td>Search Target Right</td>
<td>1734</td>
</tr>
<tr>
<td>SelectCalSet</td>
<td>1735</td>
</tr>
<tr>
<td>Set All Segments</td>
<td>1736</td>
</tr>
<tr>
<td>Set BBPorts</td>
<td>1738</td>
</tr>
<tr>
<td>Set Cal Info</td>
<td>1739</td>
</tr>
<tr>
<td>SetCallInfoEx</td>
<td>1741</td>
</tr>
<tr>
<td>Set Center</td>
<td>1743</td>
</tr>
<tr>
<td>Set CW</td>
<td>1744</td>
</tr>
<tr>
<td>Set Electrical Delay</td>
<td>1745</td>
</tr>
<tr>
<td>Set FailOnOverRange</td>
<td>1746</td>
</tr>
<tr>
<td>Set IsolationPaths</td>
<td>1747</td>
</tr>
<tr>
<td>Set PowerAcquisitionDevice</td>
<td>1749</td>
</tr>
<tr>
<td>Set Frequency LowPass</td>
<td>1750</td>
</tr>
<tr>
<td>SetPortMap</td>
<td>1751</td>
</tr>
<tr>
<td>Set Reference Level</td>
<td>1752</td>
</tr>
</tbody>
</table>
Set SBPorts
Set SSBPorts
SetupMeasurementsForStep
Set StandardsForClass
Set Start
Set Stop
Show Marker Readout
Show Status Bar
Show Stimulus
Show Table
Show Title Bars
Show Toolbar
Single
Store
StoreConfiguration
StringToNACalClass
StringtoNAErrorTerm2
SweepOnlyCalChannelDuringCalAcquisition
TestsetCatalog
UserPreset
UserPresetLoadFile
UserPresetSaveState
Write Data
Write Raw
WriteSnpFileWithSpecifiedPorts

Events
OnCalEvent
OnChannelEvent
OnDisplayEvent
OnHardwareEvent
OnMeasurementEvent
OnSCPIEvent
OnSystemEvent
Examples

CalSet_Examples
Getting Trace Data from the Analyzer
Perform a Guided Cal using COM
Perform an ECal
Perform a Source Power Cal
Perform an Unguided Cal using COM
Perform an Unknown Thru or TRL Cal
Perform Global Delta Match Cal
Writing Cal Set Data using COM
Upload a Source Power Cal
Upload Segment Table
Create and Cal an SMC Measurement
Create and Cal a VMC Measurement
Create an SMC Fixed Output Meas
Create a Pulsed Measurement
Create a Balanced Measurement
Perform an ECAL Confidence Check
Limit Line Testing Example with COM
E5091Testset Control
Errors and the SCPIStringParser Object
External Testset Control
PathConfiguration Example
Events Example

Concepts

Configure for COM-DCOM Programming
COM Fundamentals
Getting a Handle to an Object
Collections in the Analyzer
COM Data Types
PNA Automation Interfaces
Working with the Analyzer's Events
Read and Write Calibration Data using COM
C and the COM Interface
Using .NET

SCPI

Commands

SCPI Command Tree
Common Commands
Abort
Calculate
  Correction
  Custom
  Data
  Equation
  Filter
  Format
FSimulator
Function
GCDa ta
Limit
Marker
Math
Mixer
Normalize
Offset
Parameter
RData
Smoothing
Transform
Control
Display
Format
Hardcopy
Initiate
Memory 2005
Output 2014
Route 2016
Sense
  Average 2017
  Bandwidth 2019
Correction
  Correction 2021
  Guided Cal 2046
  Cal Kit 2066
  Cal Stds 2074
  Cal Sets 2095
  Extensions 2109
  Session 2118
  SMC 2124
  VMC 2132
Couple 2142
FOM 2144
FOMSegment 2153
Frequency 2164
Gain Compression 2168
IF 2183
IF (PNA-X) 2188
Mixer 2197
MixerEmbedLO 2215
Multiplexer 2229
Noise 2240
Offset 2248
Path 2252
Power 2257
Pulse 2258
Roscillator 2262
Segment 2263
Sweep 2273
XAxis 2282
Source 2283
Source Correction 2294
Status 2310
System 2326
Trigger 2347

Examples
Catalog Measurements 2361
Create an FOM Measurement 2362
Create an S-Parameter Measurement 2365
Create a Balanced Measurement 2366
Channels, Windows, and Measurements using GPIB 2371
Setup Sweep Parameters 2373
Setup the Display 2374
Triggering the PNA 2376
GPIB using Visual C 2381
Guided 2-Port or 4-Port Cal 2385
Guided 2-Port Comprehensive Cal 2388
Guided ECal 2394
Guided Mechanical Cal 2396
Guided 1-Port on Port 2 2398
Guided TRL Calibration 2400
Guided Unknown Thru or TRL Cal 2402
Global Delta Match Cal 2404
Unguided ECAL 2405
Unguided 2-Port Mech Cal 2406
Unguided 1-Port Cal on Port 2 2408
Unguided 2-Port Cal on a 4-Port PNA 2410
Unguided_Cal_on_Multiple_Channels 2419
ECAL Confidence Check 2423
Source and Receiver Power Cal 2426
Upload a Source Power Cal 2435
Perform a Sliding Load Cal 2439
Load Eterms during Cal Sequence 2440
Create New Cal Kit 2441
Modify a Calibration Kit 2447
Create and Cal a VMC Measurement 2449
Create and Cal an SMC Measurement 2452
Create an SMC Fixed Output Measurement 2454
Create and Cal a GCA Measurement 2457
Create and Cal a Noise Figure Measurement 2462
Show Custom Window during Calibration 2465
Getting and Putting Data 2468
Getting and Putting Data (definite block transfers) 2470
Control an External Test Set 2473
Transfer Data using GPIB 2475
Establish a VISA Session 2477
Status Reporting 2479
Create a Custom Power Meter Driver 2481
GPIB Pass Through 2485
PNA as Controller and Talker/Listener 2486
Send SCPI Commands using a Socket Client 2488
Concepts
GP-IB Fundamentals 2492
The Rules and Syntax of SCPI Commands 2497
How to Configure for GPIB, SCPI, and SICL 2501
Getting Data from the Analyzer 2505
Understanding Command Synchronization 2509
Calibrating the PNA Using SCPI 2514
The PNA as a USB Device 2519
Reading the Analyzer’s Status Registers 2520
Configure for VISA and SICL 2523
VEE Examples
VEE Pro Runtime 2526
Basic Control VEE 2527
3.8 GHz Frequency Adjust
10 MHz Reference Adjust
Display Test
LO Power Adjust
Offset LO Power Adjustment
Operators Check
Option H11 Verification
Phase-Lock IF Gain Adjust
System Verification
Source Cal
Receiver Cal
Receiver Display

IF Access
IF Access User Interface Settings
External Test Head Configuration
PNA-X IF Path Configuration

Controlling External Devices
E5091 Test Set Control
External Testset Control
Interface Control
USB to GPIB Adapter
Aux IO Connector
Handler IO Connector
Test Set IO Connector
Power IO Connector (PNA-X)
Configure an External LO Source
Synchronize an External PSG Source

Specifications
E8356A, 57A, 58A
E8801A, 02A, 03A
N3381A, 82A, 83A
E8361A
E8362A, 63A, 64A
<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>E8362B, 63B, 64B</td>
<td>2831</td>
</tr>
<tr>
<td>N5230A 2-Port</td>
<td>2877</td>
</tr>
<tr>
<td>N5230 4-Port</td>
<td>2955</td>
</tr>
<tr>
<td>N5242A</td>
<td>2990</td>
</tr>
<tr>
<td>N5250A</td>
<td>3110</td>
</tr>
<tr>
<td>Glossary</td>
<td>3117</td>
</tr>
</tbody>
</table>
What’s New in PNA Code Version A.08.00

- Noise Figure Application (Opt 029)
- Gain Compression Application (Opt 086)
- ‘Sweep’ Trigger Mode
- Custom Cal Window settings (remote only)
- New Equation Editor Functions
- Minimum Number of Points = 1

See New 8.0 Programming Commands
To check your PNA code version, click Help, then About Network Analyzer

What’s New in PNA Code Version A.07.50

- USB / LAN power sensor support
- Increased Number of Points to 20,001
- ‘Extra’ Security Setting
- Wideband Pulsed Application (PNA-X)
- Expanded right-click mouse capabilities (PNA-X)
- IF Path Configuration for all receivers (PNA-X)
- External Source Control
  - Consistency improvements
  - Generic (Non-Agilent) sources are NOT supported in this release. This could result in errors in remote programs.
- Copy Source Power Cal Macro

Note: This firmware revision can be installed on ALL PNA models that use Windows XP.

Highlighted text on this page (until 6.04) describes features that are NEW for most PNA models. These features have already been released for PNA-X models.

See New 7.50 Programming Commands
What's New in PNA Code Version 7.22

- Enhanced Response Calibration
- PNA-X Support for Millimeter-wave (Please read CAUTION)
- External Source Control for ALL measurement types
- Embedded LO Measurements
- ADC Receiver Measurements

What's New in PNA Code Version 7.21

- Wider IF Bandwidths (PNA-X Only)
- Produce receiver power calibration of PNA reference receiver
- Isolation Cal (SCPI and COM only)

What's New in PNA Code Version 7.20

New PNA-X models includes the following features:

- Internal Second Source (some models)
- Improved Front-Panel User Interface
  - 10.4 inch Hi resolution LCD Touchscreen
  - Fully functional Hardkey / Softkey layout
  - Trace Zoom
  - Trace Max - isolates a single trace
  - Marker Drag with mouse or touchscreen
  - Expanded Right-click mouse capabilities
  - Custom Trace Titles
  - Memory Normalize
- Increased Rear-Panel Capabilities
  - Auxiliary Triggering
- Pulse I/O
- Power I/O
- IF Path Configuration
- RF Path Configurator
- Measurement Classes
- Source and Receiver Attenuation Offset
- Updated Pulsed Application
- True Mode Stimulus Application (Webpage)

New PNA Preferences Help Topic

See New 7.20 Programming Commands

---

**What's New in PNA Code Version 7.1**

- New PNA-L 4-port models
- FOM and Power dialog support for 2 Internal Sources

See New 7.1 Programming Commands

---

**What's New in PNA Code Version 6.2**

- Option 551 Multiport Test Set Control
- QSOLT Calibration Method
- Calibration Preferences
- Unlimited number of windows
- Source Power Cal using a PNA receiver only
- Choose ports for saving sNp files
- Channel Trigger State added to Status Bar
- FCA and Cal Set viewer data can be saved to *.prn files
- Cal Channel created for performing calibrations

New in PNA Help:
What's New in PNA Code Version 6.04

- Updated Print and Page Setup dialog
- Selectable Power Sweep retrace level
- Turn power OFF during a retrace in single band sweep.
- Inverse Smith and Unwrapped Phase added to PNA display formats.
- Opt. 082 SMC Measurements
- Equation Editor
- Remote SCPI over LAN from non-windows PC using Sockets/Telnet
- Citifiles recalled to channel 32 and below.
- ECal User-Characterization allowed beyond ECal module frequency range

New in PNA Help:

- FCA Measurement Examples (VMC and SMC)
- Comparing the PNA Delay Functions
- PNA Online Web Help

What's New in PNA Code Version 6.0

- Calibrate using an External Trigger Source
  (This could affect your remote programs.)
- Calibrate with an Offset Load Standard
  (Cal Kits that you use may now include this standard.)
- Corrected Measurement visible in Cal Window
- External Testset Control
- New FCA capabilities
- **Characterize Adaptor Macro** creates S2P files from two 1-port Cal Sets.

- **1.1 GHz CPU and related capabilities**

- **Error-checking and Disk Defragmenter recommendation**

- **Agilent VEE Runtime Installed**

- ***.csa file type is default for Save As** and **Auto Save**

- **Bandwidth Markers search for "Valley" response**

- **SimCal SCPI Preference**

- **Application Code (software) Revision number** now contains 6 digits instead of 4. (This could affect your remote SCPI and COM programs.)

- Rev 6.0 is NOT supported on PNA models using Windows 2000. For more information, see the [PNA support website](#).

- Rev 6.0 is NOT supported on PNA models N3381A, N3382A, N3383A.

Last modified:

9/28/06 Cross-browser
PNA User Accounts and Passwords

When the PNA power is switched on, it automatically logs into Windows using the default user name and password. This gives anyone full access to the analyzer. The following steps can be taken to increase security of your PNA.

- Require users to logon when the PNA computer is turned ON - Learn how to enable this feature
- Setup individual accounts on the PNA with varying level of access - Learn how to Add or Change User Accounts and Passwords

Please read about Anti-virus protection for your PNA

Existing User Accounts

The following user accounts already exist on new PNAs:

- **Default User Account**
  Beginning in April 2004, PNAs were shipped from the factory with the default user name is PNA-Admin and the password is agilent.
  For PNAs shipped before that, the default user name is Administrator and the password is either tsunami or left blank.
  These accounts are created by Windows and cannot be deleted.
  We recommend you change the password and, if desired, the user name.
  DO NOT FORGET YOUR NEW PASSWORD. You will not be able to start your PNA without it.

- **Agilent Account** This Administrator account is created by Agilent for service purposes. Each PNA has a unique password for this account. Although allowed by Windows, please do not delete this account.

- **Guest Account** This account allows anyone to type in any name, without password, and gain limited access to the PNA files. This account is created by Windows and cannot be deleted. It can be renamed. This account is turned OFF when the PNA is shipped.

Notes

- Although allowed by Windows, do NOT setup an Administrator account without a password. Internet viruses look for, and exploit, this condition.

- You can create as many user accounts as you like.

- The user name is not case sensitive. The password IS case sensitive.

- The PNA local policies are set so that, if logon is required, you must retype the user name (and password) every time. Do not change the local policies on the PNA.

How to Require Users to Logon when the PNA Computer is turned ON.

How do I know which Operating System I have?
On the Windows taskbar, click **Start**, then **Settings**, then **Control Panel**.

Double click **Users and Passwords**

Check **Users must enter a user name and password to use this computer.**

To turn this function OFF, perform the same procedure, but clear the checkbox. The account that is selected when the checkbox is cleared is the account that is automatically logged on when the PNA is turned ON.

### Add or Change User Accounts and Passwords

If the analyzer is in a secure environment, you can setup PNA users by name and grant various levels of access. This is particularly important when the PNA is remotely controlled or accessed over LAN.

You can designate a person as the administrator and then configure the PNA to allow others to use it with reduced permissions. That is, other people can be signed on to use the analyzer but they will not have the ability to perform all of the administrative functions that you can as the administrator.

### How to add or change a user account and password

#### How do I know which Operating System I have?

<table>
<thead>
<tr>
<th>Windows 2000</th>
<th>Windows XP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>In the analyzer System menu, point to Configure, and click Control Panel.</strong></td>
<td><strong>Click Start, then point to Settings, then click Control Panel</strong></td>
</tr>
<tr>
<td><strong>In the Control Panel window, scroll down and select the Users and Passwords application.</strong></td>
<td><strong>Click User Accounts</strong></td>
</tr>
<tr>
<td><strong>On the Users tab, if the Add button appears dimmed, select the Users must enter a user name and password to use this computer check box near the top of the window.</strong></td>
<td><strong>Follow the prompts to:</strong></td>
</tr>
<tr>
<td></td>
<td>• Change an account</td>
</tr>
<tr>
<td></td>
<td>• Create a new account</td>
</tr>
<tr>
<td></td>
<td>• Change the way users log on or off.</td>
</tr>
<tr>
<td><strong>Click Add to enter the information for yourself or for another user.</strong></td>
<td><strong>CAUTION:</strong> Although allowed by Windows, do NOT allow an Administrator account without a password. Internet viruses look for, and exploit, this condition.</td>
</tr>
<tr>
<td><strong>In the User name box, enter a user name for the user. In the Full name box, enter the full name of the user.</strong></td>
<td></td>
</tr>
</tbody>
</table>
In the **Description** box, enter a description for the user. Then, click **Next**.

In the **Password** box, have the user type a password. Have the user retype the password in the **Confirm password** box. Then, click **Next**.

Select the level of access that you wish to grant this user.

**Note:** Standard users and restricted users are **NOT** able to switch GPIB modes and install firmware.

There are several other levels of security that you may grant in the **Other** list. A description of each of these other levels is displayed beneath the **Other** box when it is selected. Then, click **Finish**.

**Note:** Standard users and restricted users are **ABLE** to switch GPIB modes and install firmware.

In the **Users for this computer** box, validate the user name and security level group of the user.

If you want this user to be able to use the network analyzer without entering their password each use, clear the **Users must enter a user name and password to use this computer** check box. Click **OK**.

When the **Automatically Log On** window is displayed, have the new user type their password in the **Password** box and have them retype the password in the **Confirm Password** box.

Click **OK** to complete this user addition.

In the **File** menu, click **Close** to close the Control Panel.
PNA Computer Properties

The PNA uses a personal computer and a Windows operating system. The following are common tasks that you may need to perform on the PNA computer.

- **View or change Full Computer Name**
- **Check IP Address**
- **Check the amount of RAM**
- **Check CPU Speed**
- **Set Time and Date**
- **Turn OFF Speaker**

### Other Administrative Task Topics

**View or change Full Computer Name**

Your PNA has a unique computer name that identifies it on a network. To view or change the computer name, you must first minimize the PNA application.

**How do I know which Operating System I have?**

<table>
<thead>
<tr>
<th>Windows 2000</th>
<th>Windows XP</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the desktop, right-click <strong>My Computer</strong></td>
<td>On the desktop, right-click <strong>My computer</strong> Icon</td>
</tr>
<tr>
<td>Click <strong>Properties</strong></td>
<td>Click <strong>Properties</strong></td>
</tr>
<tr>
<td>Click the <strong>Network Identification</strong> tab at the top of the dialog box</td>
<td>Click the <strong>Computer Name</strong> tab at the top of the dialog box</td>
</tr>
<tr>
<td>Click <strong>Properties</strong></td>
<td>Click <strong>Change</strong> next to &quot;..rename this computer..&quot; message</td>
</tr>
<tr>
<td>Type your new <strong>Computer Name</strong></td>
<td>Type your new <strong>Computer Name</strong></td>
</tr>
</tbody>
</table>

**Note:** To add your computer to a domain, or to set up the networking configuration, contact your company's I.T. department. This setup is custom for each company.

To restore the PNA application, click **PNA Analyzer** in the task bar at the bottom of the screen.

**Check IP Address**

If your PNA is connected to a LAN, you can view the IP address and other networking information.
1. Minimize the PNA application

2. Click Start, then Run

3. Type cmd, then click OK

4. At a DOS prompt, type ipconfig /all

Check the amount of RAM

Random Access Memory (RAM) is the amount of working memory in your computer. The PNA application can require up to 512 MB of RAM depending on the settings you use. If your PNA is operating slowly when you have more than four windows open or if you routinely use more than 1601 data points, you may need to upgrade to 512 MB.

To view the amount of PNA RAM, you must first minimize the PNA application.

How do I know which Operating System I have?

<table>
<thead>
<tr>
<th>Windows 2000</th>
<th>Windows XP</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the desktop, right-click My Computer</td>
<td>On the desktop, right-click My computer Icon</td>
</tr>
<tr>
<td>Click Properties</td>
<td>Click Properties</td>
</tr>
<tr>
<td>Click the General tab at the top of the dialog box</td>
<td>Click the General tab at the top of the dialog box</td>
</tr>
<tr>
<td>The amount of RAM appears at the bottom of the window.</td>
<td>The amount of RAM appears at the bottom of the window.</td>
</tr>
</tbody>
</table>

To restore the PNA application, click PNA Analyzer in the task bar at the bottom of the screen.

Check CPU Speed

The speed of the PNA processor (CPU) is a factor in determining how quickly the PNA processes data. See PNA configurations to learn if you can upgrade your PNA CPU. To check your PNA CPU speed, you must first minimize the PNA application.

How do I know which Operating System I have?
<table>
<thead>
<tr>
<th>Windows 2000</th>
<th>Windows XP</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the desktop, right-click <strong>My Computer</strong></td>
<td>On the desktop, right-click <strong>My computer</strong> Icon</td>
</tr>
<tr>
<td>Click <strong>Manage</strong></td>
<td>Click <strong>Properties</strong></td>
</tr>
<tr>
<td>Open <strong>System Tools</strong> folder, then click <strong>System Information.</strong></td>
<td>Click the <strong>General</strong> tab at the top of the dialog box</td>
</tr>
<tr>
<td>Click <strong>System Summary.</strong></td>
<td>The CPU speed appears near the bottom of the window</td>
</tr>
<tr>
<td>After refreshing, the CPU speed appears at the end of the <strong>Processor</strong></td>
<td></td>
</tr>
</tbody>
</table>

To restore the PNA application, click **PNA Analyzer** in the task bar at the bottom of the screen.

**Set Time and Date**

Both Windows 2000 and XP

To set the time and date on your PNA, you must first minimize the PNA application.

1. Move the cursor to the lower corner of the screen
2. When the taskbar appears, double-click on the displayed time. This opens the **Date/Time Properties** dialog box.
3. Change the date, time, and time zone as appropriate.

To restore the PNA application, click **PNA Analyzer** in the task bar at the bottom of the screen.

**Turn OFF | ON Speaker**

When the PNA is generating errors, you may want to turn the speakers off to quiet the beeping. [Learn more about errors.]

1. To turn ON or OFF the PNA speaker, you must first minimize the PNA application.
2. Then click **Start, Control Panel,** then **Sounds and Audio Devices.**
3. Under **Device Volume,** check **Mute.**
Run Error Check and Disk Defragmenter

When the PNA is shutdown unexpectedly or power is removed without first shutting down, large amounts of Hard Disk Drive space is rendered unusable. If shutdown in this manner enough times, the PNA could become unstable and no longer work.

This Hard Disk Drive space can be recovered by first running Windows Error-checking to find and correct errors on the disk, and then the Disk Defragmenter to recover Hard Disk Drive space. These programs should be run routinely, about every 1 to 4 weeks, depending on how often the PNA is unexpectedly shutdown.

To learn more about Disk Defragmenter, see the Windows Help file.

Follow this procedure to run these programs:

<table>
<thead>
<tr>
<th>Windows 2000</th>
<th>Windows XP</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the desktop, double-click My Computer</td>
<td>On the desktop, double-click My Computer</td>
</tr>
<tr>
<td>Select Local Disk (C:)</td>
<td>Select System OS</td>
</tr>
<tr>
<td>Click File, then Properties</td>
<td>Click File, then Properties</td>
</tr>
<tr>
<td>Click the Tools tab</td>
<td>Click the Tools tab</td>
</tr>
</tbody>
</table>

### Error-checking

- Click Check Now.
- Check Automatically fix file system errors.
- Click Start.
- Click Yes to run disk check on next restart.
- Manually restart the PNA. The disk check will run before Windows restarts.

Approximately every six months, check the second box in addition to the first box. The error-checking process takes much longer, but performs a more complete check.

### Defragmentation

- Click Defragment Now...
- Click Defragment to begin the defragment process.
- Click Close when defragmentation is complete.
Recovering from PNA Hard Drive Problems

The leading cause of PNA failures is problems with the PNA Hard Disk Drive (HDD). These problems are usually preventable (see Preventing PNA HDD Problems), and in many cases, recoverable. The following could save you weeks of downtime and the cost of replacing your PNA HDD.

This document is now on the Agilent PNA Support Website: http://na.tm.agilent.com/pna/. When at this webpage, click the Hard Drive Recovery link.

If your PNA does experience a Hard Disk Drive Problem, you will not be able to access this Help file, but you may be able to access the Internet from another computer.
Microsoft Windows® XP / 2000 Considerations

In this topic:

- Microsoft Windows on the PNA
- Using USB
- Plug & Play Stability and Security
- LAN Connections
- Single and Double Click option
- Windows XP Theme
- Printing

Microsoft Windows on the PNA

- Beginning in April 2004, the PNA is shipped from the factory with a modified version of Microsoft Windows XP operating system. Previously, the PNA was shipped with Windows 2000. The PNA application performs identically using these two operating systems.

- Beginning in Dec. 2005 with PNA Rev 6.0, firmware cannot be upgraded on PNA models that use Microsoft Windows 2000. For more information, see the PNA support website.

To determine which Operating System is installed on your PNA:

1. Minimize the PNA application
2. On the PNA desktop, click Start.
3. Along the side of the Start menu appears one of the following:
   - Windows 2000 Professional
   - Windows XP Professional

VERY IMPORTANT  Protect your hard drive!

The leading cause of PNA failures is problems with the PNA Hard Disk Drive (HDD). These problems are usually preventable, and in many cases, recoverable. Learn more about protecting your PNA.

Using USB

The PNA has at least two USB ports for connecting devices: one on the front panel and at least one on the rear.
The main advantages of USB are “hot” connects and disconnects and fast data transfer speeds. Electronic Calibration modules are now available with USB connections.

The first time you plug a device into a USB port there is some wait time. Windows reports it is identifying the hardware, then searching for the correct driver, then installing the driver (if it was found).

Connecting that same device back into that same port later is quick and easy, but if you move the device to a different USB port, you will have to wait through the hardware ID and driver search again.

Learn about USB limitations.

Note: Certain USB devices (such as ECAL modules) require you be logged on with Administrator privileges the first time you plug them into the PNA. This must be done for each serial number. Click Next to choose the default settings when installing new USB devices.

Plug & Play Stability and Security
Plug & Play capabilities is similar to Win 95 and 98. It provides both a stable and secure operating environment. You may notice also that it greatly reduces the number of required reboots.

LAN Connections
Windows supports DHCP and fixed IP addressing. Also, “Hot” connect and disconnect of the LAN cable, as well as a visual indicator of LAN status in system tray area, makes LAN connections more intuitive. In addition, the Hardware Wizard helps users with system hardware configuration.

Single and Double Click option
By default, Windows allows a single-click method of launching icons. To revert to double-clicking, click Start, then Settings, then Control Panel, then click Mouse. In the Mouse Properties dialog, select Double-click to open an item. Then click OK.

Windows XP Theme
The PNA application is designed for, and best viewed in, Windows Classic theme. To change the theme from Windows XP to Windows Classic,

1. Minimize the PNA application.
2. Right-click on the Desktop, then click Properties.
3. On the Theme tab, under Theme select Windows Classic.

Printing
Adding a printer should be done outside of the PNA application. Learn more.
Features, Requirements, and Limitations

Features

- Fast, easy, and complete Gain Compression measurements for amplifiers.
- Many compression parameters to choose from, including gain, input power at compression, output power at compression, input match, and compression level.
- Several compression methods to choose from, including deviation from linear gain, deviation from max gain, back-off, and X/Y.
- Three acquisition methods to choose from: Power per Freq, Freq per Power, and SMART Sweep.
- SMARTCal Calibration Wizard to guide you through Full 2-Port or Enhanced Response calibration, plus Source Power calibration.
- Supports Frequency domain Wideband Pulse measurements. However, Time domain measurements are NOT supported.

Requirements
- PNA-X with Opt 086 (software option only) must be enabled.

- When performing an optional calibration:
  - ECal module or Calibration Kit
  - Power meter and power sensor

Limitations with GCA

- Number of points limited to 20,001 for two-dimensional acquisitions, 10,000 points for SMART Sweep.

- Standard CW power sweep is NOT supported in a Gain Compression channel.

- No Independent IFBW, Independent power levels in segment table.

- Stepped sweep mode only.

- Linear, Log, and Segment frequency sweep modes only.

- 2-port DUTs ONLY.

The following PNA Features are NOT Available in a Gain Compression channel:

- **Unratioed receiver measurements** (A, B, R)

- **ECal User Characterization**

- **Fixture Deembedding**

- **FOM** or **FCA**

- **External Test Set Control** (Option 551)

- **Interface Control**

- **Copy Channel**

- **IF Path Configuration**

- **Time Domain**

- **Equation Editor** - Equation traces are not allowed in the GCA channel, but GCA traces can be referenced in an equation trace that resides in a standard channel. Learn more.

- **Port extensions**

- **Balanced measurements**

- **Point and Sweep trigger**

- Save **Formatted Citifile** data.
• Time Domain Pulse measurements in the **Wideband Pulse** App are NOT supported.

**Gain Compression Application Concepts**

**What is Gain Compression**

An amplifier has a region of linear gain, where the gain is independent of the input power level. This gain is commonly referred to as small signal gain. As the input power is increased to a level that causes the amplifier to approach saturation, the gain will decrease. The 1 dB gain compression is defined as the input power level that causes amplifier gain to drop 1 dB relative to the linear gain.

![Gain Compression Diagram](image)

**Terms used in GCA**

**Linear Power Level**  The specified input power that yields linear gain (also known as 'small-signal gain') in the amplifier.

**Reference gain** The measured gain that is used as a reference for determining compression level. The **Compression Method** that is used could cause this value to be different.

**Compression level**  The specified amount of gain reduction from the reference gain.

**Target gain** The gain at the specified compression level. Although this term does not appear in GCA, it is important to understand when discussing the various compression parameters.

For example, when using **Compression from Linear Gain** method with the following settings:

- Linear gain (measured at Linear Input power) = 10.2 dB
- Compression level (specified) = 1 dB
- Target gain = 9.2 dB

This is called 'Target' gain because GCA will search for the closest measured gain to 9.2000 dB. It may not measure this gain exactly.

**Compression point** The operating point at which the measured gain is closest to the Target Gain. All **compression parameters** report data for this operating point.

**Understanding the GCA Displayed Traces**

One of the most important concepts to remember with GCA is that, each frequency data point represents many measurements using different input power levels.
Some things to notice about how GCA displays compression data:

1. The X-axis values are ALWAYS frequency. Imagine behind each frequency data point, a traditional power sweep curve with corresponding measurements and calculations to find the specified compression point.

2. The Y-axis values are always reported at the compression point. The value that is displayed depends on the compression parameter that you choose. The S-parameters that are displayed in a GCA channel are always measured at the linear and reverse power level.

Example: Five of the six GCA compression parameters are displayed in the above image. The missing trace, DeltaGain21 is discussed below.

- Markers are placed at 4.549 GHz for all of the parameters.
- Tr 3 CompIn21 (Input power at the compression point) shows the marker value to be -5.4117 dBm. This is the power into the DUT that was required to achieve the compression point. Notice that this is about the same input power required to achieve the specified compression at ALL frequencies.
- Tr 5 CompGain21 (Gain at the compression point) shows the marker value 9.6443 dB. This is the measured gain at the compression point.
- It is NOT possible to see the gain at a different input power at this frequency by viewing a GCA compression parameter. The compression parameters display values ONLY at the compression point. However, this data CAN be viewed by saving saving 2D data to a csv file, or displayed on the PNA by running a macro at that single frequency.

Gain Compression Parameters

There are several Gain Compression parameters, as well as standard S-parameters that can be measured in a GCA channel.
How to add or change GCA Parameters

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>PNA-X</strong> and <strong>'C' models</strong></td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press <strong>MEAS</strong></td>
<td>1. click <strong>Response</strong></td>
</tr>
<tr>
<td>2. then select a parameter</td>
<td>2. then <strong>Measure</strong></td>
</tr>
<tr>
<td></td>
<td>3. then select a parameter</td>
</tr>
</tbody>
</table>

**Linear S-Parameters**

For convenience, the standard S-parameters are offered in a GCA channel. S11 and S21 are measured at the specified Linear Input level. S22 and S12 are measured at the specified Reverse power level.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>When Measured</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td>Input Match</td>
<td>Always</td>
</tr>
<tr>
<td>S21</td>
<td>Gain</td>
<td>Always</td>
</tr>
<tr>
<td>S22</td>
<td>Output Match</td>
<td>See Reverse</td>
</tr>
<tr>
<td>S12</td>
<td>Reverse Isolation</td>
<td>See Reverse</td>
</tr>
</tbody>
</table>

**Compression Parameters**

*Note:* The following table assumes: DUT Input = PNA port 1 and DUT Output = PNA port 2.

When the Port mapping is different, the parameters in GCA are updated accordingly. For example, with Input = port 2 and Output = port 1, then "Compln12" would be displayed.

The raw data for these parameters are always measured.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompIn21</td>
<td>Input power at the compression point.</td>
</tr>
<tr>
<td>CompOut21</td>
<td>Output power at the compression point.</td>
</tr>
<tr>
<td>CompGain21</td>
<td>Gain at the compression point.</td>
</tr>
<tr>
<td>CompS11</td>
<td>Input Match at the compression point.</td>
</tr>
<tr>
<td>RefS21</td>
<td>Linear Gain value used to calculate the compression level. This is calculated differently depending on the compression method.</td>
</tr>
<tr>
<td>DeltaGain21</td>
<td>CompGain21 MINUS Linear Gain (in Log Mag format). This trace can be used to learn a lot about the DUT compression point. [Learn more]</td>
</tr>
</tbody>
</table>

**Compression Methods**

GCA offers the following methods to find the compression point of an amplifier using GCA:

**Compression from Linear Gain**

The Reference Gain is measured using the specified Linear (Input) Power Level. The Target Gain is calculated as the Linear Gain minus the specified Compression Level. For example 8.3 dB - 1 dB = 7.3 dB.

**Compression from Max Gain**

Available ONLY in [2D Acquisition modes].

The linear region of an amplifier gain may not be perfectly linear. After all data is acquired at each frequency, the highest gain value is used as the Reference (S21) Gain. The Target Gain is found in the same way as Compression from Linear Gain.

**Backoff and X/Y method**

These two compression methods are very similar.

- Both methods specify a difference in input power (X axis) between the linear region and compression point.
- For the Y-axis difference:
  - **Backoff method** specifies Compression Level which is a difference in Gain.
  - **X/Y method** specifies Delta Y which is a difference in Output Power.

GCA searches for these points differently for [2D sweeps] and [SMART sweep].

The following images show how Backoff and X/Y method is calculated at ONE frequency.
The compression point (yellow circle) is where 10 dB more input power yields 1 dB less gain than at the reference point (blue circle).

**Acquisition Modes**

The GCA offers three modes for data acquisition: Two 2D sweep modes, and SMART sweep.

**Note:** A traditional power sweep at a single frequency is NOT offered in the GCA channel. However, macros are provided to easily measure and view this data along with GCA data. Learn more.
**2D (two-dimensional) Sweeps**

This is the easiest method to understand, and the least efficient for finding the compression point. Both 2D sweep modes work as follows:

1. All GCA measurements begin by measuring S-parameters at the specified Linear Power level. Reverse parameters are measured ONLY if Full 2-port calibration is applied or if a reverse parameter is displayed. Learn more about Cal choices.

2. Gain measurements are then made at ALL of the specified frequency and power values. Although these are conceptually 2-Dimensional sweeps, a single sweep is constructed in firmware. See Data Points Limit.

3. After data has been measured, a search is performed to find the compression point. You can choose to interpolate between the two measured points closest to the target gain. Learn more.

As each sweep is performed, dots are plotted next to the \textbf{Ch} indicator in the lower left corner of the display to indicate progress for the current sweep.

**Note:** For Backoff and X/Y compression method, GCA does not verify that the specified Start - Stop power range is at least the size of the specified Backoff or X value. The closest compression point is always reported.

### 2D Sweep Modes

- **2D Sweep Power per Frequency** - Input power is stepped from Start to Stop at each specified frequency. From the following example you can see that the device is exposed to the highest power level (p3) at the first frequency (f1). This could heat the device early in the measurement and affect compression results.

The following examples show (frequency, power) values for three frequency points and three power points, resulting in a total of 9 measurements:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1,p1</td>
<td>f1,p2</td>
<td>f1,p3</td>
<td>f2,p1</td>
<td>f2,p2</td>
<td>f2,p3</td>
<td>f3,p1</td>
<td>f3,p2</td>
<td>f3,p3</td>
<td></td>
</tr>
</tbody>
</table>

- **2D Sweep Frequency per Power** - Frequency is swept from start to stop at each specified power level as follows:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1,p1</td>
<td>f2,p1</td>
<td>f3,p1</td>
<td>f1,p2</td>
<td>f2,p2</td>
<td>f3,p2</td>
<td>f1,p3</td>
<td>f2,p3</td>
<td>f3,p3</td>
<td></td>
</tr>
</tbody>
</table>

### Viewing and Saving 2D Data

It is NOT possible to plot ALL of the 2D measurement data on the PNA display. However, it can be saved to a .csv file and then read into an Excel spreadsheet. The initial S-parameter measurement data is not saved to this file.

To save 2D data:

- With a 2D measurement active, click \textbf{File}, then \textbf{Save As}, then select \textbf{File Type} .csv file.
The last complete 2D sweep data is saved. See Save Data Types.

You can also view on the PNA all power sweep information at a selected frequency using a macro. Learn more.

**SMART Sweep**

SMART Sweep is usually the fastest and most accurate method to measure Gain Compression. Unlike the 2D acquisition modes which measure all of the specified frequency / power points, SMART Sweep performs a series of power search iterations. At each frequency, an 'intelligent guess' of input power is made to find the compression level that is within tolerance. This guess is further refined with each successive power search iteration sweep.

SMART Sweep continues to iterate until one of the following conditions occur:

1. **ALL data points are within tolerance.** When the compression level for a data point achieves the specified tolerance, it continues to be measured and input power changed to improve the measurement within tolerance.

2. **The specified compression level can NOT be achieved for the remaining frequencies that are not in tolerance.** Either the Start power is too high or the Stop power is too low.

3. **Maximum iterations have been achieved.** If a measured gain is not within the specified tolerance before the specified Max number of Iterations has been reached, then the last power reading is used as the compression point.

**The Iteration Counter, Dots, and Bangs**

Next to the Ch indicator, in the lower left corner of a GCA window, the following annotation appears:

- An iteration counter is incremented each time input power is adjusted.

- A dot appears when another 10% of the frequency points are within tolerance.

- ! (bangs) are displayed after the last iteration. Each bang represents 10% of the data points that are NOT within tolerance.

**SMART Sweep and Compression Method**

The intelligent guess process works differently depending on the compression method. This is important because Backoff and X/Y compression methods subject the DUT to significant changes in input power during an iteration sweep. This can affect the DUT and the measurement results.

Learn all about Backoff and X/Y compression methods.

ALL GCA measurements begin by measuring S-parameters at the specified Linear Power level. Reverse parameters are measured ONLY if Full 2-port calibration is applied or if a reverse parameter is displayed. Learn more about Cal choices.

- **Backoff and XY** Because both compression methods specify the separation between the 'linear" region and the "compressed" region, each iteration requires two sweeps at different power levels over the same frequency range. The first sweep measures the DUT at the Backoff or X power level. The second sweep
measures the DUT at the compressed power level, specified by the Start and Stop power range. At the beginning of the second sweep, the power level rises by the Backoff or X value. The specified Settling Time is applied at this point to allow the DUT time to react to this significant change in power level. Also Safe Sweep can be used to minimize this change in input power.

- **Compression From Linear Gain** After the reference gain is measured at the linear input power, the next iteration measures the DUT at a power level half way between the linear power level and the stop power. The next sweep, depending upon the compression level of the DUT, either increments or decrements the power by ¼ of the difference between stop power and start power. The third iteration sweep then uses a curve-fit algorithm to precisely find the compression point.

  Note: The DUT can be subject to significant changes in power from one iteration sweep to the next. This can be minimized by the use of SAFE Sweep and careful selection of the corresponding settings.

- **Compression from Max Gain** NOT offered with SMART Sweep.

### Using the Gain Compression Application

The following is a general procedure for performing a GCA measurement. The challenge with GCA is configuring a measurement that yields the true compression performance of YOUR DUT. This requires knowledge of the Gain Compression settings and knowledge of the DUT.

See specific dialog boxes below.

1. Disconnect the DUT if preset or default power levels may damage the PNA or DUT.
2. **Preset** the PNA, or configure a suitable User Preset that will be safe in case the DUT is connected.
3. Create a GCA channel. **Learn how**. The default trace is S21.
4. Start GCA Setup dialog and configure the measurement settings based on the DUT, adapters, attenuators, booster amplifiers, and fixtures to be used in the measurement.
5. Save the instrument state (optional).
6. Connect DUT and apply bias and RF power as appropriate. The default measurement for a GCA channel is S21 (amplifier gain). Inspect the gain measurement to ensure the DUT is operating as expected.
7. Add GCA compression parameter traces. **Learn how**.
8. Adjust the measurement settings to yield satisfactory compression parameters. See GCA Measurement Tips.
9. Start and complete the GCA Calibration wizard.
**How to start the Gain Compression Setup dialog**

To provide quicker access, use the Setup softkey. [Learn how](#).

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
</tbody>
</table>

1. Press **Freq**
2. then **[Gain Compression Setup]**

1. Click **Stimulus**
2. then **Frequency**
3. then **Gain Compression Setup**

---

### Frequency tab - Gain Compression - dialog box help

Configures the frequency settings over which Gain compression is to be measured, as well as the measurement method.

#### Sweep Type

Choose a method in which to sweep frequency: Linear, Log, and Segment Sweeps. This setting applies to all data acquisition modes.

#### Segment Sweep

**Note:** The segment table shown on the dialog is 'READ-ONLY'.

Learn how to [Create and edit the Segment Sweep table](#).

**Independent IFBW** and **Power** are NOT available.

**CW sweep** is NOT available. A traditional gain compression measurement using power sweep at a single CW frequency can be performed in a standard S-parameter channel. [See the Single frequency macro](#).
**Data Acquisition Mode**

Specifies HOW the gain compression data is collected.

**SMART Sweep**

- At each frequency, input power is 'intelligently' adjusted to find a measured gain equal to the target gain.
- Faster and more accurate than 2D sweeps to measure Gain Compression point at a number of frequencies.
- [Learn ALL about SMART Sweep](#)

**2D (two-dimensional) Sweeps**

- **Sweep Power per Frequency**  Performs a series of power sweeps at each successive frequency.
- **Sweep Frequency per Power**  Performs a series of frequency sweeps at each successive power level.
- [Learn ALL about 2D sweeps](#)

**Sweep Settings**

Click each to learn more about these settings.

- **Number of points**  Number of frequency points to measure. The Frequency points may be limited due to the number of specified Power points. [See Data Points Limit](#).
- **IF Bandwidth**  Set this value to yield acceptable trace noise when measuring gain at the linear power level. This level of noise contributes directly to the accuracy of compression point. A lower value (narrower IFBW) allows for more accurate, but slower, measurements. [See GCA Measurement Tips](#) to see how to best set IFBW.
- **Start / Stop, Center / Span**  frequencies. Set the frequency range over which to measure Gain compression.

**Data Points Limit**

The maximum number of measurement data points depends on Acquisition method and Compression method as follows:
### Power tab - Gain Compression dialog box help

Configures RF power and Power Sweep settings for Gain Compression measurement.

**Power ON (All channels)** Check to turn RF Power ON or clear to turn power OFF for all channels.

**Input Port**

Select the PNA port that is connected to the DUT Input.

**Linear Power Level** The input power that yields the linear gain of the DUT. The linear gain is used as the reference gain when calculating the Compression from Linear Gain. Input match is also measured at this power level.
**Source Attenuator** Specifies the attenuator setting associated with the port connected to the input of the DUT. This attenuator will affect the range of available power into the DUT. Learn more about **Source Attenuation**.

All PNA channels in continuous sweep must have the same attenuation value. Learn more.

**Receiver Attenuator** Specifies the attenuator setting for the receiver associated with the input of the DUT. When the power into the receiver test port is around +10 dBm, the PNA receiver may be in compression. However, with receiver attenuation, lower input power levels may become too noisy to make accurate power measurements. In this case, lower IFBW to reduce noise. Learn more about **Receiver Attenuation**.

**Source Leveling** Specifies the leveling mode. Choose Internal. Open Loop should only be used when doing **Wide Band Pulse measurements**.

**Output Port**

Select the PNA port that is connected to the DUT Output.

**Reverse Output Power** Sets power level into the output of the DUT for reverse sweeps. Port power is automatically uncoupled.

Reverse power is applied to the DUT ONLY under the following conditions. Otherwise, this setting is ignored.

- When Linear Output Match or Linear Reverse Isolation parameters are requested.
- When Full 2-port correction is used. You can perform a full 2-port cal and downgrade to an Enhanced Response Cal to prevent reverse power from being applied to the DUT. Learn more.

**Source Attenuator** Specifies the attenuator setting for the port connected to the DUT output. This setting will affect the range of available power at the DUT output port.

**Receiver Attenuator** Specifies the attenuator setting for the receiver associated with the DUT output port.

**Source Leveling** Specifies the leveling mode. Choose from: Internal (normal operation) or Open (use ONLY for **WB Pulse measurements**).

**Power Sweep**

**Power Points** Number of power points to measure for 2D acquisition modes. The Power Points may be limited due to the number of frequency data points. See Data Points Limit. This setting is NOT available in SMART Sweep, which uses only enough power points to find the specified compression level.

**Start** and **Stop Power**

- **2D sweep** In Backoff, X/Y, and Compression from Max Gain methods, sets the range of power levels that are applied to the DUT to find BOTH the **Reference Gain** and **Compression point**. Make sure this range is wide enough to include both. For example, if the Backoff level is 10 dB, then the power range must be greater than 10dB. Otherwise, GCA will report a compression value using the closest reference gain and compression point, which may be inaccurate. In Compression from Linear Gain, the reference gain is measured at the Linear Power Level, so the Start and Stop power levels are used to find the compression point.

- **SMART sweep** Sets the range of power over which GCA will search for the compression point. The reference gain is found using the Linear Power Level, Backoff, and X values, depending on the Compression Method. To reduce the number of iterations that are required to find the compression point, limit the Start / Stop power range to the input levels that will achieve compression. Do not include the linear region.
**Note:** If your DUT requires more input power to achieve compression below 3.2 GHz, use the PNA-X Hi-power mode, available from the RF Path Configuration dialog. The disadvantage to this is higher harmonic content.

**Power Step (Size)** Calculated value from current Start, Stop, and Points settings. This setting can NOT be changed directly.

**Path Configuration** click to launch the RF Path Configuration dialog.

---

**Compression tab - Gain Compression** dialog box help

**Compression Method**

Learn ALL about these Compression Methods

- **Compression from Linear Gain** The specified compression level is measured from the linear gain. The linear gain is measured using the **Linear Power Level** that is specified on the Power tab.

- **Compression from Max Gain** The specified compression level is measured from the maximum gain level. Not available in SMART sweep.

- **Compression from Back Off** This compression method uses the Compression Level and Back Off values for finding the compression point.

- **X/Y Compression** This compression method uses the specified parameters (X and Y) as the criterion for finding the compression point.

**2D Sweep - Compression Point Interpolation**

Check the box to calculate and display interpolated compression traces.
The **Target gain** is calculated using a complex linear ratio between the two closest measured values. All compression parameters are then interpolated using this same ratio.

Clear the box to display compression parameters for the closest compression point, either high or low, to the level specified in the Compression Method setting.

**SMART Sweep**

*Learn ALL about Smart Sweep.*

**Tolerance**  Specifies an acceptable range for measuring the compression level. Reducing this value can significantly increase the number of iterations that are required to find the compression point.

**Maximum Iterations**  Specifies the maximum number of power search iterations SMART Sweep is allowed. Reducing this value can cause SMART sweep to terminate before all compression levels are found to within the specified tolerance.

**Show Iterations**  When checked, the compression parameter traces are updated at the completion of each power search iteration. When cleared, compression parameter traces are updated when SMART Sweep completes the power search iteration process.

**End of Sweep**  Specifies the power level applied to the DUT at the completion of a GCA measurement.

GCA performs numerous power and frequency sweeps on the DUT during the overall measurement process. This setting has no affect on these intermediate sweeps. This setting only applies at the end of the very last sweep in the GCA channel.

In addition, this setting applies ONLY to the GCA channel. All other channels operate independently of this setting. Therefore, the power applied to the DUT after all channels have been measured may be different from this setting.

Choose from:

- **Default**  Use the default PNA method.  *Learn more.*
- **RF OFF**  RF power is turned off when GCA completes a measurement cycle.
- **Start Power**  RF power is set to the start power level.
- **Stop Power**  RF power stays at the stop power level.

**Settling Time**

Used ONLY in SMART Sweep when Back Off or X/Y compression algorithms are selected.

This setting allows additional dwell time when the input power changes from the back-off level to the compression level.  *Learn more.*
Safe Sweep Mode dialog box help

For use with SMART Sweep ONLY.

When enabled, Safe Sweep increases the input power to the DUT by the specified amounts, allowing the compression point to be achieved gradually. While this will increase the number of iterations required to achieve compression, it also minimizes the possibility of driving the DUT too far into compression.

**Safe Mode (Enable)** Check to enable Safe Sweep.

**Coarse Increment** Sets the maximum change in input power, up or down, which will be applied to the DUT from one iteration to the next. Default = 3.0 dB.

Without Safe Sweep, the maximum change in input power can be the entire Backoff or X value when using these compression methods.

**Fine Increment** Once the Fine Threshold has been achieved, this becomes the maximum change in input power, up or down, which will be applied to the DUT. Default = 1.00 dB

**Fine Threshold** Specifies the compression level in which Safe Sweep changes from the COARSE to the FINE increment. Default = .75 dB. This means that, by default, the PNA uses the Fine Power Adjustment when compression reaches 0.75 dB.

GCA Measurement Tips

There are many settings in the Gain Compression Application. Here are a few tips when using GCA to learn as much as possible about the compression characteristics of your DUT in the most efficient manner.

**DUT Compression Characteristics and GCA**

Although GCA provides excellent results with a wide variety of amplifiers, it works best with amplifiers which have a monotonic compression curve. In some cases where the compression curve is not monotonic, for example if the amplifier gain expands before it compresses, the correct compression level may not be found.

To help a SMART sweep find the correct compression point, limit the Start and Stop power levels around the anticipated compression point. Learn more.

The following two power-sweep traces are examples of non-monotonic gain:
DeltaGain

A DeltaGain trace is the best way to see how closely GCA is actually measuring to the desired compression level. In addition, you can view the phase of DeltaGain to see the phase deviation between the compressed gain and the reference gain. DeltaGain is calculated as:

- \[ \text{DeltaGain} = \frac{\text{Measured Gain (watts)}}{\text{Ref Gain (watts)}} \]

- In LogMag format: \[ \text{DeltaGain} = (\text{Measured Gain}) - (\text{Ref Gain}) \]

With SMART Sweep, DeltaGain (in LogMag format) shows how soon certain frequencies achieve the specified tolerance. Learn more.

Some other settings which may be helpful:

- Trigger source: Manual allows you to analyze data and make adjustments while allowing the device to cool.
- Construct Limit Lines around the compression point at the tolerance level.
- Use this macro as a starting point. When edited or run from an external computer (either with remote desktop or a mapped drive) you can make setting changes in the macro and quickly rerun the measurement.

The following image shows a DeltaGain21 trace using SMART Sweep. The Limit Lines were added manually.
In the above image:

| Relevant Settings | Method = Compression From Linear Gain  
|                   | Compression level = 1  
|                   | Iteration Tolerance = 0.05 dB.  
|                   | Maximum Iterations = 10  |

| Displayed Results | A data point on -1.00 indicates that, at that frequency, the exact compression level (1 dB) was measured.  
|                   | Several frequencies did not achieve the specified tolerance (0.05 dB) before the Max Iterations (10) was reached.  
|                   |   - FAIL and red data points outside the limit lines.  
|                   |   - Nine dots (....) indicate that 90% of the data points achieved the specified compression level.  
|                   |   - one ! indicates that 10% of the data points did not achieve compression.  
|                   |   - [Learn more about the Iteration Counter and annotation](#) |

**SMART Sweep Tips**

- Compression from Linear Gain is the easiest compression method to understand and control in SMART Sweep. [Learn more](#).

- If SMART Sweep requires more than twenty iterations, this is an indication that something is wrong. Try changing the Tolerance setting, Frequency Range, Start / Stop power range, IF bandwidth, or [Dwell Time](#).

- If the number of iterations required to achieve the desired compression level changes significantly from one set of measurements to the next, this could be due to other effects, such as heating. Try increasing the dwell time or using a [wide-band pulse](#) measurement configuration.

- If the DUT should not be significantly overdriven into compression, or the changes in the input power should be limited, use [Safe Sweep](#) mode with Deviation from Linear Gain compression method.

**Single Frequency Macro**
Because GCA displays only the compression point for each frequency, and not the entire power sweep, it can be difficult to see some of the more subtle aspects of a measurement. However, it is easy to see a traditional power sweep at a single frequency using one or both macros that are provided with GCA.

With a 2D sweep (NOT SMART Sweep) a script that is stored on the PNA hard drive automatically creates a traditional power sweep measurement in a standard channel using the same stimulus setting as the GCA channel. Use a marker in the GCA channel to specify the frequency for the measurement.

The script has two modes of operation:

- **View Mode** displays all of the previous 2D sweep data at that frequency.
- **Measure Mode** performs a new measurement at that frequency.

Both modes create a new S-Parameter channel using the same stimulus settings as the GCA channel, including port power, attenuator, IF Bandwidth, and dwell settings. The new channel does not support calibration or pulse characteristics.

- To see noise on a measurement, or use the **Measure macro** in continuous sweep. Adjust the IFBW and averaging until the noise versus sweep speed meets your needs.
- To see other effects of your DUT at a specific frequency, use the **View macro** and the Measure macro with 2D sweep mode. Both macros present data using a standard channel. The View macro shows 2D data at a specific frequency, while the Measure macro shows freshly measured data at the same frequency. Ideally, the data from these two would be identical. However, changes in your DUT behavior due to heating or other effects can cause these to be different. If significant differences exist, try:
  - Using the 2D Frequency per Power setting rather than Power per Frequency
  - Adjusting the dwell time
  - Adjusting IFBW
  - Use a wide-band pulse configuration

**How to setup the Macros**

Each macro must be setup separately.

1. Press **Macro**, then **Macro Setup**.
2. Select a blank line, then click **Edit**.
3. In **Macro Title**, type a short description such as Meas GCA or View GCA.
4. Click **Browse**, then navigate to C:\Program Files\Agilent\Network Analyzer\Applications\GCA\**GCA.vbs**
5. In **Macro run string parameters**:
   1. Type **M** for the Measure macro or **V** for View macro.
   2. **Optional**: Supply the following additional parameters in any order:
To run the program from a remote computer, specify the full computer name of the PNA.

Channel in which to create the measurement. If not specified, Measure is created in Ch30 and View is created in Ch31.

**Example:** Run string parameters for the Measure macro run from a remote computer in Channel 5.---- M MyPNA 5.

6. Click **OK**.

**How to run the Macros**

**On a GCA channel:**

1. **Create a 2D sweep.** Either Power per Freq or Freq per Power. Both macros always create a power sweep at the frequency of interest.

2. Create a **ComplIn** trace.

3. On the ComplIn trace, right-click and select **Add Marker**. Drag the marker to the frequency of interest.

4. Press **Macro**, then select either by the short description your provided in Step 3.

Last Modified:

23-Aug-2007   New topic
Noise Figure Application (Opt 029)

The Noise Figure Application makes fast, easy, and accurate noise figure measurements using the PNA-X.

- **Features, Requirements, and Limitations**
- **Noise Concepts**
- **How the Noise Figure Application Works**
- **Noise Parameters**
- **Using Noise Figure App**
- **Noise Figure Measurement Tips**

See Also

- Noise Figure Calibration
- Agilent Noise Figure App Note 57-1

### Noise Figure Application Features

- Cold Noise Source method includes match correction for highly accurate noise measurements.
- Operates from 10 MHz to 26.5 GHz.
- Measures noise figure values ranging from 0 to 30 dB.
- Measures amplifiers with gain ranging from -20 to +40 dB.
- ENR values are interpolated for frequencies between the supplied data points.

### Requirements

- PNA-X with option 029
- Agilent 346C Noise Source: Covers the same frequency range as the PNA-X.
- An adapter may be necessary to connect the Noise Source to the PNA port 2 reference plane during calibration.
- Noise Tuner (N4691B ECal module -m-f recommended) Opt 029 provides an additional cable and adapter to connect the ECal module to the front-panel connectors. Learn more.
- Cal Kit (or second ECal module) with same connector type and gender as DUT connectors.
- Recommended: An accurate thermometer. Learn more.

**Limitations with Noise Figure**

All PNA functions are supported except the following:

- Does NOT work with FCA (opt 083) or Frequency Offset (opt 080).
- All frequency sweeps are STEPPED. Analog sweep is NOT available.
- No External Test Set Control (Opt 550 or 551)
- No Receiver calibration.
- No Enhanced Response Cal
- No ECal User Characterization.
- No Fixture Deembedding.
- No Pulsed Measurements
- No Copy Channels
- No saving Formatted Citifile data.

**Noise Concepts**

The following conceptual information is a short summary taken from the Agilent Noise Figure App Note 57-1.

All electronic circuits have some degree of random noise. The most common form is thermal noise, which increases as the temperature of the circuit increases.

The signal-to-noise (S/N) ratio of components in a communications system is a very important parameter. To improve the S/N ratio, it is usually easier and more cost-effective to reduce noise than to increase signal power. In order to reduce noise, an accurate method to measure noise is required.

**Noise Figure**

Noise figure is the degradation in the signal-to-noise ratio as a signal passes through a device. For example, in the following images:
At the INPUT of an amplifier:
The noise floor is -100 dBm, the signal is at -60 dBm, **40 dB** above the noise floor.

At the OUTPUT of the same amplifier:
The gain has boosted the signal AND the noise floor by 20 dB.
The amplifier then added 10 dB of its own noise.
The output signal is now only **30 dB** above the noise floor.
Since the degradation in signal-to-noise ratio is 10 dB, the amplifier has a **10 dB noise figure**.

For consistency, noise measurements are calculated as if using a 1 Hz bandwidth, although measurements are almost always made at higher bandwidths.

The following formula shows the lowest possible noise power in dBm at 290° K (room temperature). The only way to measure noise lower than this is to make the measurement at a lower temperature.

- \[ P = 10\log(4.0 \times 10^{-21} \text{ watts}/.001 \text{ watt}) \]
- \[ P = -174 \text{ dBm} / \text{Hz} \]

**How the Noise Figure Application Works**

The noise figure application includes two noise receivers which measure the noise coming out of the DUT. The noise receivers are calibrated using a characterized noise source. [Learn more about the noise calibration process.](#)

A major source of noise measurement error is caused by a poor impedance match at the DUT input. Therefore, during every measurement, the Noise Figure Application uses an ECal module to present at least four different impedances at the input of the DUT. This "Noise Tuner" is connected to the PNA port 1 front-panel loops which is in the PNA internal source path. From the measurements at various impedance states, the PNA calculates the noise out of the DUT as though the PNA were exactly 50 ohms. No assumptions are made regarding the input impedance of the DUT.

Here is how a noise figure measurement is made:
• The DUT input is always connected to the PNA source at port 1.

• The DUT output is always connected to the PNA receivers at port 2.

The sweep numbers are annotated on the PNA display as they occur.

1. With the noise tuner in the THRU state, S-parameter measurements are made to accurately characterize the gain of the DUT. This requires sweeps in both forward and reverse directions. (sweep #1 and #2).

2. The noise measurements are performed next. PNA source power is turned OFF and the noise tuner is switched to the first impedance state.

3. At each frequency, the noise receiver samples a large number of readings in order to attain one valid measurement. If Noise Averaging is selected, the specified number of measurements are made and averaged together to obtain one noise measurement. This continues for all frequencies (sweep #3).

4. The next noise tuner impedance state is switched IN and the noise measurements in step 3 are repeated. This occurs until measurements are made at all impedance states. At least four impedance states must be used. (sweeps #4, #5, #6+)

5. Calibration error terms are applied and calculations made to simulate the measurement with a perfect 50 ohm input impedance. The sweep result is plotted on the PNA display.

6. The PNA begins sweeping again with step 1.

**PNA-X Block Diagram with Noise Figure components**

---
Noise Figure Components are shaded yellow

- At test port 1 front-panel loops, a **DPDT switch** connects the noise tuner (ECal module) in series with Source1 providing several different input impedances. [Learn more](#).

- At test port 2, **DPDT switch** and **coupler** to route RF from the DUT output to **two noise receivers**. The appropriate receiver is automatically switched as required for frequency being measured.

### Making S-parameter measurements and the Noise Tuner Switch

The default setting for the port 1 DPDT switch is EXTERNAL, as shown in the above diagram. This setting always provides incident power through the front panel loop. When an ECal module is connected, it may NOT be in the THRU state, which is necessary for accurate S-parameter measurements. This can be changed in any of the following ways:

- Set the switch to INTERNAL for the S-parameter channel using the path configuration dialog.

- Set the switch to INTERNAL for the S-parameter channel using the following commands
  
  - **SCPI** - `SENS:PATH:CONF:ELEM:STAT "Port1NoiseTuner", "Internal"
  
  - **COM** - `PathConfiguration.Element("Port1NoiseTuner").Value = "Internal"

- Change the preferred default setting to INTERNAL using **SCPI** or **COM**.
Set the Noise Tuner (ECal module) to the THRU state using \texttt{CONT:ECAL:MOD:PATH:STATE}.

**Noise Figure App vs Noise Figure Analyzer**

In comparing the PNA Noise Figure App measurements with the NFA Series Noise Figure Analyzer measurements, you may obtain different results. This is because the Noise Figure Analyzer assumes that the DUT has a perfect 50 ohm input. The PNA Noise Figure App measures the source match and calculates the noise figure as though it were a perfect 50 ohm match. In addition, the PNA measures the amplifier gain with vector error correction applied to reduce measurement uncertainty.

**Noise Parameters**

Several noise parameters, as well as standard parameters, can be measured in a GCA channel.

<table>
<thead>
<tr>
<th>How to add or change Noise Figure Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Using front-panel HARDKEY [softkey] buttons</strong></td>
</tr>
<tr>
<td>For PNA-X</td>
</tr>
<tr>
<td>1. Press \texttt{MEAS}</td>
</tr>
<tr>
<td>2. then select a parameter</td>
</tr>
<tr>
<td>3. then select a parameter</td>
</tr>
</tbody>
</table>

**Noise Parameters that are offered**

- **Noise Figure** - Explained above in [Noise concepts](#).
- **T-Effective** - The effective temperature, in Kelvin, of the measured noise level. For example 290° K = -174 dBm/Hz.
- **DUT Noise Power Density** - The total noise generated by the DUT, without system noise.
- **DUT Relative Noise Power** - DUT Noise Power Density MINUS 290° K, expressed in K and normalized to room temperature.
- **System Noise Power Density** - The total noise measured at the noise receivers. This includes noise generated by the DUT plus the noise generated by the PNA noise receivers and other system components.
- **System Relative Noise Power** - System Noise Power Density MINUS 290° K, expressed in K and normalized to room temperature.

**Standard Parameters that are offered**

- **S-parameters**: S11, S21, S22, S12
• **Unratioed parameters** using the following notation: (Receiver, source port). These parameters REPLACE the active GCA measurement. To do this (from front-panel ONLY), press **Meas**, then [More], then [Receivers].

  - (R1,1), (R2,2), (A,1), (A,2), (B,1), (B,2)

**Using the Noise Figure Application**

Use the following general procedure to make measurement with the Noise Figure App:

1. **Connect Tuner and Noise Source.**
2. **Create a Noise Figure Measurement.**
3. **Make Noise Figure Settings.**
4. **Perform Calibration** First copy your Noise source ENR file to the PNA.
5. Connect the DUT:

   - DUT Input to PNA port 1.
   - DUT Output to PNA port 2. For highest Noise Figure accuracy, there should be the least amount of electrical loss possible between the DUT output and the PNA Port 2.
6. Measure Noise Figure.
7. **Optional** Click File, then Save to save Noise Figure data in the following formats: (available ONLY when NF correction is ON.)
   - *.CTI Citifile
   - *.PRN
   - *.nco Noise Correlation Matrix data in S2P format. [Learn more about this data.](#)

See Also: Measurement Tips

**Connect Noise Tuner and Noise Source**

1. Connect the noise source to the **28V connector** on the PNA-X rear panel. The Noise Source is turned ON and OFF automatically as needed during a calibration. Connect the noise tuner to Port 2 reference place when prompted during calibration.

2. Connect the noise tuner (ECal module) On the PNA front panel, remove the **Port 1** jumper cable SOURCE OUT / CPLR THRU. Connect tuner using the cable (N5242-20137) and adapter (85082-60013) supplied with Opt 029.
Create a Noise Figure Measurement

1. On the PNA-X front panel, press **Meas**, then **[Measurement Class]**

2. Select **Noise Figure Cold Source**, then either:
   - **OK** delete the existing measurement, or
   - **New Channel** to create the measurement in a new channel.

3. A Noise Figure measurement is displayed. To select additional parameters to display, click **Response**, then **Measure**, then select a parameter from the list.

How to start the Noise Figure Setup dialog

To provide quicker access, use the Setup softkey. **Learn how**.

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>FREQ</strong></td>
<td>1. Click <strong>Response</strong></td>
</tr>
<tr>
<td>2. then <strong>[Noise Figure Setup]</strong></td>
<td>2. then <strong>Measure</strong></td>
</tr>
<tr>
<td></td>
<td>3. then <strong>Noise Setup</strong></td>
</tr>
</tbody>
</table>
**Noise Figure Setup** dialog box help

**Note:** In this topic, the term **Jitter** is used to describe the trace-to-trace fluctuations in a measurement. In other topics, this is called 'trace noise'.

**Bandwidth/Average**

The following two settings work together to achieve the optimum balance of measurement accuracy versus speed. Both settings can be changed after calibration to make faster measurements with minimal effect on calibration accuracy.

- **Noise Bandwidth** Increase the bandwidth to make faster measurements. However, a wider setting reduces the frequency resolution of the measurement. More frequencies are essentially smoothed together to produce a flatter response, which could hide the actual noise performance of the DUT.

- **Noise Averaging Factor** Increase the number of averages to reduce jitter. This also increases measurement speed. For maximum accuracy, increase the averaging factor for the noise calibration. It can then be reduced to improve measurement speed.

**Noise Receiver Gain**

With knowledge of your DUT gain, set the appropriate amount of receiver gain in order to optimize the power level at the noise receiver.

The following values reflect the SUM of the DUT gain (dB) **PLUS** NF (dB). For example: DUT gain = 20 dB; NF = 10 dB; SUM = 30 dB.

- Select **High** if the SUM is relatively low (<30 dB).
- Select **Medium** if the SUM is about average (20 dB to 45 dB).
- Select **Low** if the SUM is relatively high (>35 dB).

There is considerable overlap in these settings. Because all three gain settings are calibrated with each Noise Calibration, this setting can be changed after calibration to achieve the least amount of jitter without overpowering the noise receiver.

When too much power is detected at the noise receiver, a warning message appears, and the next lower gain setting is automatically selected.

Only ONE gain setting can be used for the entire frequency range of your noise measurement. Therefore, it may be necessary to use two noise channels with different frequency ranges and gain settings to achieve the
very highest noise figure accuracy.

**Ambient Temperature**

Enter the room temperature at the time of the measurement, in Kelvin. For best results, use a thermometer to read the temperature at the PNA test port 1 or the DUT input cable.

This ambient temperature number has an inverse relationship to the noise figure. When using the effective noise temperature (Te) format, a 3 degree increase in the ambient temperature will make the overall measurement result drop 3 degrees.

**Impedance States**

- **Noise Tuner** Displays the ECal module to be used as a noise tuner. Select the Noise Tuner during calibration on the Select Cal Method dialog.

- **Max Acquired Impedance States** Select the number of impedance states in which to make noise measurements. At least FOUR impedance states are required. Learn more

---

**Frequency Tab - Noise Figure** dialog box help

These settings can also be made from the normal PNA setting locations. Click links below to learn how.

**Sweep Type**

Choose a sweep type. Learn more.

**Sweep Settings**

Click each to learn more about these settings.

- **Number of points**

- **IF Bandwidth** For standard PNA receiver measurements. This setting is important for improving noise measurement accuracy. Learn more.

- **Start / Stop, Center / Span** frequencies.
**Power Tab - Noise Figure** dialog box help

**Note:** S-parameter power settings are critical for accurate Noise Figure measurements. [See Noise Figure Measurement Tips.](#)

Configures RF power settings for the S-parameter measurements that occur before noise measurements. Input power to the DUT is turned OFF during noise measurements.

These settings can also be made from the normal [Power setting](#) locations.

**Power ON (All channels)** Check to turn RF Power ON for all channels.

**DUT Input Port**

PNA Port 1 is connected to the DUT Input. This can NOT be changed.

- **Note:** Input power levels are critical for accurate Noise Figure measurements. [Learn more.](#)

- **Power Level** The input power to the DUT during S-parameter measurements.

- **Source Attenuator Auto** Check to automatically select the correct attenuation to achieve the specified input power. Clear, then select attenuator setting that is used achieve the specified Power Level. [Learn more about Source Attenuation.](#)

  All PNA channels in continuous sweep must have the same attenuation value. [Learn more.](#)

- **Receiver Attenuator** Specifies the receiver attenuator setting for port 1.

- **Source Leveling** Specifies the leveling mode. Choose Internal. Open Loop should only be used when doing [Wide Band Pulse measurements](#) (not available with Noise figure measurements).

**DUT Output Port**

PNA Port 2 is connected to the DUT Output. This can NOT be changed.

- **Output Power** Sets power level in to port 2 for reverse sweeps. Port power is automatically uncoupled.

  Reverse sweeps are always applied to the DUT when Full 2-port correction is applied. [Enhanced Response Cal](#) is NOT available for Noise Figure measurements.
**Source Attenuator** Specifies the source attenuator setting for reverse power.

**Receiver Attenuator** Specifies the receiver attenuator setting for port 2. [Learn more about Receiver Attenuation.](#)

**Source Leveling** Specifies the leveling mode. Choose Internal.

---

**Noise Path Configurator** dialog box help

**Port 1 Noise Tuner Switch** The orange line between CPLR THRU and SRC OUT represents the Noise Tuner. The **External** setting switches IN the Noise Tuner when making noise measurements.

**Port 2 Noise Receiver Switch** allows you to make Noise Receiver measurements.

To prevent premature wear on the above two Noise switches, the PNA does not allow these switches to be thrown when sweeping a Noise channel and non-Noise channel. To make Noise Figure measurements and non-Noise Figure measurements in different channels and continuously trigger both, set these switches to the same state as the Noise channel:

- With the **non-Noise Figure channel** active, go to **Noise Path Configurator**.
- Set Noise Tuner switch to **External**. This routes source power to the front-panel loops, and to the Noise Tuner when connected. Use `CONT:ECAL:MOD:PATH:STATE` to set the internal state of the Noise Tuner to THRU, which creates a small amount of additional loss in the source path.
- Set Noise Receiver Switch to **Noise Receiver**.

---

**Noise Figure Measurement Tips**

**Note:** In this topic, the term **Jitter** is used to describe the trace-to-trace fluctuations in a measurement. In other topics, this is called 'trace noise'.

Noise Figure measurements are extremely sensitive and vulnerable to small changes in temperature and the
surrounding environment. Cell phone usage and other wireless devices can affect measurement results.

For highest Noise Figure measurement accuracy and stability, there should be the least amount of electrical loss possible between the DUT output and PNA Port 2.

**S-Parameters**

S-parameters are used to measure the gain of the DUT before each series of noise measurements. Jitter in the S-parameter measurements corresponds directly to jitter in the noise measurements.

In general, for best measurement accuracy, the power level at the B receiver (port 2) should be close to $+10 \text{ dBm}$. Much below this level, measurements have more jitter. Above this level, the B receiver starts to compress, although there is no warning or annotation that shows this condition is occurring.

The best way to monitor power at the B receiver is to display a B,1 measurement. With your DUT in place and powered ON, change the input power to the device and note the power at the B receiver.

- For low-gain amplifiers, use 5 dB of source attenuation to improve the uncorrected match of port 1.
- For high-gain amplifiers, source and receiver attenuation may be required. Use the lowest possible attenuation values.

**S-parameter Calibration**

During a noise calibration, it is also important that the power level at the B receiver (port 2) be close to $+10 \text{ dBm}$. However, this can be challenging since calibration is performed without the DUT in place. Because of this, it is often necessary to set source power higher during the calibration than during the measurement. This will cause the ‘C _ annotation on the status bar. However, measurement results are accurate as long as the step attenuators and other configuration switches are in the same position and all receivers remain in their linear range (below $+10 \text{ dBm}$).

It is best to find the optimum power and attenuation settings for both the calibration and subsequent noise measurements before performing a calibration.

**IF Bandwidth**

Jitter is further reduced by narrowing the IF bandwidth. If the calibration needs to be performed at a low source power, or with receiver attenuation due to high DUT gain, the IF bandwidth should be reduced during the calibration to reduce jitter. The IF bandwidth can then be increased to improve measurement speed. The C _ annotation can be ignored when changing IFBW after calibration.

**Noise Settings**

See Noise Figure dialog box help for a complete description of these important settings.

**Temperature**

Noise Figure measurements are extremely sensitive to temperature. As such, there are two settings that require an accurate temperature measurement: At the DUT input, and at the Noise Source connector.
Radio-Frequency Electromagnetic Field Immunity

When a 3Vm-1 radio-frequency electromagnetic field is applied to an N5242A with Opt 029 according to IEC 61000-4-3:1995, degradation of performance may be observed. When the frequency of the incident field matches the frequency of a measured noise figure or gain, the values displayed will deviate from those expected. This phenomenon will only affect that specific frequency, and the analyzer will continue to perform to the specification at all other frequency sample points.

The N5242A with Opt 029 may be unable to calibrate a chosen frequency sample point if the frequency matches that of an incident electromagnetic field.
The Narrowband Pulsed Application is a Visual Basic program that provides a user interface for making pulsed measurements.

Learn about the New Wideband Pulsed Application.

- **Required Options**
- **Physical Connections**
- **Using the Narrowband Pulsed Application**
- **How to Configure Pulse Generators and Receivers**
- **Calibration in Pulse Mode**
- **Pulse Profiling**
- **Signal Reduction versus Gate Width**
- **Pulsed Frequency Converter Measurements**
- **Writing your own Narrowband Pulsed Application**

The following enhancements were made in PNA Rev. 7.2:

- **Enhanced Pulse Measurement Capabilities**
- **Support for Internal Pulse Generators / Modulators (PNA-X only)**

For more conceptual information see our Pulsed Measurement App Notes. See PNA-X Block Diagram of IF Path / Pulse Generators / Source Modulation

---

### Required Options and Equipment

The PNA H08 option provides the Narrowband Pulsed Application. The following options are also required. If your PNA does not have the required options, a message is displayed on the screen. For more information, see Pulsed-RF Measurements Configuration Guide

- E836x models: **Opt 014 (front panel access)** and **Opt 080 (frequency offset)**. To use the internal receiver gating feature of the Narrowband Pulsed Application, your PNA must have the **H11 hardware option**.
- PNA-X models: None; however **Opts 021, 022, and 025** greatly enhance speed, performance, and convenience.
- PNA-L models: HO8 NOT available
Agilent 81104A or 81110A Pulse Generator with ONLY the 81105A or 81111A output modules. The 81112A module does NOT have selectable 50 ohm/1K ohm output impedance/load compensation to drive the 1K ohm PNA IF gates. For more information, see the 81100 Family of Pulse Pattern Generators Technical Specifications at: http://cp.literature.agilent.com/litweb/pdf/5980-1215E.pdf

**Physical Connections**

Each 81110A Pulse Generator has two output modules. Each output can drive a PNA IF Receiver or Source Modulation (Z5623A H81).

**Connect the Pulse Generators as follows:**

81110A front panel connectors

- Connect GPIB cables to the 81110As and PNA.
- Connect the PNA 10 MHz Ref Out to the 81110A 10 MHz IN.
- If using two 81110As for a total of 4 outputs, then connect the TRIGGER OUT of one to the EXT INPUT of the other 81110A.
- Connect the 81110A OUTPUTs to the PNA rear panel IF inputs to be gated. The outputs are mapped in the Pulsed Generator Configuration dialog box.

**Connect the Z5623A H81Pulse Test Set (optional) to the PNA front-panel port 1 loops as follows:**

<table>
<thead>
<tr>
<th>PNA</th>
<th>H81</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src Out</td>
<td>Source IN</td>
</tr>
<tr>
<td>CPLR THRU</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>RCVR R1 IN</td>
<td>RCVR R1 Out</td>
</tr>
</tbody>
</table>

**See Also**

- PNA Front-panel loops
- PNA-X rear-panel
- PNA IF connectors
- 81110A Documentation
- Z5623A H81 Documentation
Using the Narrowband Pulsed Application

How to start the Narrowband Pulsed Application

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>PNA-L</strong> and <strong>E836x</strong> models</td>
<td></td>
</tr>
<tr>
<td>1. Press <strong>Macro</strong> until <strong>Pulse</strong> is visible</td>
<td>1. Click <strong>System</strong></td>
</tr>
<tr>
<td>2. then <strong>F1</strong></td>
<td>2. then <strong>Macro</strong></td>
</tr>
<tr>
<td></td>
<td>3. then <strong>Pulse</strong></td>
</tr>
</tbody>
</table>

For **PNA-X**

1. Press **SYSTEM**
2. then **Macro**
3. then **Pulse**

See Also

See programming commands to launch the Macro remotely.
See how to write your own custom Narrowband Pulsed Application.

Keypad Data Entry

The PNA front-panel Numeric Entry and Navigation keys can be used for dialog box input. Also, a keyboard can be used to enter values, including alpha characters for prefixes (for example, u for usec.) After typing values, first press **Enter**, then press **Tab** to go to the next field.

The following is an image of the main dialog box:
**Pulsed Application Main** dialog box help

**Note:** An error message may appear on the PNA stating that the response frequency has exceeded the maximum allowed frequency.

The Narrowband Pulsed Application may set the offset frequency (option 080) of the PNA to some value other than zero (the default value). If the stop frequency is set to the maximum of the PNA model, then the error message will appear.

To fix this, set the stop frequency to a value that is at least 2 KHz less than the maximum allowed. For example, if you have a 20 GHz PNA, and the stop frequency is set to 20 GHz, and the error message appears, then set the stop frequency to 19.999998 GHz.

See Block Diagram of IF Path / Pulse Generators / Source Modulation

**Configure**

You can configure more than one channel to make pulsed measurements, but the channels must use the same pulse generator settings.

Only the Agilent 81110A Pulse Generator is supported with the Narrowband Pulsed Application. Refer to the 81110A documentation for pulse repetition frequency and duty cycle capabilities.

See also:

- Configure Receivers
- Converter Measurements

**Edit / Undo** Pulse Application settings revert to those when Apply was last pressed.
**Desired PRF and IFBW**  Enter the DESIRED values. When **Calculate** is pressed, one or both of these values may change.

- **Pulse Repetition Frequency (PRF):** Frequency of the pulses from the Pulse Generator.
- **Pulse Repetition Interval:** 1/PRF  Changes to either PRF or this setting changes both.
- **Receiver IF Bandwidth:** IF Bandwidth of the PNA. Choose a setting from 1 Hz to 10 KHz.
- **Fixed PRF**  When checked, (default setting) the **Calculate** algorithm will NOT adjust the PRF, but only change the IF Bandwidth.

**Modulation/Gates**  The Source Modulation and four PNA receiver gates can each have their own, or share, Pulse Generator outputs. Shared outputs have identical Width and Delay values. To configure and enable outputs, click **Configure,** then **Pulse Generators** to launch the **Pulsed Generator Configuration** dialog box.

- **Width**  Pulse Width.
- **Delay**  The delay that occurs before the pulse.

**Duty Cycle**  Calculated Duty Cycle of the source and each of the selected receivers. Updated when **Calculate** is pressed.

**Pulse Mode On**  When this box is checked, the PNA is enabled for Pulsed measurements. The PNA **Status Bar** annotation indicates the following:

- **G**  Internal IF gates enabled.
- **F**  Filtering for Pulsed Measurements enabled.

**Apply**  All selections are sent to the pulse generator and the active channel of the PNA.

**Calculate**  All selections are calculated and valid PRF and IFBW values are entered in their fields. If these settings are not acceptable, try changing the values you previously entered and click **Calculate** again. When acceptable values are attained, click **Apply** to send these values to the pulse generator and PNA.

**Pulse Profile**  Launches the Pulse Profile dialog box. Same as clicking **View / Pulse Profile.** If not available, check **Pulse Mode ON,** click **Calculate,** then **Apply.**

**Minimize**  Click to minimize the dialog box to make changes in the PNA application. To see the dialog again, select **Macro, Pulse,** or turn the **Status Bar** ON.

**Save**  All settings from the Narrowband Pulsed Application are saved in a *.ppf file. These settings are NOT saved with PNA instrument state.

**Recall**  Restore settings from the specified *.ppf file that were previously saved.

**Close**  Closes the dialog box without saving changes.
How to configure Pulse Generators / Modulators and Receivers

From the Pulse App main dialog box

Learn about...

- **Configure Receiver Gain**
- **Converter Measurements**
- **No Pulse Generators**  When checked, the Narrowband Pulsed Application does NOT attempt to communicate with internal or external pulse generators. This setting is used for troubleshooting purposes.
- **No SW Gating**  When checked, the improved SW gating sensitivity is turned OFF. This setting is used for troubleshooting purposes.
Pulsed Generator Configuration dialog box help

See Block Diagram of IF Path / Pulse Generators / Source Modulation

This dialog may look different depending on the PNA model and number of receivers available.

Configures either the internal pulse generators (PNA-X models with relevant options), or Agilent 81110A Pulse Generator outputs. You can configure each 81110A Pulse Generator with either one or two 81111A output modules.

The Source Mod and four PNA receiver gates can each have their own, or shared, pulsed generators allowing identical Width and Delay values which are selected on the Main dialog.

To share an external generator output between one or more PNA inputs, use the same GPIB address and output module for each PNA input.

Internal Pulse Gen Output (available ONLY on the PNA-X opt 025)

Specify the Pulse Gen (1 through 4) to use to modulate each of the PNA receiver IF gates or Sources.

External Pulse Generator settings

- GPIB Addr: The GPIB address of the 81110A.
- Output: The output module of the 81110A.
- Master: The 81110A that uses the 10 MHz reference signal from the PNA.
- Enabled: Turns the pulse output ON.

External Gate/Modulator settings

- High: Specify a 'TTL-High' voltage level
- Low: Specify a 'TTL-Low' voltage level
- Ext Impedance: Impedance of the modulator used to create the pulse.
- Complement: When this box is cleared, TTL HIGH is the pulse. When checked, TTL LOW is the pulse.

Using Internal Modulators When this box is checked, the voltage, impedance, and complement values are forced to settings that prevent damage to the internal modulator.

Using Internal Pulse Generators Makes the appropriate settings on this dialog available.

Using Internal PNA gates When this box is checked, the voltage, impedance, and complement values are forced to settings that prevent damage to the internal gates.
**Receiver Gain Configuration** dialog box help

See Block Diagram of IF Path / Pulse Generators / Source Modulation

This dialog may look different depending on the PNA model and number of receivers available.

Sets the gain of each PNA receiver manually or automatically.

- **Auto** - The PNA selects the best gain level to make pulsed measurements.
- Use the following to manually set the gain for each receiver.

  - **Low** - about 0 dB of gain
  - **Medium** - about 17 dB of gain
  - **High** - about 24 dB of gain

The PNA-X has the following attenuation settings:

  - **Low** - 30 dB of attenuation
  - **Medium** - 15 dB of attenuation
  - **Hi** - 0 dB of attenuation

---

**Calibration in Pulse Mode**

To perform a calibration in pulse mode (option H08), first configure and apply the pulse parameters (PRF, Pulse Width, Delays, IF gating, and so forth) **before** calibrating the system. This will ensure the PNA is configured properly during the calibration and measurement.

When performing **Unknown Thru** or **TRL calibrations**, ALL receivers must be gated. Otherwise, the error terms will not be correct after the calibration has completed. This can be accomplished by either having a separate pulse generator output for each of the IF gates, or by connecting pairs of the IF gates together with BNC-T's. For example, if the pulse generator does not have enough outputs, then connect the R1 and R2 IF gates to the same pulse generator output. Also, connect the A and B IF gates to either separate outputs (recommended) or one output (reduces flexibility). The error terms will then be valid after the calibration is complete.

**Pulse Profiling**

Pulse profiling provides a time domain view of the pulse envelope. Profiling is performed using a measurement technique that "walks" a narrow receiver "snapshot" across the width of the pulse. This is analogous to using a camera to take many small snapshots of a wide image, then piecing them together to form a single, panoramic view.

- Pulse Profiling can be performed using ratioed or unratioed measurements.
- Pulse Profiling is performed at a single CW frequency.
How to perform Pulse Profiling

From the Pulse App main dialog box,
Click the Pulse Profile button. or:

If this setting is unavailable, check Pulse Mode ON, click Calculate, then Apply.

Pulse Profile dialog box help

Learn about Pulse Profiling (scroll up)
See Block Diagram of IF Path / Pulse Generators / Source Modulation

Modulation / Gates
These setting duplicate those found on the main Pulse App dialog box.
In Pulse Profile, the Gate Delay settings (highlighted in yellow) are significant only with certain Measurement Parameter and Couple Gates settings.

Time Parameters

Start, Stop  These two combine to make the window of the assembled pulse profile. To view the entire pulse, the start and stop values must be at least as wide as the Source Modulation Width plus Delay value.

Step  Each consecutive snapshot is incremented by this value until the stop value is reached. Therefore, the number of points for the pulse profile measurement can be calculated as: (Stop - Start) / Step. The higher the number of points, the longer it takes to make the measurement.
**Measurement Parameter**

**CW Freq.** Frequency of the PNA source.

**Source Port** The PNA port supplying the source power. Only required for single receiver (unratioed) measurements.

**Param(eter)** Only those receiver gates (and relevant measurements) that are configured in Pulsed Generator Configuration are available.

**Note:** When a single receiver (unratioed) is selected, Gate Delay Settings (highlighted in yellow on above dialog image) are ignored.

If the reference receiver gate is NOT configured, the average of the Source Modulation pulse is used as the reference. For example: With S21 Selected, but ONLY B receiver gate is configured, then...

- **Source Mod**

- **B Rec Gate**

- **R1 (Not Gated)**

**B Gate is walked across the Source Modulation pulse.**

**Source Modulation pulse average is used as reference (not gated).**

**Coupled Gates** Used when the appropriate receiver gates are configured for your S-parameter measurement ONLY. This setting is ignored when a single receiver (Param) is selected.

- **Uncoupled** (box cleared) The reference gate is FIXED at the delay setting as the test gate is walked across the Source Modulation pulse as dictated by the Time Parameter settings.

  **For example:**

  - **S21 Selected, B and R1 receiver gates configured, Gates Uncoupled**

    - **Source Mod**

    - **B Rec Gate**

    - **R1 Rec Gate**

    **B Gate is walked across the Source Modulation pulse.**

    **R1 gate is fixed at pulse width and delay setting.**

- **Coupled** (box checked) The reference gate is walked synchronously with the test gate as dictated by the Time Parameter settings. Only the difference between the test and reference gate delay values is significant; NOT the absolute values.
For example:

S21 Selected, B and R1 receiver gates configured, Gates Coupled

B gate delay = 3 microseconds,

R1 gate delay = 2 microseconds

Difference = 1 microsecond

B Gate is walked across the Source Modulation pulse.

R1 gate is fixed at pulse width and delay setting.

B gate leads R1 gate by 1 microsecond.

Data Format  Log Magnitude, Linear Magnitude, or Phase (only available if S-parameter selected).

Buttons

Show Gates  Allows you do change the receiver gating width and delay while looking at the results.

Apply Gate Settings  Click after making changes to gate settings.

Continuous Sweep  Check, then click Measure, to continuously measure pulse profiling.

Measure  Click to start the pulse profile measurement. Becomes Stop when continuously sweeping.

Marker to Delay  After making a measurement, you can drag the display maker to any point along the trace. Click this button and the marker time is entered into the Receiver Delay field on the main dialog box.

Save Data  Saves time domain data to the PNA hard drive in any of the following formats:

- Touchstone (*.s1p)
- Comma delimited (*.prn)
- Citifile (*.cti)

Learn more about these data formats

Signal Reduction versus Gate Width

<table>
<thead>
<tr>
<th>Signal Reduction versus Gate Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRF = 1 MHz</td>
</tr>
</tbody>
</table>

The following two figures show the performance of the internal IF gates as the width is narrowed.
The following is a zoomed image of the shaded area (above).

- The straight line shows the theoretical loss in dynamic range due to duty cycle effects when using narrowband detection.
- The curved (red) line shows the actual measured performance of the gates.
- The minimum gate width for <1dB deviation from theoretical is approximately 20ns.

See the specifications for the option H11 and option H08.
Pulsed Frequency Converter Measurements

The Narrowband Pulsed Application works with both FCA (option 083) and standard Frequency Offset (opt 080) measurements. On the Configure menu, check Converter Measurements. When checked, this setting prevents the Narrowband Pulsed Application from overwriting frequency offset values. This may limit the number of PRF and IFBW solutions that are returned when Calculate is pressed on the main Pulsed Application dialog box.

Note: Pulse Profiling can NOT be performed with frequency converter measurements.

Writing your own Narrowband Pulsed Application

You can use the Narrowband Pulsed Application or use an example program as a template for making your own Narrowband Pulsed Application.

The Narrowband Pulsed Application uses a custom .dll to perform the calculations that are necessary to make pulsed measurements. Use the COM Method below to send and return values to agilentpnapulsed.dll. Then use SCPI or COM commands to control the PNA.

<table>
<thead>
<tr>
<th></th>
<th>E836x</th>
<th>PNA-X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example Program</td>
<td>E836x Create</td>
<td>PNA-X Create</td>
</tr>
<tr>
<td>COM Methods</td>
<td>ConfigNarrowBand3</td>
<td>ConfigEnhancedNB2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ConfigEnhancedNBIFAtten</td>
</tr>
<tr>
<td>SCPI commands</td>
<td>SCPI</td>
<td>SCPI</td>
</tr>
<tr>
<td>COM commands</td>
<td>COM</td>
<td>COM</td>
</tr>
</tbody>
</table>

Install and Register the Pulsed .dll on your PC

To create your own Narrowband Pulsed Application, or run the Narrowband Pulsed Application from a remote PC, you must do the following:

1. Copy the following files from the PNA C:\program files\agilent\network analyzer\ to a directory on your PC.
   - agilentpnapulsed.dll
   - OffsetList.txt
   - prfbw.txt
   - prfbwmixer.txt

2. To register the ActiveX DLL in Microsoft Windows Operating System:
From a command prompt on your PC, navigate to the directory where you copied the DLL.

- Type: `regsvr32 agilentpnapulsed.dll` and press Enter

For Operating Systems other than Windows, see their associated help files to learn how to register DLL files.

Last Modified:

- 20-Feb-2008       Added physical connections
- 6-Nov-2007        Edited Ext Impedance
- 22-Jun-2007       converted to NB pulsed
- 122-Jun-2007      Updated for MX
Wideband Pulsed Application

The Wideband Pulsed Application configures the PNA-X internal pulse generators and modulators for measuring pulsed S-parameters using the wideband mode detection technique.

The Wideband Pulse Application is designed to be used with the PNA-X with Opt 021, 022, and 025.

**Note:** Wideband Pulse application is NOT supported on the E836x and PNA-L models.

See Also

- To learn more about wideband detection, see Application Note 1408-12.
- See a Visual Basic example: Create a Wideband Pulsed Measurement using the PNA-X
- Learn about the Narrowband Pulsed Application.

Download and Install the Wideband Pulsed Application

This application is installed and run as a macro on the PNA-X. Learn more about macros.

2. Click the download link
3. Save the downloaded file to the PNA hard drive
4. Double-click the downloaded file to install the Wideband Pulsed Application on the PNA.
5. Configure the macro. Learn how. The application is installed at C:\Program Files\Agilent\Network Analyzer\Applications\WB Pulse\Wideband_pulse.exe

To learn more about Wideband pulsed application, click Help in the application.

Last Modified:

22-Jun-2007 MX New topic
Frequency Offset Mode

Frequency Offset Mode (FOM) provides the capability to have the PNA Sources tune to frequencies that are different (offset) from the PNA Receivers.

PNA Option 080 provides you with the hardware and basic software capability to make Frequency Offset Measurements. This topic discusses the PNA settings that are relevant to making these types of measurements. See Frequency Converting Device Measurements for more information on making specific device measurements.

**Note:** The Frequency Converter Application Option 083 simplifies the task of making extremely accurate frequency offset measurements on MOST frequency converting devices.

---

**Frequency Offset Dialog Box**

**Setup Examples**

**Test Set (Reference Switch) Dialog Box**

---

**Other Frequency Offset topics**

---

**How to make Frequency Offset settings**

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Channel</td>
</tr>
<tr>
<td></td>
<td>2. then Frequency Offset</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press STIMULUS</td>
<td>1. Click Stimulus</td>
</tr>
<tr>
<td>2. then [Frequency Offset]</td>
<td>2. then Frequency</td>
</tr>
<tr>
<td></td>
<td>3. then Frequency Offset</td>
</tr>
</tbody>
</table>
The following are major changes to FOM:

- Stimulus and Response are now called Sources and Receivers.
- Sources and Receivers settings can be made in two ways:
  1. By **Coupling** to the Primary (Channel) settings. This is the only method used in previous releases.
  2. By **Uncoupling** and setting Sources and Receivers values independently. This is the new, simplified method.

- External sources can be controlled from this dialog. Learn more.

**Note:** Source2 supplies power for ports 3 and 4. **Turn Source2 power ON** using the Power and Attenuators dialog. This (Frequency Offset) is the only dialog for controlling the frequency of Source 2. Learn more about Source2.

**Frequency Offset (ON/OFF)** Enables Frequency Offset Mode on ALL measurements that are present in the active channel.

When FOM is NOT enabled, all frequencies are the same as the active channel.

**Tip:** First make other settings on this dialog box, then click **Frequency Offset** ON.

- **Primary** The current Active Channel settings. When a Source or Receiver is coupled to the Primary settings, its Sweep Type is the same as that of the Primary. The frequency settings of the coupled range are mathematically derived from the Primary settings using the Multiplier, Divisor, and Offset values. With this approach, only the Primary settings need to be changed in order to affect change in the coupled Sources and Receivers. Changes to the Primary channel settings occur when Frequency Offset is checked ON. See example using Primary and Coupled setting.

- **Tip:** Primary settings are ONLY used when Sources and Receivers are Coupled. It is often easier to Uncouple, then set Sources and Receivers independently.

Source and Source2 if available. Learn more about Internal Second Source.

- **Receivers** All receivers that are used in the channel, including Reference receivers, are tuned to the specified frequency settings.
Mode

**Coupled**  Source and Receiver settings are mathematically derived from the Primary settings using Multiplier, Divisor, and Offset values. [Learn more.]

**Uncoupled**  Source and Receiver settings are entered independently, without reference to Primary settings. When Uncoupled, Source and Receiver Ranges can use separate sweep types.

Sweep Type  Click to change the type of sweep for each range. Only available for Primary and Uncoupled Sources and Receivers.

Unsupported Sweep Type combinations

- Power Sweep and Segment Sweep can NOT be used together.
- Uncoupled Log Sweep yields invalid data whenever the sources are offset from the receivers.
- Coupled Log Sweep is allowed only for the following two conditions:
  1. The offset = 0, the multiplier = 1, and the divisor = 1.
  2. The multiplier = 0

Settings  To change settings, click IN the appropriate Settings cell, then click Edit.

- If coupled, invokes the **Coupled dialog**.
- If uncoupled or Primary invokes the **Uncoupled settings dialog**.

X-Axis  Select the settings to be displayed on the X-Axis.

X-Axis Point Spacing  Only available when a Segment Sweep Type is selected as the X-Axis display. [Learn more.]

**Note:**  When Frequency Offset is enabled, ALL receivers on the channel, including the reference receivers, tune to the new offset frequencies, Therefore the source and reference receiver will be at different frequencies. Therefore, FOM measurements that include a reference receiver, which includes all S-parameters, display invalid data.

To measure and display measurements at both the source and receiver frequencies, you must use two channels. Use Equation Editor to calculate the conversion loss. [See a calibrated FOM conversion loss example.]

[Learn how to calibrate frequency offset measurements.]
Coupled settings dialog box help

Coupled Formulas:

Range Start = [Primary Start x (Multiplier / Divisor)] + Offset
Range Stop = [Primary Stop x (Multiplier / Divisor)] + Offset

Where:

Offset  Specifies an absolute offset frequency in Hz. For mixer measurements, this would be the LO frequency. Range is +/- 1000 GHz. Offsets can be positive or negative.

Multiplier  Specifies (along with the divisor) the value to multiply by the stimulus. Range is +/- 1000.

- Negative multipliers cause the stimulus to sweep in decreasing direction. For downconverter mixer measurements, this would be for setups requiring the Input frequency to be less than LO frequency. See an example.
- 0 (zero) as the multiplier nulls the Primary setting. Then the Offset value adds to zero.

Divisor  Specifies (along with the multiplier) the value to multiply the stimulus. Range is 1 to 1000.
Primary and Uncoupled settings dialog box help

This dialog will vary depending on the sweep type:

**Linear and Log frequency**

Uncoupled Log sweep yields **invalid data** whenever the sources are offset from the receivers.

Select Start/Stop or Center/Span

**Frequency** Enter values

**Points** (Primary only) Enter number of data points for the sweep.

**Power**

**CW Freq** Enter frequency in Hz.

**Points** (Primary only) Enter number of data points for the power sweep.

**CW Time**

**CW Freq** Enter frequency in Hz.

**Sweep Time** Enter time to complete one sweep. Enter 0 for the fastest sweep.

**Segment Sweep** Edits are made exactly like the standard segment table.

For Advanced Users: Uncoupled Segment Sweep offers great flexibility in configuring measurements. In segment sweep mode:

- The **OK** button is NOT available until the total number of data points for all segments matches the number of Primary data points.

- **Independent IF Bandwidth** and **Independent Sweep Time** are available ONLY on the Primary (channel) and the Uncoupled **Receivers** - NOT Sources.

- **Independent Power** is available ONLY on the Primary (channel) and the Uncoupled **Sources** - NOT Receivers.

**Setup Examples**

Although the Frequency Offset settings can be used with many types of devices, these examples include mixer terminology.

See a Mixer Compression and Phase (AM-PM) Measurement using FOM.

See a calibrated FOM conversion loss example.

1. **Fixed LO - Upconverter**

- **Swept Stimulus (Mixer Input)**: 1000 MHz - 1200 MHz

- **Fixed LO**: 1500
1. **Swept Response (Mixer Output):** 2500 MHz to 2700 MHz

Make the following settings on the FOM dialog

**Source:** Uncoupled
- Sweep Type: Linear
- Click Settings, then Edit. In the Source dialog:
  - Start Frequency = 1000 MHz
  - Stop Frequency = 1200 MHz

**Receiver:** Uncoupled
- Sweep Type: Linear
- Click Settings, then Edit. In the Receiver dialog:
  - Start Frequency = 2500 MHz
  - Stop Frequency = 2700 MHz

**LO Settings**
Set external source to CW - 1500 MHz.

**Source2:** Uncoupled (Only with Second PNA Internal Source)
- Sweep Type: CW Time
- Click Settings, then Edit. In the Source2 dialog:
  - CW Frequency = 1500 MHz

2. **Fixed LO - Downconverter (Input < LO)**

- **Swept DECREASING Stimulus (Mixer Input):** 1100 MHz to 1000 MHz
- **Fixed LO:** 2500 MHz
- **Swept INCREASING Response (Mixer Output):** 1400 MHz to 1500 MHz

Make the following settings on the FOM dialog

**Primary:** Not used

**Source** (Input): Uncoupled
- Sweep Type: Linear
- Click Settings, then Edit. In the Source dialog:
  - Start Frequency = 1100 MHz
  - Stop Frequency = 1000 MHz

**Receiver** (Output): Uncoupled
- Sweep Type: Linear
- Click Settings, then Edit. In the Receiver dialog:
  - Start Frequency = 1400 MHz
Stop Frequency = 1500 MHz

**LO Settings**
Set external source to CW - 2500 MHz.

**Source2**: Uncoupled (Only with [Second PNA Internal Source](#))
- Sweep Type: CW Time
- Click Settings, then Edit. In the Source2 dialog:
  - CW Frequency = 2500 MHz

---

### 3. Swept LO - Fixed Output - Upconverter

Swept External LO measurements in Frequency Offset Mode can be very difficult. The external LO source must be synchronized with the swept output or input (as in this case). See [Synchronizing and External Source Control](#) to see how this is done. The Frequency Converter Application Opt 083 performs makes these measurements easily.

- **Swept Stimulus (Mixer Input)**: 1000 MHz to 1100 MHz
- **Swept LO**: 1500 MHz to 1400 MHz
- **Fixed Response (Mixer Output)**: 2500 MHz

Make the following settings on the FOM dialog

**Source**: Uncoupled
- Sweep Type: Linear
- Click Settings, then Edit. In the Source dialog:
  - Start Frequency = 1000 MHz
  - Stop Frequency = 1100 MHz

**Receiver**: Uncoupled
- Sweep Type: CW Time
- Click Settings, then Edit. In the Receiver dialog:
  - CW Frequency = 2500 MHz

**LO Settings**

- If using external source, set to sweep from 1500 - 1400 MHz.
- If using **Source2** ([Second Internal Source](#)): set to Uncoupled, then:
  - Sweep Type: Linear
  - Click Settings, then Edit. In the Source2 dialog:
    - Start Frequency = 1500 MHz
    - Stop Frequency = 1400 MHz
4. **Power Sweep for Mixers**

To measure the gain compression of a mixer, the input power to the mixer is swept. The input and output frequencies are fixed but offset from one another.

This is a good use of Coupled settings because the same compression test can be performed at several different frequencies. With coupled Source and Receiver ranges, the Primary (channel) frequency can be easily changed from the front panel. The coupled source and receiver frequencies will update accordingly.

- **Swept Input Power**: -10 dBm to 0 dBm
- **Fixed Input Frequency**: 1500 MHz
- **Fixed LO**: 500 MHz
- **Fixed Output**: 2000 MHz

Make the following settings on the FOM dialog

**Primary**:
- Sweep Type: Power Sweep
- Click Settings, then Edit. In the Primary dialog:
  - CW Frequency = 1500 MHz

**Source**: Coupled
- Default settings make CW Frequency: 1500 MHz (same as Primary)

**Receiver**: Coupled
- Default settings make Sweep Type: CW Time
- Click Settings, then Edit. In the Receiver dialog:
  - Offset = 500 MHz

**LO Settings**
- If using external source, set to CW: 500 MHz.
- If using **Source2** ([Internal Second Source](#)), set to Coupled, then:
  - Sweep Type: Power Sweep
  - Click Settings, then Edit. In the Source2 dialog:
    - CW Frequency = 500 MHz

---

**Test Set Reference Switch**

PNA models with [option 081](#) have a switch in the test set that allows you to bypass the port 1 reference receiver through the front panel Reference 1 connectors. This switch lets you easily switch between standard S-Parameter measurements and measurements using a reference mixer. You could use this feature to make standard S11 measurements and converter transmission measurements relative to a reference (“golden”) mixer.

**Note**: The Frequency Converter Application [Option 083](#) simplifies the task of making extremely accurate phase
measurements on MOST frequency converting devices.

### How to access the Test Set dialog box

#### Using front-panel HARDKEY [softkey] buttons

<table>
<thead>
<tr>
<th>For N5230A and E836xA/B models</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. <strong>Navigate using MENU/DIALOG</strong></td>
<td>1. Click <strong>Channel</strong></td>
</tr>
<tr>
<td>2. then <strong>[Channel]</strong></td>
<td>2. then <strong>Test Set</strong></td>
</tr>
<tr>
<td>3. then <strong>[More]</strong></td>
<td></td>
</tr>
<tr>
<td>4. then <strong>[Path Config]</strong></td>
<td></td>
</tr>
<tr>
<td>5. <strong>select</strong></td>
<td></td>
</tr>
</tbody>
</table>

#### For PNA-X and 'C' models

| 1. Press **TRACE/CHAN** | 1. Click **Trace/Chan** |
| 2. then **[Channel]** | 2. then **Channel** |
| 3. then **[More]** | 3. then **More** |
| 4. then **[Path Config]** | 4. then **Path Config** |
| 5. **select** | 5. **Select** |

### Test Set dialog box help

**Note:** This feature is only available on PNA models with **Option 081** - external reference switch.

#### R1 Input Path

- **Internal: bypass R1 Loop**  Connects the port 1 source directly to the R1 receiver.
- **External: flow through R1 Loop**  Allows direct access to the R1 receiver through the Reference 1 front-panel connectors.

See [block diagram of reference switch](#).
25-Feb-2008  Added link to AM-PM procedure
16-Oct-2007  Minor edits
11/21/06  MQQ Modified for new dialog
Many frequency offset measurements can be made using the PNA with option 080. The following is a list of some of those measurements and how they are made.

- Conversion Loss
- Conversion Compression
- Return Loss and VSWR
- Isolation
- Harmonic Distortion

See Also: Frequency Offset Measurement Accuracy
Frequency Offset Measurement Accuracy

This topic discuss methods that can be used to make accurate frequency offset measurements.

**Calibrations**

**Mismatch Errors**

**Accurate and Stable LO**

See other Mixer Measurement topics

**Calibrations**

With Frequency Offset measurements, the stimulus and response frequencies are different. Standard calibration error terms are calculated using reference measurements. Therefore, traditional calibration methods such as full 2-port SOLT cannot be used with frequency offset.

*Source and Receiver Power calibrations* can be used to calibrate your Frequency Offset measurements.

*Frequency Converter Application* (option 083) offers fully calibrated scalar and vector frequency offset measurements.

**Source Power calibration:**

- Sets accurate power level at stimulus frequencies regardless of the receiver that will be used in the measurement.
- Can be copied to other channels with copy channels feature.
- Can be interpolated.

**Receiver Power Cal:**

- Requires a source cal to have already been performed and applied.
- Cannot be copied to other channels.

**Therefore:**

- Start by performing a source power cal over the combined stimulus and response frequencies.
- **Copy the channel** to other needed channels and the source power cal is copied.
- Change the frequency range of the copied channel to response frequencies.
- Perform a receiver cal at the response frequencies on individual channels.
- Change the frequency range to stimulus frequency and switch frequency offset ON.
- On Status Bar, ensure that source and receiver cals are ON (source cal will be interpolated).

See Frequency Offset Conversion Loss Measurements to see a step-by-step example.

**Mismatch Errors**

Mismatch errors result when there is a connection between two ports that have different impedances. With S-parameter measurements, these mismatches are measured and mathematically removed during a full 2-port calibration. This is much more difficult with frequency offset measurements. A much easier solution is to use high-quality attenuators on the input and output of the mixer.

By adding a high-quality attenuator to a port, the effective port match can be improved by up to twice the value of the attenuation. For example, a 10-dB attenuator, with a port match of 32 dB, can transform an original port match of 10 dB into an effective match of 25 dB. However, as the match of the attenuator approaches the match of the original source, the improvement diminishes.

**Note:** The Frequency Converter Application (option 083) uses calibration techniques that correct for mismatch errors.

![Mismatch Errors Graph]

The larger the attenuation, the more nearly the resulting match approaches that of the attenuator, as shown in the following graphic. However, excessive attenuation is not desired because that will decrease the dynamic range of the measurement system.

**Accurate and Stable LO**

When using frequency offset mode, if the LO signal is not accurate and stable, the output signal will not be at the expected response frequency. As a result, the output signal can fall on the skirts of the PNA receiver IF filter, or fall completely outside of the receiver filter passband.

Also, the LO power level is critical in mixer measurements. Be sure to monitor these power levels closely.
Conversion Loss (or Gain)

**What is Conversion Loss?**
Conversion loss is defined as the ratio of the power at the output frequency to the power at the input frequency with a given LO (local oscillator) power. This is illustrated in the graphic below. A specified LO power is necessary because conversion loss varies with the level of the LO, as the impedance of the mixer diode changes.

**Why Measure Conversion Loss?**
Conversion loss (or gain in the case of many converters and tuners) is a measure of how efficiently a mixer converts energy from the input frequency to the output frequency. If the conversion loss response of a mixer or converter is not flat over the frequency span of intended operation, valuable information may be lost from the resulting output signal.

**How to Measure Conversion Loss**
Conversion loss is a transmission measurement. It is measured by applying an input signal (stimulus) and an LO signal at specific known power levels, and measuring the resulting output signal level. Because the output frequency is different from the input frequency, frequency offset mode (option 080) must be used for this measurement.
Note: This measurement is made much easier if your PNA has the Frequency Converter Application

Equipment Setup

![Equipment Setup Image]

Example: A calibrated Conversion Loss (Down-converter) measurement

Swept Input with Fixed LO = Swept Output

- RF Input: 3.1 - 3.3 GHz
- LO: 2.2 GHz
- IF Output: 900 - 1100 MHz

PNA setup and calibrate on channel 1

1. On channel 1 create an unratioed R measurement over the ENTIRE input and output frequency span (.9 - 3.3 GHz). This will be the base source power cal that will be copied to the R and B channel measurements.

2. Perform a source calibration using a power meter. This makes the power level at the input of the mixer very accurate.

Setup Reference measurement on channel 2

1. Copy channel 1 to channel 2 which will display the reference input to the mixer. The channel 1 source power cal is copied with the other channel settings.

2. Change measurement to R1 unratioed.

3. Change RF Input frequency to 3.1 - 3.3 GHz. The source power cal becomes interpolated.

4. Perform receiver power cal. Do not need to make physical connections. The PNA source is internally connected to the R1 receiver. Makes the R receiver read the source power level.
Setup B measurement on channel 3

1. Copy channel 1 to channel 3. This channel will display the output of the mixer. The channel 1 source power cal is copied with the other channel settings.

2. Change measurement to B unratioed.

3. Change IF Output frequency to .9 - 1.1 GHz. This causes the source power cal becomes interpolated.

4. Connect thru line from port 1 to port 2.

5. Perform receiver power cal. This makes the B receiver read the source power at the IF Output frequencies.

6. Turn OFF receiver power cal. This prevents an error when changing to input frequencies (next step).

7. Change RF Input frequency to 3.1 - 3.3 GHz. This changes the channel back to the mixer RF Input frequencies.

8. Enable Frequency Offset.

9. Change Offset to (-2.2 GHz). This tunes the B receiver to the IF Output frequencies .9 to 1.1 GHz. Note: The minus sign indicates a down-converter measurement.

10. Turn ON receiver power cal.

Measure the Mixer

1. Connect the mixer.

2. Adjust scaling to suit your needs.

3. Enable markers to read power levels for each trace.

The display below shows:

- Ch3 B receiver (bottom trace) absolute output power.
- Ch2 R1 receiver measurement (top trace) absolute input power to the mixer.

With this method, the conversion loss math (B/R1) can be performed with Equation Editor (not shown). The B/R1 ratio measurement is not supported with receiver power Cal turned on. However, conversion loss (C21) measurements can be made directly and are much easier using the Frequency Converter Application, FCA (Opt 083).
Conversion Compression

What is Conversion Compression?
Conversion compression is a measure of the maximum input signal level for which a mixer will produce linear operation. It is very similar to the gain compression experienced in amplifiers.

To understand conversion compression, you must first understand conversion loss. This is the ratio of the mixer output level to the mixer input level. This value remains constant over a specified input power range. When the input power level exceeds a certain maximum level, the constant ratio between input and output power levels begins to change. The point at which the ratio has decreased 1 dB is called the 1-dB compression point. This is illustrated in the graphic below.

Why Measure Conversion Compression?
Conversion compression is an indicator of the dynamic range of a device. Dynamic range is generally defined as the difference between the noise floor and the 1-dB compression point.

How to Measure Conversion Compression
The equipment and setup used to measure conversion compression are essentially the same as for measuring...
conversion loss and is illustrated in the following graphic.

The PNA performs a power sweep using frequency-offset mode and the resulting display shows the mixer's output power as a function of its input power. The 1-dB compression point (or others such as 3-dB) can be determined using markers.

Measurement Accuracy Considerations

Equipment Setup Considerations

- The couplers in the PNA have very good directivity. If the return loss of the DUT is bad, the reflected signal gets sampled by the PNA and can result in errors. This relates to error in DUT gain. To increase the accuracy, an attenuator can be added between the PNA's source port and the DUT's input port. Normally a 6- to 10-dB attenuator is sufficient. Addition of this attenuator, however, decreases the available drive to the DUT.

- With high drive levels the PNA can be driven into compression resulting in measurement error. With excessive drive levels, the PNA can be damaged. Add an attenuator between the output of the DUT and the receiver input of the PNA to avoid these problems.

Calibration Considerations

- Source power calibration can be used to provide a high level of accuracy for this measurement.

- If your PNA has the Frequency Converter Application (option 083), you can perform a Scalar Mixer Calibration to obtain a more accurate measurement.
What is Isolation?
Isolation is a measure of the leakage, or feedthrough, from one port to another. The more isolation a mixer provides, the lower the amount of feedthrough. Isolation is measured at the same frequency as the stimulus, not the converted or shifted frequency. Therefore, Frequency Offset capability is not necessary for these measurements.

Three main isolation terms are of interest for mixer measurements:
- LO-to-OUT isolation ($V_{LO}$)
- LO-to-IN isolation ($V_{LO}$)
- IN-to-OUT feedthrough ($V_{IN}$)

Why Measure Isolation?
Any unwanted signal "leaking" through the device will mix with the desired output signal creating intermodulation products, adding to intermodulation distortion. These unwanted signals may be difficult to filter out.

How to Measure Isolation
Use the following setups to measure the isolation of a mixer:
Note the following:

- The Input to Output isolation is very dependent on the LO power level. Isolation should be measured with the LO power at its normal operating level.

- Each of the ports not being tested should be terminated with an impedance typical of actual operation. This may not always be the characteristic impedance, Z0 (usually 50 or 75 ohms). For example, if the OUT port of a mixer is intended to be directly connected to a filter, then this filter should be used when measuring the LO-to-IN feedthrough.
Measuring Converters vs. Mixers

Measuring IN-to-OUT feedthrough of a converter is identical to that of a mixer. The IN-to-OUT feedthrough is generally very small for a converter due to the inclusion of an IF filter in the device. Because of this, the measurement may require the PNA to have increased dynamic range.

Measuring LO leakage (LO-to-OUT and LO-to-IN) of a converter requires a different technique because the LO port is typically not accessible:

- The PNA can be tuned to the frequency of the LO signal and either the OUT or IN port connected to the PNA receiver port. The PNA source port is not connected.

- A spectrum analyzer can be connected to either the OUT or IN port and tuned to the frequency of the LO signal.
Harmonic Distortion

What is Harmonic Distortion?
Harmonics are multiples of any signal appearing at the mixer input and also multiples of the LO input. The distortion of the mixer's output characteristics caused by these harmonics is referred to as harmonic distortion. Harmonic distortion is caused by non-linearities in the device.

Harmonics are NOT signals created by two or more signals interacting (mixing); these signals are known as intermodulation products, which result in intermodulation distortion.

Why Measure Harmonic Distortion?
- It can degrade the performance of devices connected to the output of the mixer.
- The harmonics can also mix with other signals present in the mixer, adding to the intermodulation distortion of the mixer.

How to Measure Harmonic Distortion
The harmonics can be measured using the PNA with Frequency Offset (option 80). The frequency of the LO to the mixer is set to zero and multiplier of the RF input is used to set the IF frequency (the harmonic). The equipment setup is shown below.

Since harmonics are specified in dBc, the fundamental RF and both the second and third harmonics are measured and the differences calculated. Multiple channels can be used to do this.

1. Connect the equipment.
2. Setup the measurement for calibration. See also Measurement and Accuracy Considerations.
   - Use three channels and frequency offset mode:
     - Channel 1 = F1 to F2
     - Channel 2 = F1 to 2F2 (frequency offset mode, multiplier = 1)
     - Channel 3 = F1 to 3F2 (frequency offset mode, multiplier = 1)
• Perform a source power calibration and receiver power calibration over the entire frequency range. See Measurement and Accuracy Considerations.

• Reduce the frequency span and increase the frequency offset multiplier on Channels 2 and 3:
  Channel 2 = F1 to F2 (frequency offset mode, multiplier = 2)
  Channel 3 = F1 to F2 (frequency offset mode, multiplier = 3)
  **Note:** Because the frequency span has been changed from that used for calibration, the source and receiver calibrations will be interpolated.

• Connect the DUT, make the measurement, and calculate the harmonic response:
  Set up markers on Channels 1, 2 and 3, and determine the difference between the marker values to get the dBc value of each harmonic.
  Channel 1 - Channel 2 = 2nd harmonic (dBc)
  Channel 1 - Channel 3 = 3rd harmonic (dBc)
  **Note:** Be sure to set the markers to the appropriate stimulus. Channel 2 markers should be set to twice the frequency of Channel 1 markers. Channel 3 markers should be set to three times the frequency of Channel 1 markers.

**Measurement and Accuracy Considerations**

**Equipment Setup Considerations**

• A filter must be used at the input of the mixer to remove the PNA source harmonics.

**Calibrations**

• If your PNA has the Frequency Converter Application (FCA), you can perform a Scalar Mixer Calibration to obtain a more accurate measurement.
Return Loss and VSWR

What are Return Loss and VSWR?

Return loss and VSWR are both linear reflection measurements, even when testing frequency conversion devices, because the reflected frequency is not converted. These measurements are essentially the same as for filters and amplifiers. Learn more about Reflection Measurements.

Why Measure Return Loss and VSWR?

Devices which have poor return loss and VSWR result in loss of signal power or degradation of signal information.

How to Measure Return Loss and VSWR

Setup the PNA measure return loss and VSWR as you would any two-port device. Connect your frequency converting device as shown in the following diagrams:

![RETURN LOSS AND VSWR OF MIXER INPUT PORT](image)

![RETURN LOSS AND VSWR OF MIXER OUTPUT PORT](image)
RETURN LOSS AND VSWR OF MIXER LO PORT
Frequency Converter Application Known Issues

To see the current list of known FCA issues, please visit [http://na.tm.agilent.com/fca/](http://na.tm.agilent.com/fca/) and click the known FCA issues link.
Frequency Converter Application

The Frequency Converter Application (Option 083) simplifies testing of frequency converting devices.

**Note:** Option 082 allows you to make only SMC calibrations and measurements.

- Advanced calibration techniques that provide exceptional amplitude and phase accuracy.
- Simple setup using PNA models with **Internal Second Source**.
- Control of external signal sources for use as local oscillators.

- A graphical set-up dialog box that lets you:
  - quickly set up the PNA for single or dual conversion devices.
  - calculate and choose where mixing and image products will fall.

For more information, see the following topics:

- Using FCA
- Configure Your Mixer
- FCA Calibrations
- Configure an External LO Source
- SMC with a Booster Amp
- Characterize Adaptor Macro
- Measure a DUT with an Embedded LO

**Examples**

- How to make a VMC Measurement
- How to make an SMC Measurement
Notes:

- For a detailed understanding of FCA, see our [Mixer Measurements App Notes](#).
- Please submit FCA issues that you find, as well as enhancement requests, to fca_support@agilent.com. See [Known Issues with the FCA](#).
- FCA is **NOT** supported on PNA-L Models. However, [Opt 082](#) (SMC only) IS supported on PNA-L Models.
- FCA is **NOT** supported when using external [Millimeter Modules](#).
- [Copy Channels](#) does NOT work with FCA.

Last modified:

- 13-Feb-2008 Added note about Copy Channels
- Nov 28, 2006 Added clarification to PNA-L note
Using the Frequency Converter Application (Option 083)

- **What's New in FCA**
- **Overview**
- **How to Create a Measurement**
- **FCA Measurements Offered**
- **FCA Measurement Settings**
  - Change a Measurement
  - Speed Up Swept LO SMC Measurements
  - Use Nominal Incident Power
  - Select X-axis Display
  - Save Trace Data
  - Avoid Spurs

**Examples** (not in this topic)

- **How to make a VMC Measurement**
- **How to make an SMC Measurement**

**Note:** Please submit FCA issues that you find, as well as enhancement requests, to fca_support@agilent.com (See Known FCA Issues.)

Not sure if your analyzer is equipped with Option 083? [Here's how to identify your analyzer.](#)
What’s new in FCA with Rev 7.2

- Support for PNA-X and Internal Second Source as LO

What’s new in FCA with Rev 6.2

- Option 082 allows you to make SMC calibrations and measurements. (VMC is NOT available.)

What’s new in Rev 6.0:

- Calibrated Swept LO measurements.
- Create any of the Mixer measurements that are offered. For example, in the past if you wanted an SC12 measurement, you first had to create an SC21 measurement, and then change it to SC12. You can also create more than one mixer measurement at the same time.
- FCA calibrations are streamlined for consistency and ease of use.
- Added SMC Power meter and offset settings.
- Embed/De-embed networks for Waveguide, in-fixture, or on-wafer measurements.
- Characterize Adaptor Macro creates S2P files from two 1-port cal Sets.
- SMC-Forward and SMC-Reverse measurements can now be performed in the same channel. Therefore, we no longer refer to them as separate measurement types.
- Previous Instrument State files that include an FCA measurement can NOT be recalled by Revision 6.0.
- FCA is NOT supported when using External Test Set Control.

Overview

The following is an overview of how to make an FCA measurement:

1. DECIDE to make either a Scalar measurement or Vector measurement. The calibration method is unique to each of these. See a comparison of these two measurement types.
2. CREATE one or more FCA Mixer measurements.
3. Setup and CALIBRATE your Scalar or Vector measurement.
How to Create an FCA Measurement

**Note:** An FCA measurement and a non-FCA measurement can NOT reside on the same channel.

- **PNA-X:** First assign a VMC or SMC measurement class to a channel. [Learn how](#).
- **E836x and PNA-L:** From the [New Trace dialog](#), click **Application**

Then change the default measurement to one you choose by doing the following:

How to Change an FCA Measurement

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press <img src="image" alt="Measure" /></td>
<td>1. Click <strong>Trace</strong></td>
</tr>
<tr>
<td>2. then <img src="image" alt="S11" /> <img src="image" alt="SC21" /> <img src="image" alt="S22" /> <img src="image" alt="Incident Power" /></td>
<td>2. then <strong>Measure</strong></td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press <strong>Meas</strong></td>
<td>1. Click <strong>Trace/Chan</strong></td>
</tr>
<tr>
<td>2. then select parameters</td>
<td>3. then <strong>Measure</strong></td>
</tr>
<tr>
<td></td>
<td>3. then select parameters</td>
</tr>
</tbody>
</table>

**New FCA Measurement** dialog box help

Select one or more Scalar Mixer or Vector Mixer measurements.

- SMC and VMC measurements MUST be made on separate channels.
- After you create a mixer measurement, you can [configure the FCA measurement](#) and make other FCA settings.

**FCA Measurements Offered**

Learn how to [change the FCA measurement](#).
Important Note: Connecting your DUT to the PNA using FCA:

RF and IF terminology is NOT used in the FCA because the PNA does not know how the DUT is labeled or how it will be used. Instead, the general terms INPUT and OUTPUT are used.

- **INPUT** - The DUT port connected to PNA Port 1.
- **OUTPUT** - The DUT port connected to PNA Port 2.

INPUT and OUTPUT Frequencies are specified using the Configure Mixer dialog box.

Vector Mixer/Converter Measurements

- **VC21 Conversion Loss/Gain (default)** - stimulus at Input, response at Output
- **S11** - stimulus and response at Input
- **S22** - stimulus and response at Output
- **R1** - stimulus at Input, measures absolute power at the R1 receiver (uncorrected)
- **B** - stimulus at Input, measures absolute power at the B receiver (uncorrected)
- **VC12** (reverse conversion loss) is NOT offered because of the reference mixer.

See Also: Measure a DUT with an Embedded LO

Scalar Mixer/Converter Measurements

![Diagram of PNA with ports labeled INPUT and OUTPUT](image)

Ratioed

- **SC21 Conversion Loss/Gain** - stimulus at Input, response at Output
- **SC12 Conversion Loss/Gain** - stimulus at Output, response at Input
- S11 stimulus and response at Input
- S22 stimulus and response at Output

Unratioed These measurement do NOT use a reference receiver.

- IPwr (Incident Power) - stimulus and response at Input
- RevIPwr (Reverse Incident Power) - stimulus and response at Output
- OPwr (Output Power) - stimulus at Input, response at Output
- RevOPwr (Reverse Output Power) - stimulus at Output, response at Input

See Also: SMC with a Booster Amp

Channel / Window Selections

Channel Number Select the channel for the new traces.

Create in New Window

- Check to create new traces in a new window.
- Clear to create new traces in the active window. When the PNA traces per window limitation has been reached, no more traces are added.

Auto-Create Windows Check to create new traces in as many windows as necessary. See PNA number of windows limitation.

FCA Measurement Settings

Most of the FCA measurement settings in the remainder of this topic are made using the following menu selection. The choices will be slightly different depending on the active FCA measurement.
How to select several FCA measurement settings

1. First create an FCA measurement, then...

Using front-panel HARDKEY [softkey] buttons

For N5230A and E836xA/B models

1. Navigate using MENU/DIALOG

For PNA-X and 'C' models

1. Press MEAS
2. then select setting

PNA Menu using a mouse

1. Click Trace
2. then Measure
3. select setting

Speeding Up Swept LO SMC Measurements

Swept LO measurements require that an external LO source step in frequency. This can be extremely slow depending on your measurement setup. The following features together will significantly speed up your SMC (NOT VMC) swept LO measurement:

- BNC External LO trigger method
- Use Nominal Incident Power
- Apply Cal Set or Cal Type

Use Nominal Incident Power

Each data sweep of a fully corrected SC21 measurement actually requires FOUR data sweeps. Three of the sweeps are not displayed. When you select Use Nominal Incident Power, the reference receiver (R1 or R2) does not measure incident power. Instead, the incident power is assumed to be at the level that was set with the Source Power Calibration that is done as part of every SMC measurement. The degradation in accuracy is very negligible if the input or output of your test device is well-matched. This selection eliminates sweeps ONLY when either:

- Output Power is measured OR
- SMCRsp is applied.
This selection applies to all SMC measurements. This selection never eliminates VMC sweeps.

See how to select Use Nominal Incident Power.

**Select X-axis Display for FCA Measurements**

FCA measurements typically have more than one swept parameter. You can choose to view the response (output) of the measurement on the Y-axis while displaying any of the swept parameters (Input, LO1, LO2, Output) on the X-axis of the PNA display.

For example, the following image shows an SMC Fixed Output response versus the swept Input.

- **Output**: 100 MHz (data trace)
- **Input**: 2 GHz to 23 GHz (X-axis)
- **LO**: 1.9 GHz to 22.9 GHz (not shown)

Marker annotation shows Output power at Input frequency.

See How to Select X-axis Display

**Save Trace Data**

You can save your Frequency Converter measurement data in S2P format to disk.

**Note:** This is the only method to save Frequency Converter .S2P files from the front panel. Do NOT click File, Save As... to save these S2P data files.

Beginning with PNA release 6.03, save FCA .S2P files remotely using the standard Save SNP programming commands.

See How to Save Trace Data
**Save Data to File** dialog box help

Allows you to save Frequency Converter measurement data to an S2P file. The data is saved in S2P format much like standard PNA data. [Learn more about S2P files.]

**Note:** This is the only method to save Frequency Converter .S2P files. Do NOT click **File, Save As...** to save these S2P data files.

**S2P Data Format**  Select the data format. This selection is independent of the PNA display.

**Save As**  Click to specify a file name and location for the saved data.

**Exit**  Closes the dialog box without saving the data. To save the data, you must click on the Save As button before clicking the Exit button.

**Notes:**

Each record contains 1 stimulus value and 4 parameters (total of 9 values) as follows:

Stim Real(p1) Imag(p1) Real(p2) Imag(p2) Real(p3) Imag(p3) Real(p4) Imag(p4)

where \( p_X \) is the parameter depending on measurement type:

<table>
<thead>
<tr>
<th>Measurement Type</th>
<th>p1</th>
<th>p2</th>
<th>p3</th>
<th>p4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalar</td>
<td>S11</td>
<td>SC21 (FWD)</td>
<td>SC12 (REV)</td>
<td>S22</td>
</tr>
<tr>
<td>Vector</td>
<td>S11</td>
<td>VC21</td>
<td>VC12</td>
<td>S22</td>
</tr>
<tr>
<td>Mixer Characterization</td>
<td>Directivity</td>
<td>Source Match</td>
<td>Reflection Tracking</td>
<td>M21</td>
</tr>
</tbody>
</table>

- If correction is OFF, data is only saved for the active parameter. Zeros are saved for all other parameters.
- If correction is ON, data is saved for all of the parameters.

All files contain the following Header Information: Brackets [ ] contain parameters.

\![Agilent [Instrument Model Number]: [version]]
\![Mixer S2P File: [Mixer Measurement Type]]
\![Parameters: [Parameter List]]
\![Calibration State: [On/Off]]
Avoid Spurs

The Avoid Spurs feature of the Frequency Converter Application attempts to prevent unwanted mixing products from appearing on the PNA screen. The Avoid Spurs feature does not significantly impact measurement speed.

**Note:** The Avoid Spurs feature is OFF by default for FCA calibrations. For highest accuracy, make measurements with the Avoid Spurs feature at the same state (ON or OFF) as was used when calibrating.

- To enable Avoid Spurs, check **Avoid Spurs** on the **Mixer Setup** dialog box.

**Description**

A spur, or spurious signal, is a term used to describe the unwanted product of two signals mixing together. When you configure the mixer setup dialog box for a desired Output, the PNA computes the frequencies of potential unwanted signals. By manipulating internal PNA hardware, these signals are avoided and do not appear on the PNA display. This means you do not need to use external filters to prevent spurious signals from appearing on the PNA display.

The time required for the PNA to compute the frequencies of unwanted spurious signals MAY be noticeable depending on the number of data points in your measurement. However, once computed, the time required for the PNA to avoid the spurs is usually insignificant.

**Limitations**

The Avoid Spurs utility cannot avoid every spur. However, when there is a choice of spurs to avoid, it will avoid the largest spur.

**The Computation of Avoided Spurs**

The Avoid Spur computer avoids the following spurs:

- LO, and its interaction with internal PNA components, and 16 of its harmonics.
- Input frequencies and 16 of its harmonics.
- Undesired Image frequencies (Sum or Difference) and 16 of its harmonics.

---

Last modified:

5-Oct-2007 Added link to embedded LO
Frequency Converter Application (Option 083) offers two advanced calibration choices for mixer or converter measurements that provide exceptional amplitude and phase accuracy.

**Note:** Option 082 allows you to make only SMC calibrations and measurements.

- Comparison of Scalar and Vector Mixer Cals
- SMC Setup and Overview
- VMC Setup and Overview
- FCA Calibration Wizard
- How to Perform an FCA Calibration
- Apply an FCA Cal Set and Cal Type

**Examples** (not in this topic)

- How to make a VMC Measurement
- How to make an SMC Measurement

Not sure if your analyzer is equipped with Option 083? [Here's how to identify your analyzer.](#)

To learn more about the FCA capability and improving FCA measurement accuracy, see [FCA App notes.](#)

Please submit FCA issues that you find, as well as enhancement requests, to [fca_support@agilent.com](mailto:fca_support@agilent.com)  [See Known FCA Issues.](#)

**Other Frequency Converter Application topics**

**Comparison of Scalar and Vector Mixer Cals**

<table>
<thead>
<tr>
<th></th>
<th>Scalar Mixer Calibration</th>
<th>Vector Mixer Calibration</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Overview</strong></td>
<td>Provides highest Scalar (amplitude only) accuracy for measurements of conversion loss/gain.</td>
<td>Provides unparalleled accuracy for measurements of relative phase and absolute group delay.</td>
</tr>
<tr>
<td></td>
<td>Combines SOLT and power-meter calibration.</td>
<td>Uses combination of SOLT standards and a reciprocal mixer/filter pair during calibration.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>After calibration, both reciprocal and non-reciprocal mixers and converters can easily be measured.</td>
</tr>
</tbody>
</table>
| **Types of Transmission Measurements** | Both forward (SC21) and reverse (SC12) directions.                  | Amplitude response VC$_{21}$  
Phase response  
Group delay |
| **Equipment Required**     | Power meter and sensor                                                                   | Calibration mixer/filter combination (must be reciprocal $S_{21} = S_{12}$.)  
Reference mixer  
External source |
|                             | **Common equipment for both SMC and VMC**                                                |                                                                                         |
|                             | • Mechanical cal kit or ECal module                                                       |                                                                                         |
|                             | • PNAs with [one GPIB port](https://www.agilent.com) require the Agilent 82357A USB/GPIB Interface (Contact component_test@agilent.com for this product. The 82357B is not a direct replacement.) |                                                                                         |


**SMC Calibration Setup and Overview**
**Note:** When using a PNA-L or PNA-X with Internal Second Source, the external source is NOT necessary. **Learn which PNA ports can be used for the LO.**

Connect **External Source** and **Power Meter** to the PNA GPIB using any of the following methods:

- For PNAs with **two GPIB ports**, connect these devices to the Controller port.

- For PNAs with one GPIB port:
  - The **Agilent 82357A USB/GPIB Interface** - **highly recommended** - allows for the use of a remote PC to control the PNA.
  - The standard GPIB Interface - with the following limitations:
    - The PNA cannot be controlled remotely as talker / listener over GPIB. First put the PNA in System Controller mode. **Learn how.**
    - Available only on PNA releases 4.2 and later.

Learn how to **Configure an External LO Source**

**Overview of the Scalar Mixer Calibration.**
The Calibration Wizard guides you through this process.
1. Connect a power meter / sensor to PNA Port 1. At each step of the input and output frequency, the PNA measures:
   - input match of the power sensor
   - source power of the PNA

2. Perform two 2-port SOLT calibrations: one over the INPUT frequencies and one over the OUTPUT frequencies of the DUT. (If your DUT is a linear device, the calibration uses only the INPUT frequency range.) Use either a mechanical calibration kit or an ECal module.

   **How to configure two power sensors** to cover the SMC measurement frequency range.

   Using a dual channel power meter, with both sensors connected:

   1. At the SMC **Select DUT Connectors** dialog, click **View / Modify Source Cal Settings**
   2. At **Source Calibration Settings** dialog, click **Power Meter Config**
   3. At **Power Meter Settings** dialog, click **Sensors**
   4. At **Power Sensor Settings** dialog, clear the "**Use this sensor only...**" checkbox for both sensors.
   5. Then enter the **Min** and **Max Frequencies** for both sensors.

   During the SMC Cal, you will be prompted to connect each sensor at the appropriate time.
Note: When using a PNA-X with Internal Second Source, the external source is NOT necessary.

- See note regarding LO power out both second source ports
- Learn which PNA ports can be used for the LO.
- Measure a DUT with an Embedded LO

Reference mixer provides a phase reference for the measurements. The reference mixer is connected in the reference receiver path of the network analyzer, between the source out and receiver R1 in ports, as shown below.
The reference mixer is considered part of the test system setup like the test cables. It remains in place during the entire calibration and measurement process. The reference mixer is switched in and out of the measurement path by the PNA as needed. See how to manually switch the reference mixer.

The reference mixer does not need to be reciprocal and does not have to match the calibration mixer or the mixer-under-test in performance. The only requirement of the reference mixer is that it cover the same frequency range as the mixer under-test. In general, it is valuable to select a reference mixer that can be used with a variety of different setups. For example, a broadband mixer can be used in place of several narrow-band alternatives.

A low pass filter on the output of the reference mixer can be used to suppress the LO leakage signal that comes out of the reference mixer output. It is not strictly needed, but ensures that the PNA will not have any source unlock or unlevel errors due to the LO leakage.

- Connect the Reference Mixer INPUT to PNA Ref 1 Source out
- Connect the Reference Mixer OUTPUT to PNA Rcvr R1 In

Calibration mixer/filter is characterized either before or during a VMC calibration. It is used during the VMC calibration as the THRU standard. The calibration mixer/filter combination must meet the following requirements:

- The mixer must be reciprocal over the frequency range of the mixer under test. This means that it has the same magnitude and phase response in the up-converting and
down-converting directions (C21 = C12) as shown in the following diagram.

- If the Input and Output frequency ranges are overlapping, the mixer must have Input to Output Isolation greater than 10 dB more than the conversion loss in the overlapping range.

- The filter must reject the undesired mixing product, and pass the desired mixing product, at the output of the cal mixer. This requirement can be made easier by characterizing the mixer/filter as a downconverter. Learn more.

### Power splitter

#### LO Source(s)

**Note:** When using a PNA-X with Internal Second Source, the external source is NOT necessary.

- See note regarding LO power out both second source ports

- Learn which PNA ports can be used for the LO.

Connect external sources to the PNA GPIB using any of the following:

For PNAs with **two GPIB ports**,
connect to the GPIB Controller port. For PNAs with one GPIB ports:

- The Agilent 82357A USB/GPIB Interface - highly recommended - allows for the use of a remote PC to control the PNA.
- The standard GPIB Interface - with the following limitations:
  - The PNA cannot be controlled remotely as talker/listener over GPIB. First put the PNA in System Controller mode. Learn how.
  - Available only on PNA releases 4.2 and later.

Learn how to Configure an External LO Source

Overview of the Vector Mixer Calibration

The Calibration Wizard guides you through this process. The first three steps characterize the calibration mixer that is used as the THRU standard during the calibration process.

1. Perform a 2-port SOLT calibration over the INPUT frequency range of the DUT, and another 2-port SOLT calibration over the OUTPUT frequency range. Use either a mechanical calibration kit or an ECal module.

2. Characterize the input and output match of the calibration mixer/filter combination with the external LO connected and the output terminated with an open, short, and load. Learn how to connect the calibration mixer/filter. Once characterized, an S2P file is saved and can be recalled for use in subsequent VMC calibrations using the same stimulus settings.

3. Connect the reference mixer between the Source Out and Rcvr R1 front-panel connectors. Connect the output port of the calibration mixer/filter combination to PNA Port 2 (or at the end of the cable attached to the port).

4. Measure the calibration mixer/filter combination as the THRU calibration standard.

5. The PNA calculates the error terms necessary to make corrected phase measurements of your mixer/ converter under test.

To learn more about VMC capability and improving measurement accuracy, see www.Agilent.com and search for App notes (AN 1408-1) and (AN1408-3).

The FCA Calibration Wizard
The following dialog boxes are presented during SMC, VMC, and Mixer Characterization (used in VMC). Click a box to learn about that step.

Note: In the above diagram and following procedure:

- **yellow** - steps that are common to both calibration methods.
- **tan** - VMC only steps.
- **red** - SMC only steps
How to Perform an SMC, VMC, or Mixer Characterization Calibration

1. Create an FCA measurement, then...

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Calibration</td>
</tr>
<tr>
<td>2. then Calibration Wizard or Mixer Characterization</td>
<td></td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td>1. Click Response</td>
</tr>
<tr>
<td>1. Press RESPONSE</td>
<td>2. then Cal Wizard</td>
</tr>
<tr>
<td>2. then [Cal Wizard]</td>
<td></td>
</tr>
</tbody>
</table>

**Calibration Setup** dialog box help

Allows you to review and change the settings for your FCA calibration.

**Waveguide/In-fixture/On-Wafer Setup** Invokes the following Setup dialog box.

**Edit Mixer** Click to display the Configure Mixer dialog box.
Waveguide/In-fixture/On-Wafer Setup dialog box help

This dialog box appears ONLY if you checked the Waveguide/In-fixture/On-Wafer Setup box in the previous Cal Setup dialog.

Allows you to embed (add) or de-embed (remove) circuit networks on the input and output of your mixer measurement.

For Network1 (Input) and Network2 (Output) select Embed, De-embed, or None.

**Browse** Click to navigate to the .S2P file that models the network to embed or de-embed.

**Notes**

- See [To Embed or De-embed?](#) and the associated procedures

- [Characterize Adaptor Macro](#) can be used to create the S2P file.

- The S2P file for Network1 (on the input of the mixer), must cover the Input frequency range. The S2P file for Network2 (on the output of the mixer), must cover the Output frequency range.

- The frequency range of the S2P file must be the same, or larger than, the frequency range of the FCA measurement. If more frequencies are included in the file, and the data points do not exactly match those of the measurement, interpolation will be performed.

- In all cases:
  - Port 1 of each network is assumed to be connected to the PNA.
  - Port 2 of each network is assumed to be connected to the DUT.
**Calibration Mixer Characterization** dialog box help

**VMC and Mixer Characterization ONLY**

**What is Calibration Mixer Characterization?** For a brief explanation, see [Calibration Mixer](#).

**Select Mixer Characterization Method**

**Perform Characterization (requires a reference mixer)** Performs a Mixer characterization in addition to the VMC calibration. The mixer characterization file will be saved at the end for use in subsequent VMC calibrations. Choose this selection if you do NOT already have a mixer characterization file to load.

**Load characterization from file** Loads an S2P calibration mixer characterization file. Click [Browse](#) to locate the file.

- The frequency range of the S2P file MUST be the same, or larger than, the frequency range of the FCA measurement. If the S2P file frequency range is larger, or the data points do not exactly match those of the measurement, interpolation will be performed.

- The VMC calibration requires that the calibration mixer be connected in the same orientation as that in which it was characterized. The direction in which it was characterized is not part of the file that is recalled. You have to remember and connect it appropriately.

"Invalid Mixer Characterization File" is displayed if the frequency range of the S2P file is smaller than those of the measurement.

**Note:** A Mixer Characterization Cal can be performed separately. [Learn how](#).
**Measurement Direction** dialog box help

**VMC and Mixer Characterization ONLY**

This dialog box appears ONLY if your settings in the Mixer Setup dialog box indicate that your DUT is being tested as an upconverter (input < output). It allows you to characterize the Calibration Mixer / Filter as a downconverter (input > output) or an upconverter.

The following example shows why you would choose to characterize the calibration mixer as a downconverter. Consider a DUT being used as an upconverter. The input frequency is 70 MHz, the LO is 20 GHz, and the selected (+) output frequency is 20.07 GHz. If we chose (-) in the mixer setup dialog, the output frequency would be 19.93 GHz.

- **Characterize as upconverter** A very sharp cutoff filter is required to reject the undesired output of 19.93 GHz and pass the desired 20.07 GHz.

- **Characterize as downconverter** The input frequency is 20.07 GHz; the LO is 20 GHz. The sum (+) output is 40.07 GHz and the diff (-) output is 70 MHz. These are very easy to separate with a low-pass filter. The original frequencies are always used in the downconversion process, so be sure to choose a filter that will pass 70 MHz and reject 40.07 GHz.

See connection diagrams.

---

**Select DUT Connectors and Cal Kits** dialog box help

Allows you to specify the connector type of each DUT port.

The DUT port that is connected to PNA Logical Port 1.

**DUT Port 1** Specify the Mixer **Input** connector type and the Cal Kit to use.

**DUT Port 2** Specify the Mixer **Output** connector type and the Cal Kit to use.

**Mixer Out Port** (VMC and Mixer Characterization ONLY) Output port of the image filter that is connected to the calibration mixer. Specify the Cal Kit / standards to use for the measurement of the calibration mixer / filter combination.

**View / Modify Source Cal Settings** (SMC ONLY) These settings allow you change ALL SMC Source Cal and Power Meter settings. Click to invoke the Source Cal Settings dialog. See how to configure two power sensors.

**Note:** If your DUT connectors are:

- **Waveguide** Change the system impedance to 1 ohm before performing a calibration. See Setting System Impedance.
- **Not listed** (male and female) Select **Type A** as the connector type. Type A requires a calibration kit file containing the electrical properties of the standards used for calibration (see [Calibration kits](#)).

- **Unspecified** (like a packaged device) Select **Type B** as the connector type. Type B requires a calibration kit file containing the electrical properties of the standards used for calibration (see [Calibration kits](#)).

**Modify Cal** Check to invoke the **Modify Cal** dialog. If performing a Mixer Characterization Cal at the same time as VMC Cal, two Modify Cal dialogs will be presented, one after the other.

---

**Source Calibration Settings** dialog box help

**SMC ONLY** Allows you to modify the settings that are used during the **Source Calibration** portion of an SMC cal. These settings allow you to specify the accuracy of the Input power to the device.

**Note:** Be sure that the frequency range of your power sensor covers the frequency range of your measurement. This does NOT occur automatically.

**Power**

- **Power Offset** Allows you to specify a gain or loss (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier in the path to your DUT. Offset power is added to, or subtracted from, the power level that is set in the **mixer configuration dialog box**.

  For information about how and when to use this setting, see [SMC with a Booster Amp](#).

**Accuracy**

At each data point, power is measured using the specified **Power Meter Settling Tolerance** and adjusted, until the reading is within this Accuracy **Tolerance** or the **Max Number of Readings** has been met. The **last** power reading is plotted on the screen against the Tolerance limit lines.

- **Tolerance** Sets the maximum desired deviation from the specified **Cal Power** level.

- **Max Number of Readings** Sets the maximum number of readings to take at each data point for iterating the source power.
Use Reference Receiver for Fast Iteration

When checked, the first reading at each data point is used to calibrate the reference receiver. Subsequent readings, if necessary to meet your accuracy requirement, are measured using the reference receiver. This technique is much faster than using the power meter with almost no degradation in accuracy.

**NOTE:** Do NOT use the Reference Receiver for Fast Iteration feature if there is a component before the power sensor that exhibits non-linear behavior, such as a power amplifier in compression.

**Power Meter Config** Invokes the Power Meter Settings dialog box. See [how to configure two power sensors](#).

**OK** Applies settings and closes dialog.

**Cancel** Cancels changes and closes dialog.

---

Modify Frequency Cal dialog box help

For SMC and VMC calibrations - NOT for Mixer Characterization.

**Thru Calibration Options**

**Thru Cal Method** For each Thru connection, choose the Thru method. [Learn more about these choices](#).

**Mod Stds** Click to invoke the Modify Calibration Selections dialog box.

The following selections are available ONLY if using an ECal module.

**Do orientation** When this box is checked (default) the PNA senses the ECal model and direction in which the ECal module port is connected to the PNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually.

Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range. If you have an **E8361A** or **E836xB** PNA and do an ECal completely within 10 - 20 MHz OR 60 - 67 GHz, you may need to do orientation manually. There may not be sufficient power to orient the ECal module at those frequencies.

**View/Detect ECal Characterizations** Appears only if an ECal module is selected for use. Click to invoke the View ECal Modules and Characterizations dialog box. Displays a list of ECal modules that are connected to the PNA.
Specify how the ECal module is connected dialog box help

This dialog box appears when the Do orientation checkbox in the previous Modify Frequency dialog box is cleared.

Click the ECal Port that is connected to each PNA port.

Modify Mixer Cal dialog box help

Mixer Characterization ONLY. The Thru standard is not measured. Therefore, the Thru Cal Method choices are not available.

View / Detect ECal Characterizations Available ONLY if using an ECal module. Invokes the Select ECal Module and Characterization dialog box.

Select the ECal Port to be connected to the Output of the Calibration Mixer dialog box help

Select the ECal Port to be connected to the output of the image filter of the Calibration Mixer / Filter combination. See connection diagram of Calibration Mixer / Filter combination.
Vector Mixer Cal dialog box help

VMC and Mixer Characterization

Connect the Open, Short, and Load standards to the image filter output, then click Measure.'

This portion of the calibration characterizes the calibration mixer.

The connection is different depending on if the calibration mixer is an upconverter being characterized as a down converter.

**Note:**

The following are simplified connection diagrams - the reference mixer and LO signals must also be connected.

As a **Downconverter.** (The PNA automatically switches to make the S22 measurement on the device.)

As an **Upconverter**

**Done** Click to proceed to the Calibration Complete dialog. Available only after all measurements for the calibration are complete.
Scalar Mixer Calibration - Power Cal dialog box help.

SMC ONLY Perform the power-meter portion of the calibration.

Connect your power sensor to port 1 as shown in the diagram. Then click **Measure**.

**Measure** Begins the power meter measurements and then continues to the next step.

**Done** Click to proceed to the **Calibration Complete** dialog. Available only after all measurements for the calibration are complete.

**Abort Sweep** Stops the power meter measurement.

**Back** Returns to the previous dialog box.

**Next** Continues to the next calibration step. Does NOT make a measurement.

**Notes**

- Beginning with Rev 6.0, a power meter measurement is only necessary on port 1.

- SMC calibration performs 10 averages at the beginning and at the end of the power cal step to ratio the difference between normal and offset R1 measurements in the calibration band of frequencies. The averaging is done to remove a reasonable amount of noise from the ratio measurement.

- From **Source Calibration dialog** you can use the **Power Loss Compensation Table** to compensate for an adapter used to connect the power meter sensor.

- See **how to configure two power sensors**.
**Measure Calibration Standards** dialog box help

Prompts for standards to be measured. Connect the standard, then click **Measure**.

**Measure** Measures the mechanical standard and continue to the next calibration step.

**[ReMeasure]** Replaces Measure after standard has been measured. Allows you to remeasure a standard.

**Done** Click to proceed to the **Calibration Complete** dialog. Available only after all measurements for the calibration are complete.

**Back** Returns to the previous dialog box.

**Next** Does NOT make a measurement. Proceeds to the next required step.

**Cancel** Exits the Calibration Wizard.

---

**Save Mixer Characterization** dialog box help

**VMC ONLY**

Allows you to save the characterization data of your calibration mixer. When performing another VMC calibration using the same calibration mixer, this S2P file can then be recalled.

**Browse** Navigate to the location where you want to save the characterization data of your calibration mixer. Either use the default file name or enter a custom file name.

**Next** Saves the mixer characterization file and continues with the next step in the full system calibration routine.

**Finish** Replaces Next if you are only characterizing the calibration mixer instead of performing a full system calibration. Saves the mixer characterization file and exits the mixer characterization routine.
Create and Apply an FCA Cal Set or SMC Cal Type

You can create an FCA measurement and apply an existing Cal Set as you can with any PNA measurement. Learn about Cal Sets. In addition, from a Cal Set, you can apply a specific SMC Cal Type to an existing SMC measurement.

Although the Cal Type selection is available for VMC, there is only one VMC Cal Type.

How to apply an SMC Cal Type

1. Create an SMC measurement
2. Calibrate or apply an existing SMC Cal Set, then...

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>For N5230A and E836xA/B models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Calibration</td>
</tr>
<tr>
<td>2. then Manage Cals</td>
<td>2. then Cal Type</td>
</tr>
<tr>
<td>For PNA-X and 'C' models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press Cal</td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then [Manage Cals]</td>
<td>2. then Calibration</td>
</tr>
<tr>
<td>3. then [Cal Type]</td>
<td>3. then Manage Cals</td>
</tr>
<tr>
<td></td>
<td>4. then Cal Type</td>
</tr>
</tbody>
</table>
Each SMC measurement requires FOUR sweeps. Three of these are hidden. If the input and output of your mixer is well-matched to the PNA, you can apply SMCRsp Cal Type to speed up your SMC measurements. This is most noticeable when making fixed input or fixed output measurements, which requires an external LO to sweep with the PNA.

**SMC_2P**: (Response + Input + Output) All four sweeps required. Most accurate.

**SMCRsp**: No Input or Output match. Saves two sweeps.

**SMCRsp+In**: No Output match. All four sweeps required.

**SMCRsp+Out**: No Input match. All four sweeps required.
Configure a Mixer

How to Start the Mixer Setup dialog box

Learning the Mixer Setup Dialog Box

Rules for Configuring a Mixer

Using Power Sweep for Testing Mixers

Input > LO Example

Configure Swept LO Measurements

Fractional Multiplier Examples

See Also

- How to make a VMC Measurement example
- How to make an SMC Measurement example
- Measure a DUT with an Embedded LO

Note: Please submit FCA issues that you find, as well as enhancement requests, to fca_support@agilent.com (See Known FCA Issues.)

Other Frequency Converter Application topics

How to start the Mixer Setup dialog box

1. Create an FCA measurement. Then...

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. <strong>Sweep Type</strong></td>
<td>1. Click Trace</td>
</tr>
<tr>
<td></td>
<td>2. then Measure</td>
</tr>
<tr>
<td></td>
<td>3. then Configure Mixer</td>
</tr>
</tbody>
</table>
For PNA-X and 'C' models

1. Press **FREQ**
2. then **[Input, LO, or Output]**

**Programming Commands**

1. Click **Response**
2. then **Measure**
3. then **Configure Mixer**

### Learning the Mixer Setup dialog box

Click on sections of the image to learn about a setting.

**Note:** This image shows two LOs.

**Important Note:** Connecting your DUT to the PNA using FCA:

**RF** and **IF** terminology is NOT used in the FCA because the PNA does not know how the DUT is labeled or how it will be used. Instead, the general terms **INPUT** and **OUTPUT** are used to describe the following PNA behavior:

- **INPUT** - the stimulus frequencies, BEFORE conversion by your DUT.
- **OUTPUT** - the response frequencies, AFTER conversion (either UP or DOWN) by your DUT. Specify UP or DOWN conversion using the + or - symbol for each output.

See [Fractional Multiplier Examples](#) (below)
Throughout the dialog box, the Mixer / converter ports are color coded (Input, LO1, IF1, LO2, Output)

**Rules for Configuring a Mixer**

Red **Apply** and **OK** buttons indicate that one or more of the following settings are invalid.

1. The INPUT start frequency can NOT exceed the stop frequency. (The OUTPUT start frequency CAN exceed the stop frequency.)
2. INPUT or OUTPUT frequencies cannot be outside the range of the PNA.
3. Any combination of INPUT and LO which results in an OUTPUT that sweeps through zero Hz is NOT allowed.
4. The range for the numerator and denominator of a fractional multiplier is from +1 to +10. Negative values are NOT allowed.

**Power** Sets the power level of the input signal, and both LO signals.

**Frequency Format** Selects the format to specify the frequency information for each signal in your test setup. The Input, LO1, LO2, IF, and Output frequency information can be specified using start/stop or center/span formats. Only LO1, LO2, IF, and Output formats can be set to Fixed. When you select a swept LO, you can also select the information you want to display on the X-axis.

**LO1 and LO2 Source Configuration Buttons** Performs the same function as the configuration buttons on the lower diagram. The current source, or **Not Controlled** is displayed on the button label. Click to launch the **Select Source** dialog.

**Resulting Frequencies** Either sets or calculates the frequency values for each of the signals in your test setup. For example, if you enter the Input frequency range and press the Calculate button adjacent to the Input, the PNA will calculate and display the Output frequencies.

Go to the **Mixer Setup** image

**Input > LO**
These check boxes remove ambiguity when using the Calculate button to determine the INPUT frequency.

<table>
<thead>
<tr>
<th>Checkbox</th>
<th>Description</th>
</tr>
</thead>
</table>
| ![Input > LO](image) | Check if the **INPUT** is **GREATER THAN** the **LO**
Clear if the **INPUT** is **LESS THAN** the **LO** |
| ![IF1 > LO2](image) | Check if the **IF1** is **GREATER THAN** the **LO2**
Clear if the **IF1** is **LESS THAN** the **LO2** |

These boxes are only used when all 3 of the following conditions are TRUE: (If ALL 3 are NOT true, the PNA does not read these boxes).

1. Difference (Low) sideband is selected for the corresponding Calculate button AND
2. Output frequency is less than the LO frequency AND
3. The Green or Blue Calculate button is used to calculate the Input frequency.

To learn more, see this example.

**Fractional Multiplier**

The combination of (numerator / denominator) forms a fractional value that is multiplied by the input and LO frequency ranges (also the IF and LO2 frequency ranges for a test setup with two LOs). These values are used to calculate the response frequency of the PNA receiver. Use the fractional multipliers to:

- replicate the action of harmonic mixers
- replicate the action of multipliers and dividers that may exist in your test setup
- tune the PNA receiver frequency to a harmonic of the mixer/ converter

See Fractional Multiplier examples. Go to the Mixer Setup image.

**Mixer-Product Selector** Determines whether the receivers will tune to the Sum (+) or the Difference ( - ) of the Input and LO frequencies. Click the adjacent Calculate button after your selection.

**Calculate buttons** Calculates frequency information based on your other mixer settings. The mixer port settings next to the Calculate button you press remain fixed. For example, in a 1-LO scenario, specify the Input and LO frequencies, specify + (sum), then click the Calculate button next to the Input. The input remains fixed and the output frequency range is calculated for you.

**Hide / Show Diagrams** Hides and displays the test setup diagram. Your measurement trace is displayed when the diagram is hidden.

**LOs** Click 1 or 2 to select the number of LO sources in your test setup. When you select 2 LOs, the IF1 frequencies are set for you. You can also measure devices with an Embedded LO.

**Avoid Spurs** Check to invoke the Avoid Spurs feature.

**Load** Loads a previously-configured mixer attributes file (.mxr).

**Note:** A .mxr file includes an LO source name. However, it does NOT include the LO Source configuration. Therefore, when using a .mxr file that was created on a different PNA, the PNA will display an error if does not find the LO Source configuration using EXACTLY the same LO source name.

**Save** Saves the settings for your mixer/converter test setup to a mixer attributes file (.mxr).

**Apply** Applies the settings for your mixer/converter test setup to the measurement. The mixer setup dialog box remains OPEN. If shaded red, see rules.

**OK** Applies the settings for your mixer/converter test setup to the measurement. The mixer setup dialog box CLOSES. If shaded red, see rules.

**Cancel** Closes the mixer setup dialog box and does NOT apply the settings.

**Frequency Diagram:** Provides a display of the frequency information for the signals in the test setup. Go to the Mixer Setup image.
This dialog is launched when clicking the Mixer Setup LO1 or LO2 button. Click one of the following to select a source for LO control:

- An existing External Source setup. Calibrate the source using a standard Source Power Cal.
- A port number to use an internal second source
- Not controlled, the PNA will not attempt communication to control a source for the LO.
- Manage External Sources button to launch the External Source Configuration dialog.

Note: VMC measurements using a PNA-X with Internal Second Source
Source 2 is automatically configured to supply power to BOTH available ports simultaneously. This setting can NOT be changed.

In addition, power can be uncoupled to provide different power levels at each port. This feature allows power to be delivered to both the DUT LO and Reference Mixer LO without use of a splitter. See VMC setup.

**Using Power Sweep for Testing Mixers**

To measure the gain compression of a mixer, you need to sweep the input power to the mixer. The input and output frequencies are fixed but offset from one another. To set Power Sweep and the input and output frequencies of the mixer under test:

1. On the mixer dialog box, set the LO frequency, identical input start and stop frequencies, and identical output start and stop frequencies. These selections create fixed input and output frequencies.

2. On the PNA menu, click Sweep, then Sweep Type. Select Power Sweep. Do NOT change the CW
frequency on the Power Sweep dialog box. The mixer dialog box settings will not be automatically updated.

For more information, see Conversion Compression.

**Input > LO Example**

For the following single stage mixer:

- Output = 2 GHz
- LO = 3 GHz
- Diff (-) selected

Clicking **Calculate Input** could yield two Input frequencies:

Formula for **Diff**:

\[ \text{Input} - \text{LO} = \text{Output} \]

Substitute our example values in the formula:

\[ \text{Input} - 3\text{GHz} = 2\text{GHz} \]

Solving the formula can yield either:

- **Input = 5 GHz**
- **Input = 1 GHz**

(Although 1-3 = -2 GHz, the analyzer displays the absolute value of the frequency.)

**Check** - use the Input frequency (5 GHz) that is greater than LO (3 GHz)

**Clear** - use the Input frequency (1 GHz) that is less than LO (3 GHz)

**Configure Swept LO Measurements**

**Note:** With a corrected VC21Swept LO measurement, the phase data is displayed relative to the phase of the calibration mixer that was used during the VMC calibration. In addition, Group delay display format is NOT valid.
See Examples of Fixed Output Measurements

- SMC
- VMC

**Fractional Multiplier Examples**

**Example 1**

Use the LO fractional multiplier to replicate the action of the third-harmonic mixer so the PNA can accurately calculate the receiver frequency. The input and LO frequencies are known.

Enter these settings in the **Mixer Setup** dialog box:

- **Input Start Freq**: 30 GHz
- **Input Stop Freq**: 40 GHz
- **LO Fixed Freq**: 16 GHz
- **Mixer-Product Selector**: - (difference)
- **LOs**: 1
- **LO fractional multiplier**: 3/1
- **INPUT fractional multiplier**: 1/1

Click **Calculate Output**

Results:

- **Output Start Freq**: 18 GHz
- **Output Stop Freq**: 8 GHz

**Example 2**

Use the fractional multipliers to tune the PNA receiver frequency to the second harmonic of the mixer's 14 GHz
fundamental output. The input, LO, and output frequencies are known.

![Diagram](image)

Enter these settings in the **Mixer Setup** dialog box:

- **Input Start Freq**: 4 GHz
- **Input Stop Freq**: 4 GHz
- **LO Fixed Freq**: 10 GHz
- **Mixer-Product Selector**: + (Sum) of the input and LO signals
- **LOs**: 1
- **INPUT fractional multiplier** = 2/1
- **LO fractional multiplier** = 2/1

Click **Calculate Output**

Results:

- **Output Start Freq**: 28 GHz
- **Output Stop Freq**: 28 GHz

**Example 3**

Use the LO fractional multiplier to replicate the action of the divide-by-two mechanism inside the mixer package. Having done this, the PNA can accurately calculate the receiver frequency. The input and LO frequencies are known.
Enter these settings in the **Mixer Setup** dialog box:

- **Input Start Freq**: 45 MHz
- **Input Stop Freq**: 50 MHz
- **LO Fixed Freq**: 670 MHz
- **Mixer-Product Selector**: + \((\text{Sum})\) of the input and LO signals
- **LOs**: 1
- **INPUT fractional multiplier** = 1/1
- **LO fractional multiplier** = 1/2

Click **Calculate Output**

**Results:**

- **Output Start Freq**: 380 MHz
- **Output Stop Freq**: 385 MHz

---

**Last Modified:**

20-Apr-2007  MX Updated for internal second source
Configure an External Source

Beginning with PNA Rev. 7.22, an external source can be configured and controlled by the PNA for all FOM (opt 080), FCA (opt 083), or Millimeter Wave (opt H11) measurements. Without one of these options you must control an external source manually.

Also, the External Source Control feature only supports List-sweep mode, which a PSG limits to 1601 points. Manual source control supports Step-sweep mode, in which a PSG allows up to 65,535 points.

See Synchronize an External Source for help with manual source control.

After a one-time Configuration of an External Source, it must be selected for each measurement using the Select Sources dialog.

In this topic:

- How to Select an External Source
- How to Connect an External Source to the PNA
- How to Configure an External Source
- Add New Source

See Also the following examples:

- How to make a VMC Measurement
- How to make an SMC Measurement

Beginning with PNA Rev 7.5...

- Generic (Non-Agilent) sources are not supported with this release.
- With an external source selected, the PNA WILL now allow an S-parameter measurement to be made at the same time as an FCA measurement.
- All External Sources, including those used for FCA LO, are now calibrated by doing a standard Source Power Calibration.
- FCA external sources are now selected and managed using the standard dialogs shown in this topic.
How to Select an External Source

For an FCA measurement

1. In the Configure Mixer dialog box, click the LO 'Control' button
2. In the Select Source dialog, click Select External Sources to launch the Select Sources dialog.

For all other FOM (opt 080) measurements, do the following:

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td>See Remotely Specifying a Source Port</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click System</td>
</tr>
<tr>
<td></td>
<td>2. then Configure</td>
</tr>
<tr>
<td></td>
<td>3. then Select External Source</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td>See Remotely Specifying a Source Port</td>
</tr>
<tr>
<td>1. Press SYSTEM</td>
<td>1. Click Utility</td>
</tr>
<tr>
<td>2. then [Configure]</td>
<td>2. then System</td>
</tr>
<tr>
<td>3. then [Select External Source]</td>
<td>3. then Configure</td>
</tr>
<tr>
<td></td>
<td>4. then Select External Source</td>
</tr>
</tbody>
</table>
Select Sources dialog box help

See What's new with External Source Control with PNA Rev. 7.5.

For FOM (Opt 080) measurements, once you select an external source from this dialog box, it becomes available from the following PNA dialog boxes as if it were an internal PNA source:

- FOM (turn ON and set freq, power)
- Power (set power level)
- Source Power Cal (Calibrate the external source)
- New / Change Trace (set source port for a measurement)

All Sources The sources that are currently configured appear in this list. There is no limit to the number of sources that can be selected.

- Click Add All>> to move all sources to the Selected list.
- Click a source name, then click Add> to add that source to the Selected list.

Configure Click to launch the Configure External Source dialog to add a source to the list.

Selected The sources that are currently selected.

- Click <Remove All to un-select all sources.
- Click a source name, then click <Remove to un-select that source.

Move Up and Down Changes the order of the sources in the list. The order indicates the order that the sources appear in the FOM, Power, Source Power Cal, and New / Change Trace dialog boxes.

Important Notes

- All newly selected or reordered sources are preset, with source power OFF. Source power must be turned ON in the Power dialog. Frequency Offset must be enabled in the FOM dialog.
- When reordering a list of sources using Move Up / Down, any existing PNA frequency and power settings for those sources will be preset and must be reentered.
- When daisy-chaining multiple sources in Hardware List triggering, the source to receive the Trigger signal from the PNA must be the first source listed in the Selected column of this dialog.
- The PNA controls the triggering of an external source. Therefore, PNA triggering must be set to Internal, not External.
- Your source selections remain until you recall an instrument state with different selections, or perform a factory preset.
- Communication with the selected sources is checked when OK is pressed. Make sure that the source is turned ON and the GPIB address in the configure dialog is accurate.
If communication with a source is broken after the dialog box is closed, a message appears and channel triggering is put in Hold mode.
The same source can NOT be used more than once in the same channel.

How to Connect an External Source to the PNA

1. GPIB or LAN, use one of the following methods:
   - The Agilent 82357A USB/GPIB Interface.
   - Dedicated Controller and Talker/Listener GPIB ports.
   - **USB** or **LAN using Visa Alias**. Both of these interfaces are configured using Agilent ACE (IO libraries) which is installed on the PNA.
     1. In ACE, click **Add Instrument**.
     2. Select **LAN (TCPIP0)** or **USB0**, then click **OK**.
     3. Click, then enter the **IP address** of the external source.
     4. Click **Test Connection** to verify communication.
     5. Click **OK**.
     6. In the list of connected instruments, right click the external source, then **Add VISA Alias**.
     7. Enter the same PNA source name that was, or will be, used in the **Add New Source** dialog.
   - The standard GPIB Interface (One GPIB port) - with the following limitations:
     - The PNA cannot be controlled remotely as talker / listener over GPIB. First put the PNA in System Controller mode. Learn how.
     - If this method does not work initially, first close, then restart the PNA application, then put the PNA in System Controller mode, then click **Controlled** on this dialog box. This should resolve any GPIB hang-up issues with the external source.

2. External sources should always share the same 10 MHz Reference signal as the PNA. Connect a BNC cable from the PNA 10 MHz Ref Output to the External Source Input.

3. See Hardware List Triggering Connections
How to Configure an External Source

For an FCA measurement:

- In the Configure Mixer dialog box, click LO Control OR
- In the Select Source dialog, click Manage External Sources

For all other measurements:

- Click Configure from the Select Sources dialog OR
- As follows:

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td>See Remotely Specifying a Source Port</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click System</td>
</tr>
<tr>
<td></td>
<td>2. then Configure</td>
</tr>
<tr>
<td></td>
<td>3. then External Source Config</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td>See Remotely Specifying a Source Port</td>
</tr>
<tr>
<td>1. Press SYSTEM</td>
<td>1. Click Trace/Chan</td>
</tr>
<tr>
<td>2. then [Configure]</td>
<td>2. then Channel</td>
</tr>
<tr>
<td>3. then [External Source Config]</td>
<td>3. then Hardware Setup</td>
</tr>
<tr>
<td></td>
<td>4. then External Source Config</td>
</tr>
</tbody>
</table>

**Note:** If an External Source Not Found error occurs, the Agilent I/O Library may no longer be running. To check, look in the Windows task bar of the PNA for the I0 icon. If not present, restart the IO library. Click Start, Programs, Agilent I/O Libraries, IO Control.
External Source Configuration dialog box help

This dialog box is used to perform a one-time configuration of an external source.

All External Sources, including those used for an FCA LO, are calibrated by doing a standard Source Power Calibration.

Available Sources

Lists the sources that have been previously configured.

- **Add** Displays the Add New Source dialog box. Type a *unique* source name.
- **Modify** Makes changes to the selected source. Launches Modify Source dialog, exactly like Add New Source.
- **Remove** Removes an external source from your setup.

Trigger Mode  ONLY used when the external source is stepped, as with FCA swept LO measurements.

Notes

- **MM Wave Test Heads**: The PNA trigger settings are automatically configured and must not be changed.

  - The PNA automatically controls triggering of the external source. PNA triggering can be set to Internal, Manual, or External. When set to External, the trigger signal must come through a PNA rear panel connector that is not being used to trigger the external source.

  - See SCPI and COM examples of an SMC fixed output measurement.

- **Software CW (GPIB)**  Slowest method.
- The external source receives the CW frequency and trigger signal from the PNA over GPIB.
- Used with ALL sources, including generic (not listed), and Agilent 837X sources.

**Hardware List (BNC) Fastest method.**

**Note:** If the number of data points used in the measurement exceeds the capability of the external source, the PNA automatically switches to Software CW (GPIB) trigger mode. This will slow the measurement significantly.

**Trigger** Select the PNA rear panel connector to be used for triggering.

- The external source receives a list of CW frequencies from the PNA, then receives BNC trigger signals as required from the PNA.
- Used with ALL except generic (not listed) sources.
- **MM Wave Test Heads:** Hardware List mode is ALWAYS used.
- The sources must be connected as follows:

- **PNA-X models:** Connect multiple sources using the following daisy-chain, or directly using Aux1 or Aux2. See rear panel Aux connectors
- **E836x and PNA-L models** - Use rear-panel BNC Trigger connectors as follows:

  ![1 External Source](image1)

  ![Daisy-chain 2 External Sources](image2)

**Note:** Source 1, which receives the trigger out of the PNA, must be listed first on the Select Sources dialog box.
Shows the model number of the external source that is selected in the displayed list.

**Edit Commands** Only available with generic (non-Agilent) sources. Not available in the A.07.50 release.

**GPIB Address** Sets the GPIB address of the selected external source.

**Timeout (sec)** Sets a time limit for the source to make contact with the PNA. If this time limit is exceeded, the PNA stops the measurement procedure and displays a diagnostic-type error message. If this occurs, check the connections of your PNA and source.

Use the standard Source Power Calibration to calibrate the external source.

---

**Add New Source**

![Add New Source dialog box](image)

**Add / Modify New Source** dialog box help

Allows you to add or modify an external source. The new or modified source appears in the list of sources displayed in the External Source Configuration dialog box.

**Source Name** Enter a unique name for your source.

**Note:** If you enter a source name that existed since that last PNA Shutdown, the old Source Type will be remembered and displayed on the External Source dialog. Either use a new name, or delete the old name, then restart the PNA application before re-entering the name.

**Source Type** Select a source type from the scrolling list.

**Generic** (Non-Agilent) sources are not supported.

---

Last Modified:

- 11-Feb-2008  Added limitation note at top
- 23-Jan-2008  Added Selected ordering notes
- 5-Nov-2007  Added links for remote selection
- 18-Jul-2007  Edited for FCA LO Cal changes
- 30-Apr-2007  MX Modified for ALL external source config.
How to make a VMC Fixed Output Measurement

The following is a step-by-step example illustrating how to measure a mixer in swept LO mode using FCA Vector Mixer Calibration.

There are fewer components required for SMC as compared to VMC, and fewer measurement steps. Therefore, if you do NOT need to make relative phase measurements, SMC is an easier measurement. Also, ONLY SMC (not VMC) can measure the reverse conversion loss of the mixer.

This procedure can also be used for making fixed LO measurements, which is quite similar. Although the external source is still required, the physical triggering cables that connect the PNA and External Source are not required.

Required Equipment

- N5242A (PNA-X), E8362B, E8363B, E8364B or E8361A PNA series network analyzer
  - with option 083 (FCA)
  - with PNA Rev 6.03 or greater
- GPIB External Source (Agilent ESG or PSG works best) **
- Reference Mixer (see requirements)
- Calibration Mixer/Filter (see requirements)
- Power splitter **
- ECAl module with connectors that match the Input and Output connectors of the DUT. You can use adapters to make the ECAl module match the DUT connectors, but first perform an ECAl user-characterization with the adapters attached. ECAl makes the FCA calibration much easier.
- Cables and adapters
- **Optional** GPIB Power meter and sensor (for LO power calibration)

** Not necessary when using PNA-X with Internal Second source

The example mixer

The example device is a mixer with the following characteristics:

- LO and Input Frequency Range: 2 GHz to 4.2 GHz
- Output Frequency Range: DC to 1.3 GHz

We will measure:

- Fwd Conversion Loss (VC21)
- Input match (S11)
- Output match (S22)
- Rev Conversion Loss is NOT possible because of the reference mixer.

**VMC Setup**

Connect the devices as shown in the following diagram:

**Note:** This setup can also be used for SMC measurements, allowing you to make VMC and SMC measurements simultaneously on separate channels. The Reference Mixer is automatically switched during SMC measurements. The Cal Mixer/Filter is not used.

---

**Notes:**

- When using a PNA-X with Internal Second Source, the external source is NOT necessary.
  - See note regarding LO power out both second source ports
  - Learn which PNA ports can be used for the LO.
- The low-pass filter on the output of the Reference Mixer is recommended, but NOT required. Learn more.
Make Connections on the Instrument rear panels:

1. Connect the PNA and Source using two GPIB cables. A USB to GPIB adapter can also be used if you need to control the PNA from a remote PC.

2. Using a BNC cable, connect the Source 10 MHz Reference Output to the PNA 10 MHz Reference Input.

3. Using two BNC cables, connect the Source and PNA Trigger connectors as shown in the following image. This is not necessary when making fixed LO measurements.

![PNA and Source Connections Diagram]

Create the Measurement

For this document:

- Front-panel hardkeys are formatted as "Press Trace"
- Menus are formatted as "Click System"

1. Connect the DUT.

2. On the PNA, click System, then point to Configure, then click SICL/GPIB/SICL. On the SICL/GPIB/SICL dialog, click System Controller. This allows the PNA to control the Source and Power Meter.

3. On the Source, note the GPIB address.

4. Press Preset to make sure you are starting with a known state.

5. Press Trace, then Delete to delete the default trace.

6. Press Application to create a new FCA measurement.

7. Under Choose an application, select Vector Mixer/Converter.


9. Click OK.

Configure the Mixer settings
1. Press **Measure Setups**, then **Mixer**

2. Enter the Mixer setup values as shown in the image below.

**Notes:**

- Rather than enter ALL of the frequency settings, you can enter the Input and the Output frequencies, then click **Calculate LO**.

- If **Input>LO** is NOT checked, the PNA assumes you want the Input < LO frequencies, and higher LO frequencies are calculated as a result.

- The <Controlled> LO power level setting specifies the power out of the external source (not at the DUT) unless an LO power cal is performed.

- The Avoid Spurs feature is useful for eliminating spurs in test setups with excessive LO leakage.

- When the settings are valid, the background color around the **Apply** and **OK** buttons changes from Red to Green.

![Mixer Setup dialog](image)

**Configure the External LO Source**

When using a PNA-X with Internal Second Source, the external source is NOT necessary.

- **See note regarding LO power out both second source ports**

- **Learn which PNA ports can be used for the LO.**

1. Click **Not Controlled** to set up the External LO source. The following dialog appears:
2. Select a configured source from the **All Sources** column and click **Add**. If no sources appear, then click **Configure**. The following dialog appears:

Depending on the model of your source, this is what it looks like AFTER entering the settings.

3. Click **Add**, to add a source.
4. In the Add New Source dialog, type an identifying Source Name, such as the model of your source. In Source Type, select the model of source you are using. Then click **OK**.

5. Back in the External Source Configuration dialog, click **Controlled**, to tell the PNA to assume control of the source.

6. Click **Hardware List (BNC)**, which is the fastest measurement method. This method requires the BNC Trigger cables that connect the PNA and source. If not available, **Software CW** can be used, but measurements are much slower.

7. If necessary, change the GPIB Address to match that of the source. This is NOT automatically detected.

8. Click **OK** to return to the Mixer Setup dialog. The Not Controlled should now read **Controlled**.

9. Save the mixer settings in a file so you can recall them easily. Click **Save...**, then type a descriptive filename, such as “FixedOutputMixer”.

10. Click **OK** to close the Mixer Setup dialog. If there is a problem communicating with the source, the PNA will display an error here. **See Problems?**

11. The two traces should begin to sweep, as the external source steps in frequency. It should look something like this:

   ![Graph showing two traces sweeping](image)

Because of the reference mixer, the uncorrected VMC measurement can look like it has gain.
Problems?

Not sweeping:

- On the PNA, press **Sweep**, then **Trigger**, then **Continuous** to start the PNA sweeping. Watch for error messages on the PNA and source.

Problems communicating with the source:

- Press **Measure Setup**, then **Mixer** to start the Mixer setup dialog. Click **Software CW trigger**, then close the dialog. Perform the previous statement to start sweeping. If this works, then something is wrong with **Hardware (BNC)**. Check the trigger cables on the rear panel.

- As a last resort, try rebooting the PNA. First, **save the entire setup to a .csa file**. When the PNA preset measurement appears, recall the .csa file to resume at this step.

If the source is sweeping, and the PNA Input is sweeping, but there is still no output.

- Check power levels at the LO and Input.

- Check the DUT by making a fixed LO measurement - much easier.

Perform a VMC Calibration

**Note:** Optionally perform a **Source Power Cal** before the VMC Cal to specify the LO Power at the DUT. This requires a power meter be connected to the GPIB.

1. Disconnect the DUT.

2. Connect the ECal module to a PNA USB port.

3. Click **Calibration**, then **Calibration Wizard**. Because the VC21 measurement is active, the Cal Wizard automatically begins a VMC Calibration.

4. At the **Calibration Setup** dialog, click **Next**.

5. At the **Calibration Mixer Characterization** dialog, click **Next**. We will perform characterization of the Calibration mixer as part of the VMC cal. Later we will save the Calibration mixer characterization so that, in future VMC calibrations that use this same frequency range, we can recall the Calibration mixer characterization by clicking **Load Characterization from file**.

6. At the **Select DUT Connectors and Cal Kits** dialog, for **DUT Port 1** select the connector type and gender of your DUT INPUT. For **DUT Port 2** select the connector type and gender of your DUT OUTPUT. Then select ECal as the Cal Kit to use for each connector. Click **Next**.

7. At the **Select the ECal Port to be Connected** dialog, ensure that **Port A** is selected for **Port 1**, then click **Next**.

8. At the **Vector Mixer Calibration Step 1 of 3** dialog, connect the ECal module Port A to the Port 1 cable, and
Port B to the Port 2 cable. Then click **Measure**. This portion of the calibration gathers the linear (non-frequency-translating) error terms of the test setup at the input and output frequencies.

9. At the **Vector Mixer Calibration Step 2 of 3** dialog, connect the following, then click **Measure**. This portion of the calibration will connect reflection standards to characterize the S-parameters of the calibration mixer/filter.

- Port 1 cable to the Input of the calibration mixer.
- LO cable to the LO port of the calibration mixer.
- ECal module to the Output of the calibration mixer/filter.

10. At the **Vector Mixer Calibration Step 3 of 3** dialog, disconnect the ECal module and connect the Port 2 cable to the output of the calibration mixer/filter, then click **Measure**. This step completes the calibration using the characterized mixer/filter as a Thru standard.

11. At the **Save Mixer Characterization** dialog, click **Browse**, then type a unique filename and click **OK**. Then click **Next**. This saves the Calibration Mixer characterization to an S2P file. This file can be recalled for subsequent VMC calibrations.

12. At the **Calibration completed** dialog, you can choose to save the VMC calibration as a User Cal Set. Otherwise, click **Finish** to complete the VMC calibration. Correction is turned ON and applied to the VMC trace that we set up earlier.

**What is happening?**

Because Fixed Output or Fixed Input FCA measurements require an external source to sweep, the measurements are much slower. When correction is ON, you will see that there are times when nothing is happening on the screen. This is because there are background measurements being made but not displayed.

This is exactly the same as when full 2-port correction is applied to an S-parameter. All four parameters are measured, then correction is applied, then all four measurements are updated. This occurs much faster when there is no external source. With a VMC measurement, there is no VC12 (reverse transmission measurement), so there are only three background measurements. With correction OFF, the traces are updated as the data is measured. You can see this taking place by creating the following measurements.

**Create S11 Input and S22 Output Match**

1. Press **Trace**, then **Application**. Click **S11** and **S22**, then click **OK** to add these measurements to the same channel.

2. While the source is sweeping, watch the source port indicator on the front of the PNA. First, the port 1 indicator will light for two sweeps, then the port 2 indicator will light for 1 sweep while all 3 traces update.

3. Turn correction OFF for ALL measurements. Notice that the relevant traces will update as the sweep is occurring.

The following image shows the corrected Conversion Loss (VC21), Input Match (S11), Output Match (S22) and the uncorrected Conversion Loss (VC21), which is a memory trace.
How to make an SMC Fixed Output Measurement

The following is a step-by-step example illustrating how to measure a mixer in swept LO mode using FCA Scalar Mixer Calibration.

There are fewer components required for SMC as compared to VMC, and fewer measurement steps. Therefore, if you don't need to make relative phase measurements, SMC is an easier measurement. Also, ONLY SMC (not VMC) can measure the reverse conversion loss of the mixer.

This procedure can also be used for making fixed LO measurements, which is quite similar. Although the external source is still required, the physical triggering cables that connect the PNA and External Source are not required.

Required Equipment

- E8362B, E8363B, E8364B or E8361A PNA series network analyzer
  - with option 083 (FCA)
  - with PNA Rev. 6.03 or greater
- GPIB External Source (Agilent ESG or PSG works best)
- ECal module with connectors that match the Input and Output connectors of the DUT. You can use adapters to make the ECal module match the DUT connectors, but first perform an ECal user-characterization with the adapters attached. ECal makes the FCA calibration much easier.
- GPIB Power meter and sensor
- Cables and adapters

The example mixer

The example device is a down-converter mixer with the following characteristics:

- LO and Input Frequency Range: 2 GHz to 4.2 GHz
- Output Frequency Range: DC to 1.3 GHz

We will measure:

- Fwd Conversion Loss (SC21)
- Input Match (S11)
- Output Match (S22)
- Reverse Conversion Loss (SC12)

SMC Setup

Connect the devices as shown in the following diagram:
Make Connections on the Instrument rear panels:

1. Connect the PNA, Source, and Power Meter using two GPIB cables. A USB to GPIB adapter can also be used if you need to control the PNA from a remote PC.

2. Using a BNC cable, connect the Source 10 MHz Reference Output to the PNA 10 MHz Reference Input.

3. Using two BNC cables, connect the Source and PNA Trigger connectors as shown in the following image. This is not necessary when making fixed LO measurements.

Create the Measurement

For this document:
Front-panel hardkeys are formatted as “Press Trace”

Menus are formatted as "Click System"

1. Connect the DUT.

2. On the PNA, click System, then point to Configure, then click SICL/GPIB/SCPI. On the SICL/GPIB/SCPI dialog, click System Controller. This allows the PNA to control the Source and Power Meter.

3. On the Source and Power Meter, record the GPIB addresses.

4. Press Preset to make sure you are starting with a known state.

5. Press Trace, then Delete to delete the default trace.

6. Press Application to create a new FCA measurement.

7. Under Choose an application, select Scalar Mixer/Converter.


9. Click OK.

Configure the Mixer settings

1. Press Measure Setups, then Mixer

2. Enter the Mixer setup values as shown in the image below.

   Notes:

   - Rather then enter ALL of the frequency settings, you can enter the Input and the Output frequencies, then click Calculate LO.

   - If Input>LO is NOT checked, the PNA assumes you want the Input < LO frequencies, and higher LO frequencies are calculated as a result.

   - The LO power level setting specifies the power out of the external source; not at the DUT) unless an LO power cal is performed.

   - The Avoid Spurs feature is useful for eliminating spurs in test setups with excessive LO leakage.

   - When the settings are valid, the background color around the Apply and OK buttons changes from Red to Green.
Configure the External LO Source

1. Click **Not Controlled** to set up the External LO source. The following dialog appears:

   Depending on the model of your source, this is what it looks like AFTER entering the settings.

2. Click **Add**, to add a source.
3. In the Add New Source dialog, type an identifying Source Name, such as the model of your source. In Source Type, select the model of source you are using. Then click **OK**.

4. Back in the External Source Configuration dialog, click **Controlled**, to tell the PNA to assume control of the source.

5. Click **Hardware List (BNC)**, which is the fastest measurement method. This method requires the BNC Trigger cables that connect the PNA and source. If not available, **Software CW** can be used, but measurements are much slower.

6. If necessary, change the GPIB Address to match that of the source. This is NOT automatically detected.

7. Optional: Click **LO Power Calibration** to calibrate the LO Power level at the DUT.

8. Click **OK** to return to the Mixer Setup dialog. The Not Controlled should now read **Controlled**.

9. Save the mixer settings in a file so you can recall them easily. Click **Save…**, then type a descriptive filename, such as “FixedOutputMixer”.

10. Click **OK** to close the Mixer Setup dialog. If there is a problem communicating with the source, the PNA will display an error here. [See Problems?](

11. The trace should begin to sweep as the external source steps in frequency. It should look something like this:
# Problems?

## Not sweeping:

- On the PNA, press **Sweep**, then **Trigger**, then **Continuous** to start the PNA sweeping. Watch for error messages on the PNA and source.

## Problems communicating with the source:

- Press **Measure Setup**, then **Mixer** to start the Mixer setup dialog. Click **Software CW trigger**, then close the dialog. Perform the previous statement to start sweeping. If this works, then something is wrong with **Hardware (BNC)**. Check the trigger cables on the rear panel.

- Can the PNA communicate with the power meter? If not, there is something wrong with the GPIB communication.

- As a last resort, try rebooting the PNA. First, save the entire setup to a .csa file. When the PNA preset measurement appears, recall this .csa file and continue at this step.

## If the source is sweeping, and the PNA Input is sweeping, but there is still no output.

- Check power levels at the LO and Input.

- Check the DUT by making a fixed LO measurement - much easier.

## Perform an SMC calibration

1. Disconnect the DUT.

2. Connect the ECal module to a PNA USB port.

3. Click **Calibration**, then **Calibration Wizard**. Because the SC21 measurement is active, the Cal Wizard automatically begins an SMC calibration.

4. At the **Calibration Setup** dialog, click **Next**.

5. At the **Select DUT Connectors and Cal Kits** dialog, for **DUT Port 1** select the connector type and gender of your DUT INPUT. For **DUT Port 2** select the connector type and gender of your DUT OUTPUT. Then select ECal as the Cal Kit to use for each connector. Click **Next**.

6. At the **Scalar Mixer Calibration Step 1 of 2** dialog, connect the power sensor to the Port 1 test cable, then click **Measure**. The data will be used to correct for input mismatch errors. Beginning with PNA Rev 6.0, power measurements are no longer required at port 2.

7. At the **Scalar Mixer Calibration Step 2 of 2** dialog, connect the ECal module Port A to the Port 1 cable, and Port B to the Port 2 cable. Then click **Measure**. This portion of the calibration gathers the linear (non-frequency-translating) error terms of the test setup at the input and output frequencies.
8. At the **Calibration completed** dialog, you can choose to save the SMC calibration as a User Cal Set. Otherwise, click **Finish** to complete the SMC calibration. Correction is turned ON and applied to the SMC trace.

**What is happening?**

Because Fixed Output or Fixed Input FCA measurements require an external source to sweep, the measurements are much slower. When correction is ON, you will see that there are times when nothing is happening on the screen. This is because there are background measurements being made but not displayed.

This is exactly the same as when full 2-port correction is applied to an S-parameter. All four parameters are measured, then correction is applied, then all four measurements are updated. This occurs much faster when there is no external source. With correction OFF, the traces are updated as the data is measured. You can see this taking place by creating the following measurements.

**Create S12 Upconverter, S11 Input and S22 Output Match**

1. Press **Trace**, then **Application**. Click **S11, SC12, and S22**, then click **OK** to add these measurements to the same channel.

2. While the source is sweeping, watch the source port indicator on the front of the PNA. First, the port 1 indicator will light for two sweeps, then the port 2 indicator will light for 2 sweeps. During the last sweep, all 4 traces update.

3. Turn correction OFF for ALL measurements. Notice that the relevant traces update as the sweep is occurring.

With the SC12 measurement you can see the reciprocity of the mixer.

Note: With the recent improvements to FCA, this step is MUCH easier than before. SMC forward and reverse measurements can now reside in the same channel and are calibrated automatically at the same time.
1-Jan-2007  MX New topic
Embedded LO Measurements

The Embedded LO feature allows you to make VMC measurements of mixers that have a FIXED LO inside the DUT. SMC (Scalar) measurements are not allowed because phase information is used in the LO measurement process.

**Note:** This feature is available as Opt 084, and must be enabled.

Measurements of these devices are challenging for a couple of reasons:

1. The VMC measurement process requires the use of a reference mixer that has the same LO frequency as the DUT. A separate internal or external source must be used for the reference mixer LO. **This LO must be controlled by the PNA.** A PNA with an internal second source is much faster.

2. The PNA receivers need to be tuned to the correct frequency to measure the mixer output, which is highly dependent on the exact LO frequency.

For both of these reasons, the PNA is required to accurately know the frequency of the Embedded LO.

**How we measure the embedded LO**

The nominal frequency of the embedded LO is input into the **Mixer Setup dialog.** The LO source for the reference mixer is tuned to this value.

Before each DUT measurement sweep, background sweeps are made to determine the frequency of the embedded LO to a configurable degree of accuracy.

**Background sweeps...**

- **Broadband** Sweep - rough measurement of the embedded LO frequency, made around a selectable data point over a selectable frequency span. The input signal to the DUT is tuned to a selectable CW frequency. The reference mixer is not used. The B receiver is swept across a selectable span around the anticipated output frequency. The difference between the frequency of the found signal and the desired output frequency is then applied as an adjustment to the Reference Mixer LO frequency.

- **Precise** Sweep The reference mixer LO is tuned to the result of the broadband measurement. VC21 is measured at the selectable data point. Measurements of phase versus time are made, from which the exact offset frequency is computed, until either the tolerance value or maximum iterations are met.

**How to make a VMC measurement of a DUT with an Embedded LO**

1. Create a standard VMC measurement.
2. In the mixer setup dialog, enter the nominal frequency of the embedded LO as the LO frequency.
3. Perform a VMC calibration.
4. **Launch and complete the Embedded LO Mode dialog box** (below)
The LO source for the Reference Mixer can be either:

- An **Internal** source when using a [PNA-X that has two sources](#).
- An **External** source:
  - Must be controlled by the PNA. [Learn how](#).
  - Must be locked to the PNA using the [10 MHz reference](#).

### During Calibration
The LO source is shared between the Reference Mixer and the Calibration Mixer/Filter. This requires a splitter when using an external source, as shown in the following image.

![Diagram](#)

### During the Measurement
Only the Reference Mixer uses the LO source. Terminate the LO source port that is no longer used by the Calibration Mixer/Filter to ensure that the match seen by the Reference Mixer LO port does not change after the calibration, as shown in the following image. This precaution is not necessary when using the internal second source (ports 3 and 4) of the PNA-X.

---

[201]
### How to Launch the Embedded LO Mode dialog box

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Trace</td>
</tr>
<tr>
<td></td>
<td>2. then Measure</td>
</tr>
<tr>
<td></td>
<td>3. then Embedded LO</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Not Available</td>
<td>1. Click Trace/Chan</td>
</tr>
<tr>
<td></td>
<td>2. then Measure</td>
</tr>
<tr>
<td></td>
<td>3. then Embedded LO</td>
</tr>
</tbody>
</table>
The Tuning Settings balance LO measurement speed versus accuracy. You can tell that accuracy is becoming compromised when noise starts to appear on the VMC measurement trace.

Scroll up to learn more about the Embedded LO measurement process.

**Embedded LO Mode On** Check to enable measurement of the Embedded LO.

**Tuning Point** Select, or specify, the data point in the mixer sweep that will be used to find the embedded LO frequency. If a marker is enabled, that data point can be used. For broadband and Precise sweeps, choose a point in the mixer sweep where noise is least likely to be found. This is generally the center of a sweep or the center of a filter if used.

**LO Frequency Delta** The absolute difference between the measured embedded LO frequency and the LO setting that is entered in the Mixer Setup dialog. This value is updated each time the embedded LO frequency is measured. Entering a value is a way to change the LO frequency on the mixer setup without invalidating the calibration.

- **Reset** Set the LO Frequency Delta back to 0 Hz
- **Find Now** The PNA finds and measures the actual LO frequency using the current dialog settings. This data is displayed in the Status box.

**Tuning Settings** These settings determine the amount of time spent versus the degree of accuracy to which the LO Frequency is measured. You can tell that accuracy is becoming compromised when noise starts to appear on the VMC measurement trace.

- **Reset** Set all Tuning Settings back to the defaults.
- **Broadband and Precise** Do the entire tuning process for each background sweep.
**Precise only**  Does NOT perform broadband tuning on each sweep. Use this setting when the embedded LO is stable.

**Disable tuning**  Only the previously measured LO Frequency Delta is applied to the reference mixer LO and PNA receivers.

**Sweep Span**  Narrowing the sweep span limits the number of data points that are measured in the broadband sweep and makes the measurement faster.

**Max Iterations**  The maximum number of Precise sweeps to make. When this number is reached, the final measurement is used.

**Tolerance**  When two consecutive Precise measurements are made within this value, the final measurement is used. If this is not achieved within the Max Iterations value, then the last measurement is used. This is the best of the 'Tunings settings' to change to improve accuracy.

**Tuning IFBW**  IF Bandwidth used for Broadband and Precise tuning sweeps. The larger the IFBW, the faster the sweep, but the signal may not be found.

**Tune every**  Set the interval at which tuning is performed before a measurement sweep. 'Tune every 3 sweeps' means that every third measurement sweep is preceded by tuning sweeps. If the embedded LO drifts, or if regularly changing DUTs, use 'Tune every 1 sweep'.

**Status**  Allows textual and graphical representation of the Embedded LO measurement sweeps.

**Clear**  Removes the text information currently being displayed.

**Graph**  Launches the following graphical (spectrum analyzer type) display sweeps of the latest embedded LO measurement.
Embedded LO Diagnostic dialog box help

Presents a graphical (spectrum analyzer type) display of the latest embedded LO measurement.
Click Previous and Next to view available Broadband and Precise sweeps. The LO Frequency is displayed in the Marker annotation.

Last Modified:

5-Oct-2007  Added config image and text
5-Jul-2007   Update access point
6-Apr-2007   MX New topic
Characterize Adaptor Macro

This external Macro application creates an S2P file that models a device such as an adaptor, the input OR output side of a test fixture, or an on-wafer probe head. This is done by calculating the four S-parameters of the device from two 1-port calibrations; one on side A of the device and the other on side B of the device. Such S2P files can be used for embedding (adding) or de-embedding (removing) the device from subsequent S-parameter measurements and FCA calibrations.

This application, along with the FCA Embed/De-embed feature, can be especially useful when performing FCA calibrations.

- An SMC calibration requires a power meter measurement at the port 1 reference plane. This could be very difficult in on-wafer applications where the measurement reference plane is at the tip of a probe. This macro, in conjunction with the Embed/De-embed feature, enables you to model the probe and connect the power sensor at the coax connector where the probe connects.

- Likewise, a VMC calibration requires that a calibration mixer be used for the Thru standard. Again, this can be very difficult in on-wafer applications where the measurement reference plane is at the tip of a probe. This macro, in conjunction with the Embed/De-embed feature, enables you to model the probe and connect the calibration mixer at the coax connectors where the probe connects.

Also in this topic:

To Embed or De-embed?

Procedures

### How to start the Characterize Adaptor Macro

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press repeatedly</td>
<td>1. Click System</td>
</tr>
<tr>
<td>2. Press TS</td>
<td>2. then Macro</td>
</tr>
<tr>
<td></td>
<td>3. then AdaptorChar</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press SYSTEM</td>
<td>1. Click Utility</td>
</tr>
<tr>
<td>2. then [Macro]</td>
<td>2. then Macro</td>
</tr>
<tr>
<td>3. then [AdaptorChar]</td>
<td>3. then AdaptorChar</td>
</tr>
</tbody>
</table>
Important Notes

- The device to be characterized (probe, adapter...) MUST be reciprocal (S21 = S12).
- Two 1-port calcs must be performed and saved to Cal Sets BEFORE using the Characterize Adaptor application.
- The frequencies and number of points of the two Cal Sets MUST be identical.
- **CRITICAL**: The calculations that are performed to create the S2P file require that Calset 1 ALWAYS be from the side closest to the PNA and Calset 2 ALWAYS be from the other side of the device.
- If your application that uses the resulting S2P file requires that the ports be reversed, it must be done on the S2P file using an external program such as Microsoft Excel.
- The majority of this topic describes the characterization of a 2-port device from two 1-port cal sets. However, you can also use the cal sets from two 2-port calibrations, or one 2-port and two 1-port calibrations. See procedures for both.

Learn more about the Characterize Adaptor Macro.(scroll up)

**Connected** <PNA host name> The two 1-port Cal Sets can reside in another PNA. Click to connect to another PNA that is DCOM configured. Learn how to configure DCOM.

**Adaptor Type** Select the type of device to be characterized.
**Note:** The image that appears in the macro does not influence the calculations. It only appears to help you visualize the measurement reference plane of the Cal Sets.

**Select Calset 1 and Calset-2** Select a 1-port Cal Set from each list. Although all Cal sets are listed, only the Cal Sets that have error terms to satisfy a 1-port calibration may be used.

**Cal Port** Select the port within the selected Cal Set which represents the modeled device. The Cal Ports must be the same for both selected Cal Sets.

**Characterize and Save** Calculates four S-parameters, then invokes the Save As dialog with S2P file type. This button is not available until valid Cal Sets and Cal Ports are selected.

**Close** Closes the dialog box.

---

**To Embed or De-embed?**

To make an accurate measurement, the setup configuration during the DUT measurement MUST exactly match the setup configuration during Calibration. In other words, if you calibrate with an adapter, you must also measure the DUT with the same adapter.

However, the PNA provides some flexibility by allowing you to *Virtually* add (embed) or remove (de-embed) an adapter from either the measurement or an FCA calibration. Knowing how to do this can be confusing.

In the following, if you are NOT making an FCA measurement, then your only choice is A. Also, “adapter ” can mean any type of 2-port device:

To perform a calibration WITHOUT the adapter, but make DUT measurements WITH the adapter, do either of the following:

- A. Remove (de-embed) the adapter from the DUT measurement OR
- B. Add (embed) the adapter during the FCA Calibration.

To perform a calibration WITH the adapter, but make DUT measurements WITHOUT the adapter, do either of the following:

- A. Add (embed) the adapter during the DUT measurement, OR
- B. Remove (de-embed) the adapter from the FCA Calibration.

---

**Procedures**

- Create an S2P file using Characterize Adaptor Macro
- De-embed the S2P file from DUT measurement
- Embed the S2P file in DUT measurement
- Embed or De-embed the S2P file with FCA Cal
- De-Embedding a Fixture that has a THRU Standard
- De-Embedding a Fixture with No THRU Standard

Create an S2P file using the Characterize Adaptor Macro
1. Configure your PNA measurement (frequency span, power level, IF bandwidth, and number of points).

2. Perform a 1-port SmartCal at the reference plane. Save the cal to a User Cal Set using a descriptive name (for example, Ref Plane).

3. Connect the adapter to be characterized at the reference plane.

4. Perform another 1-port SmartCal at the end of the adapter. Save it to a User Cal Set using a different descriptive name (for example, Adapt End).

5. Start the Characterize Adaptor Macro.

6. In the Select Calset1 field of the dialog box, select the Cal Set for the reference plane (from step 2 above).

7. In the Select Calset2 field of the dialog box, select the Cal Set for the end of the adapter (from step 4 above).

8. Click Characterize and Save. In the resulting dialog box, enter the .S2P file name and location.

9. Click Close.

To De-embed the adapter (S2P file) from subsequent S-parameter measurements:

Note: Subsequent measurements must have the same or smaller frequency range (within the Start / Stop frequencies) as that of the S2P file.

1. Perform a 2 port SOLT calibration without the adapter/fixture.

2. Select 2-port De-embedding:
   
   1. For PNA-X and E836xC: click Response, then Cal, then More, point to Fixtures, then click 2 port De-embedding.
   
   2. For E836xA/B, click Calibration, point to Fixtures, then click 2 port De-embedding.

3. Select the Port to add the adapter to, then select User Defined (S2P file).

4. Click Use S2P file and select the S2P file created using the Characterize Adaptor macro.

5. Check Enable De-embedding, then click Close.

6. Enable Fixturing:
   
   1. For PNA-X and E836xC: click Response, then Cal, then More, point to Fixtures, then click Fixturing on/OFF.
   
   2. For E836xA/B, click Calibration, then Fixturing on/OFF.

7. Sim appears in the Status Bar to indicate that Fixture Simulation is ON.

To Embed the adapter (S2P file) into subsequent S-parameter measurements:
1. Perform a 2 port SOLT calibration including the adapter. Note the port number on which the adapter is calibrated.

2. Select Port Matching:
   
   1. For PNA-X and E836xC: click Response, then Cal, then More, point to Fixtures, then click Port Matching
   
   2. For E836xA/B, click Calibration, point to Fixtures, then click Port Matching.

3. Under Choose Circuit Model for Matching, select the Port that the adapter was on during calibration, then select User Defined (S2P file).

4. Press Use S2P File and navigate to the S2P file created using the Characterize Adaptor macro.

5. Check Enable Port Matching, then click Close.

6. Enable Fixturing:
   
   1. For PNA-X and E836xC: click Response, then Cal, then More, point to Fixtures, then Fixturing on/OFF.
   
   2. For E836xA/B, click Calibration, then Fixturing on/OFF.

7. Sim appears in the Status Bar to indicate that Fixture Simulation is ON.

**To Embed or De-embed the S2P file with FCA Cal:**

1. Configure the mixer SMC or VMC measurement (frequency span, power level, IF bandwidth, and number of points).

2. Click Calibration, then Calibration Wizard.

3. On the Calibration Setup dialog box, check Waveguide/In-fixture/On-Wafer Setup, then click Next.

4. On the Waveguide/In-fixture/On-Wafer Setup dialog box, click Help to learn how to Embed or De-embed the S2P file.

**De-Embedding a Fixture that has a THRU Standard**

A test fixture is generally regarded as a single 'bed' in which a DUT is placed. However, for modeling purposes such as this, it is separated into two circuits: Fixture A on the input of the DUT, and Fixture B on the output.

Use this procedure to perform calibrations WITHOUT the test fixture while making measurements WITH the test fixture. A calibration is performed once WITH the test fixture, and then again as it wears with use and electrical performance changes. The fixture is de-embedded from subsequent measurements to match the regular calibrations that are performed without the fixture.

If you have a THRU standard for your test fixture, you can perform a full 2-port calibration in the fixture, and from that create the required S2P files for de-embedding.
1. Perform a full 2-port CAL 1 at the connections of the PNA to the fixture as shown above. Save to MyCalSet1.

2. Perform a full 2-port CAL 2 where the DUT is inserted (reference plane). Save to MyCalSet2.

Follow the Create an S2P file procedure, beginning with step 6, using the following selections:

1. Create #1 S2P file:
   
   1. For CalSet1, choose MyCalSet1 and select CalPort=1
   2. For CalSet2, choose MyCalSet1 and select CalPort=2
   3. Save to FixtureA.s2p

2. Create #2 S2P file:
   
   1. For CalSet1, choose MyCalSet2 and select CalPort=1
   2. For CalSet2, choose MyCalSet2 and select CalPort=2
   3. Save to FixtureB.s2p

Follow steps in To De-embed the adapter ...

Perform these steps TWICE; once for each of the following S2P files:

1. For PNA Port 1, select FixtureA.s2p
2. For PNA Port 2, select FixtureB.s2p

De-Embedding a Fixture with No THRU Standard

This procedure is a slight modification of the above. Cal 2 is performed from two 1-port cals when a THRU standard for the fixture is not readily available.

1. Perform a full 2-port CAL 1 at the connections of the PNA to the fixture as shown above. Save to MyCalSet1.

2. CAL 2 is performed using two 1 port cals
- Cal2A at the Fixture A / DUT plane. Save to MyCalSet2A
- Cal2B at the Fixture B / DUT plane. Save to MyCalSet2B

In the Create an S2P file...Step 6 above, except:

1. Create #1 S2P file:
   1. For CalSet1, choose MyCalSet1 and select CalPort=1
   2. For CalSet2, choose MyCalSet2A and select CalPort=2
   3. Save to FixtureA.s2p

2. Create #2 S2P file:
   1. For CalSet1, choose MyCalSet1 and select CalPort=2
   2. For CalSet2, choose MyCalSet2B and select CalPort=2
   3. Save to FixtureB.s2p

Follow steps in To De-embed the adapter above, except:

1. For PNA Port 1, select FixtureA.s2p
2. For PNA Port 2, select FixtureB.s2p

Last modified:

12-Feb-2008   New procedures
24-Jan-2008   Fixed error in procedures and added section
30-Nov-2007   Clarified and highlight order of calsets.
26-Feb-2007   Fixed " Note: Subsequent measurements...".
12-Sept-2006  Added link to programming commands
SMC with a Booster Amp

If your mixer measurement requires more source power on the input than the PNA can provide, a booster amplifier can be used to provide the additional power. This topic describes how to configure and make a calibrated SMC measurement using a booster amplifier.

Connect

Connect the booster amplifier between the Source-Out and Coupler-Thru connectors on the front-panel as shown in the following diagram.

### Measurement and Calibration Setup

In the following procedure:
- **Test Port power** is the power level out of the source.
- **Corrected power** is the power level you require at the mixer input and output.

This procedure assumes you will applying stimulus power to the mixer **input** to make SC11 and SC21 measurements, and to the **output** of the mixer to make SC22 and SC12 measurements.

1. Determine the gain of the booster amplifier. If the gain has significant slope across the **input and output range** of the mixer, see **Booster Amp with a Gain Slope**.

2. Determine the corrected power for both the input (port 1) and output (port 2) of the mixer.

3. Calculate the Test Port power for both ports by subtracting the gain of the amplifier from both the input and output corrected power levels.

For example, the following values assume a 25 dB booster amp on port 1 as in the diagram above.

<table>
<thead>
<tr>
<th></th>
<th>Corrected Power</th>
<th>Amp Gain</th>
<th>Test Port Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1 (input)</td>
<td>0 dBm</td>
<td>25 dB</td>
<td>-25 dBm</td>
</tr>
<tr>
<td>Port 2 (output)</td>
<td>-20 dBm</td>
<td>25 dB</td>
<td>-45 dBm</td>
</tr>
</tbody>
</table>

4. On the PNA **Power dialog**, clear the **Port Power Coupled** checkbox, which allows different power levels for each port.

5. Enter the calculated **Test Port Power** values for each port.

6. During the SMC Cal Wizard **Select DUT Connectors and Cal Kits** dialog, click **View/Modify Source Cal Settings** to invoke the **Source Calibration Settings dialog**.

7. In **Power Offset**, enter the booster amplifier gain.

**Booster Amp with a Gain Slope**

SMC calibration takes place over the entire input and output range of the mixer. Therefore, the booster amplifier will also be subjected to the entire input and output frequency range of the mixer.

To compensate for a gain slope, you might have to experiment with the source attenuator setting, power-offset value, and initial power value to get a combination that will not cause the PNA source to go unlevelled during or after the cal.

For example, assume the booster amp gain is 30 dB at the low end, and 20 dB at the high end. If you enter 30 dB for the power offset value, the PNA might run out of ALC range when the actual gain drops to 20 dB. The PNA will try to increase its source power to account for the 10 dB gain drop. Therefore, pick a power offset value that is in the middle of the amplifier gain band (25dB).

If possible, select a PNA attenuator setting that puts the ALC approximately in the middle of its range at the desired corrected power with the mid-band gain. This condition means the ALC can set the power higher and lower to account for the gain slope, without unleveling.
If the gain slope is too large, then there may not be a setting that prevents a source unlevel. In this case, a flatter booster amp must be used.
Time Domain

Time Domain allows you to view a device response as a function of time. The following are discussed in this topic:

- Overview
- How the PNA Measures in the Time Domain
- Calibration for Time Domain
- Transmission Measurements
- Measurement Response Resolution
- Measurement Range and Alias Responses
- How to make Time Domain Settings
- Gating
- Window Settings

Note: Time Domain measurements are only available on PNAs with Option 010. See PNA Options.

See the updated App Note: Time Domain Analysis Using a Network Analyzer.

Overview

In normal operation, the PNA measures the characteristics of a test device as a function of frequency. With Time Domain (opt 010), the frequency information is used to calculate the inverse Fourier transform and display measurements with time as the horizontal display axis. The response values appear separated in time, allowing a different perspective of the test device's performance and limitations.

The graphic below compares the same cable reflection measurement data in both the frequency and time domain. The cable has two bends. Each bend creates a mismatch or change in the line impedance.

- The frequency domain S11 measurement shows reflections caused by mismatches in the cable. It is impossible to determine where the mismatches physically occur in the cable.
- The time domain response shows both the location and the magnitude of each mismatch. The responses indicate that the second cable bend is the location of a significant mismatch. This mismatch can be gated out, allowing you to view the frequency domain response as if the mismatch were not present. Distance Markers can be used to pinpoint the distance of the mismatch from the reference plane.
How the PNA Measures in the Time Domain

Time domain transform mode simulates traditional Time-Domain Reflectometry (TDR), which launches an impulse or step signal into the test device and displays the reflected energy on the TDR screen. By analyzing the magnitude, duration, and shape of the reflected waveform, you can determine the nature of the impedance variation in the test device.

The PNA does not launch an actual incident impulse or step. Instead, a Fourier Transform algorithm is used to calculate time information from the frequency measurements. The following shows how this occurs.

A single frequency in the time domain appears as a sine wave. In the following graphic, as we add the fundamental frequency (F₀), the first harmonic (2F₀), and then the second harmonic (3F₀), we can see a pulse taking shape in the Sum waveform. If we were to add more frequency components, the pulse would become sharper and narrower. When the PNA sends discrete frequencies to the test device, it is in effect, sending individual spectral pieces of a pulse separately to stimulate the test device.

![FREQ GRAPHIC]

During an S11 reflection measurement, these incident signals reflect from the test device and are measured at the A receiver. This is when the time domain transform calculations are used to add the separate spectral pieces together.

For example, consider a short length of cable terminated with an open. All of the power in the incident signal is reflected, and the reflections are 'in-phase' with the incident signal. Each frequency component is added together, and we see the same pattern as the simulated incident would have looked (above). The magnitude of the reflection is related to the impedance mismatch and the delay is proportional to the distance to the mismatch. The x-axis (time) scale is changed from the above graphic to better show the delay.

![TIME GRAPHIC]

Alternately, the same cable terminated with a short also reflects all of the incident power, but with a phase shift of 180 degrees. As the frequency components from the reflection are added together, the sum appears as a negative impulse delayed in time.
Calibration for Time Domain

For simplicity, we have discussed incident signals reflecting off discontinuities in the test device. By far the most common network analyzer measurement to transform to time domain is a ratioed S11 measurement. An S11 reflection measurement does not simply display the reflections measured at the A receiver - it displays the ratio (or difference) of the A receiver to the Reference receiver. In addition, the S11 measurement can also be calibrated to remove systematic errors from the ratioed measurement. This is critical in the time domain as the measurement plane, the point of calibration, becomes zero on the X-axis time scale. All time and distance data is presented in reference to this point. As a result, both magnitude and time data are calibrated and very accurate.

The following shows where the time domain transform occurs in the PNA data flow: (see Data Access Map)

1. Acquire raw receiver (A and R1) data
2. Perform ratio (A/R1)
3. Apply calibration
4. Transform data to time domain
5. Display results

Therefore, although a time domain trace may be displayed, a calibration is always performed and applied to the frequency domain measurement which is not displayed.

Transmission Measurements

The most common type of measurement to transform is an S11 reflection measurement. However, useful information can be gained about a test device from a transformed S21 transmission measurement. The frequency components pass through the test device and are measured at the B receiver. If there is more than one path through the device, they would appear as various pulses separated in time.

For example, the following transmission measurement shows multiple paths of travel within a Surface Acoustic Wave (SAW) filter. The largest pulse (close to zero time) represents the propagation time of the shortest path through the device. It may not be the largest pulse or represent the desired path. Each subsequent pulse represents another possible path from input to output.
Triple travel is a term used to describe the reflected signal off the output, reflected again off the input, then finally reappearing at the output. This is best seen in a time domain S21 measurement.

**Measurement Response Resolution**

In the previous paragraphs, we have seen that using more frequency components causes the assembled waveform to show more detail. This is known as measurement response resolution, which is defined as the ability to distinguish between two closely spaced responses.

Note: Adjusting the transform time settings improves display resolution, but not measurement resolution.

The following graphic shows the effect of both a narrow and wide frequency span on the response resolution. The wider frequency span enables the analyzer to resolve the two connectors into separate, distinct responses.

**Resolution Formula**

For responses of equal amplitude, the response resolution is equal to the 50% (-6 dB) points of the impulse width, or the step rise time which is defined as the 10 to 90% points as shown in the following image.
The following table shows the approximated relationship between the frequency span and the window selection on response resolution for responses of equal amplitude.

<table>
<thead>
<tr>
<th>Window</th>
<th>Low-pass step (10% to 90%)</th>
<th>Low-pass impulse (50%)</th>
<th>Bandpass impulse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum</td>
<td>0.45 / f span</td>
<td>0.60 / f span</td>
<td>1.20 / f span</td>
</tr>
<tr>
<td>Normal</td>
<td>0.99 / f span</td>
<td>0.98 / f span</td>
<td>1.95 / f span</td>
</tr>
<tr>
<td>Maximum</td>
<td>1.48 / f span</td>
<td>1.39 / f span</td>
<td>2.77 / f span</td>
</tr>
</tbody>
</table>

For example, using a 10 GHz wide frequency span and a normal window in Bandpass impulse mode, response resolution (in time) equals:

- Time Res = 1.95 / frequency span
- Time Res = 1.95 / 10 GHz
- Time Res = 195 ps

To calculate the physical separation (in distance) of the responses which can be resolved, multiply this value times the speed of light (c) and the relative velocity (Vf) of propagation in the actual transmission medium. In this case, Vf = 0.66 for polyethylene dielectric.

- Distance Res = 195 ps x c x Vf
- Distance Res = 195 ps x (2.997925 E8 m/s) x .66
- Distance Res = 38 mm

For reflection measurements, because of the 2-way travel time involved, this means that the minimum resolvable separation between discontinuities is half of this value or 19 mm.

Although a wider frequency span causes better measurement resolution, the measurement range becomes limited. Also, increasing the frequency range can cause a measurement calibration to become invalid. Be sure to adjust the frequency span BEFORE performing a calibration.
Measurement Range and Alias Responses

Measurement range is the length in time in which true time domain responses can be seen. The measurement range should be large enough to see the entire test device response without encountering a repetition (alias) of the response. An alias response can hide a true time domain response.

To increase measurement range in both modes, change either of these settings:

- Increase the number of points
- Decrease the frequency span

Notes:

- After making these settings, you may need to adjust the transform time settings to see the new measurement range.
- Decreasing the frequency span degrades measurement resolution.
- Make frequency span and number of points settings BEFORE calibrating.
- Maximum range also depends on loss through the test device. If the returning signal is too small to measure, the range is limited regardless of the frequency span.

Alias Responses

An alias response is not a true device response. An alias response repeats because each time domain waveform has many periods and repeats with time (see How the PNA Measures in the Time Domain). Alias responses occur at time intervals that are equal to 1/frequency step size.

The PNA adjusts the transform time settings so that you should only see one alias free range on either side (positive and negative) of zero time. However, these settings are updated only when one of the toolbar settings are changed.

To determine if a response is true, put a marker on the response and change the frequency span. A true device response will not move in time. An alias response will move.

For example, in the above graphic, the marker 1 response occurs at 14.07 inches. When the frequency span is changed, this response remains at 14.07 inches. The marker 2 response moves.

Range Formula
You can calculate the alias-free measurement range (in meters) of the PNA using the following formula for TDR (reflection) measurements:

\[ \text{Range (meters)} = \left(\frac{1}{f}\right) \times V_f \times c \]

Where:

- \( f \) = frequency step size (frequency span/number of points-1)
- \( V_f \) = the velocity factor in the transmission line
- \( c \) = speed of light = 2.997925 E8 m/s

For example: For a measurement with 401 points and a span of 2.5 GHz, using a polyethylene cable (Vf = 0.66)

- Range = \( \left(\frac{1}{2.5\times10^9 / 400}\right) \times 2.997925 \times 6.6 \times 2.997925 \times 6.6 \)
- Range = 32 meters

In this example, the range is 32 meters in physical length. To prevent the time domain responses from overlapping or aliasing, the test device must be 32 meters or less in physical length for a transmission measurement.

To calculate the one-way distance for a reflection measurement rather than round-trip distance, simply divide the length by 2. In this case, the alias-free range would be 16 meters.

**How to make Time Domain Settings**

The following launches the **Time Domain toolbar**

On the toolbar, click **More...** to launch the Time Domain dialog box

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Trace</td>
</tr>
<tr>
<td>2. then Transform</td>
<td>2. then Transform</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press ANALYSIS</td>
<td></td>
</tr>
<tr>
<td>2. then [Transform]</td>
<td>1. Click Marker/Analysis</td>
</tr>
<tr>
<td>3. then [More]</td>
<td>2. then Transform</td>
</tr>
</tbody>
</table>
Transform dialog box help

**Category**  Select Transform, Window, or Gating

**Transform**  Turns time domain transform ON and OFF.

**Coupling Settings**  Launches the Trace Coupling Settings dialog box.

**Time Settings**

The following settings adjust the display resolution, allowing you to zoom IN or OUT on a response. They do NOT adjust measurement range or measurement resolution.

These settings automatically update (when one of these values are updated) to limit the display to one alias-free response on either side of zero time.

**Start**  Sets the transform start time that is displayed on the PNA screen.

**Note:** Zero (0) seconds is always the measurement reference plane. Negative values are useful if moving the reference plane.

**Stop**  Sets the transform stop time that is displayed on the PNA screen.

**Center**  Sets the transform center time that is displayed in the center of the PNA screen.

**Span**  Sets the transform span time that is split on either side of the Center value.

**Transform Mode**

Transform modes are three variations on how the time domain transform algorithm is applied to the frequency domain measurement. Each method has a unique application.
<table>
<thead>
<tr>
<th>Mode</th>
<th>Benefit - application</th>
<th>Limitation</th>
</tr>
</thead>
</table>
| Low pass Impulse | Highest resolution.  
Most useful for seeing small responses in devices that pass low frequencies, such as cables.                                                                                                            | In both Low pass modes, frequencies down to DC and negative frequencies are extrapolated. Therefore, the Start frequency is adjusted when you click **Set Freq. Low Pass**  
Because this will affect calibration accuracy, be sure to calibrate AFTER completely setting up your time domain measurement. |
| Low pass Step     | Easiest to identify inductive and capacitive discontinuities in devices that pass low frequencies, such as cables.                                                                                                        |                                                                                                                                                                                                          |
| Band pass Impulse| Easiest method - can be used with any frequency sweep.  
Most useful for measuring band limited devices such as filters and DC blocked cables.                                                                                                                             | Does NOT show capacitive and inductive reactance  
For the same frequency span and number of points, band pass mode has twice the impulse width, which hides closely spaced responses degrading the response resolution. |

The following chart shows how to interpret results from various discontinuity impedances using Low pass Step and either Low pass or Band pass Impulse modes.

<table>
<thead>
<tr>
<th>IMPEDANCE</th>
<th>STEP RESPONSE</th>
<th>IMPULSE RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPEN</td>
<td>Unity Reflection</td>
<td>Unity Reflection</td>
</tr>
<tr>
<td>SHORT</td>
<td>Unity Reflection = 180</td>
<td>Unity Reflection = 180</td>
</tr>
<tr>
<td>RES/STOR</td>
<td>Unity Reflection</td>
<td>Unity Reflection</td>
</tr>
<tr>
<td>R &gt; Z_0</td>
<td>Unity Reflection</td>
<td>Unity Reflection</td>
</tr>
<tr>
<td>RES/STOR</td>
<td>Unity Reflection</td>
<td>Unity Reflection</td>
</tr>
<tr>
<td>R &lt; Z_0</td>
<td>Unity Reflection</td>
<td>Unity Reflection</td>
</tr>
<tr>
<td>INDUCTOR</td>
<td>Unity Reflection</td>
<td>Unity Reflection</td>
</tr>
<tr>
<td>CAPACITOR</td>
<td>Unity Reflection</td>
<td>Unity Reflection</td>
</tr>
</tbody>
</table>

**Effect on Measurement Range**

**Band pass mode** - measurement range is inversely proportional to frequency step size.

**Low pass mode** - measurement range is inversely proportional to the fundamental (start ) frequency AFTER clicking **Set Freq. Low Pass**.

**Set Freq. Low Pass**  USE ONLY IN LOW PASS MODES

Recomputes the start frequency and step frequencies to be harmonics of the start frequency. Start frequency is computed by the following formula: **Low Pass Start Frequency = Stop Frequency / Number of points.**
The computed value must always be greater than or equal to the analyzer's minimum frequency.

**Note:** The number of points or stop frequency may be changed in order to compute this value.

**Distance Marker Settings** Launches the [Distance Marker Settings](#) dialog box.

---

**Gating**

Perhaps the most beneficial feature of time domain transform is the Gating function. When viewing the time domain response of a device, the gating function can be used to "virtually" remove undesired responses. You can then simultaneously view a frequency domain trace as if the undesired response did not exist. This allows you to characterize devices without the effects of external devices such as connectors or adapters.

**Note:** When a discontinuity in a test device reflects energy, that energy will not reach subsequent discontinuities. This can "MASK", or hide, the true response which would have occurred if the previous discontinuity were not present. The PNA Gating feature does NOT compensate for this.

The following measurements images show a practical example how to use and perform gating. The test device is a 10inch cable, then a 6 dB attenuator, terminated with a short. The following four discontinuities are evident in window 2, from left to right:

1. A discontinuity in the test system cable which appeared after calibration. It is identified by marker 2 at -10.74 inches (behind the reference plane).
2. A discontinuity in the 10 inch device cable shortly after the reference plane.
3. The largest discontinuity is the attenuator and short shown by marker 1 at -12.67 dB (6 dB loss in both forward and reverse direction).
4. The last discontinuity is a re-reflection from the device cable.

We will gate IN the attenuator response. All other responses will be gated OUT.

**Window 1.** Create original S11 frequency domain trace. Shows ripple from all of the reflections.

**Window 2.** Create a new S11 trace - same channel; new window. Turn Transform ON.

**Window 3.** On the transformed trace, turn gating ON. Center the gate on the large discontinuity (2.500ns). Adjust gate span to completely cover the discontinuity. Select Bandpass gating type.

**Window 4.** On the original frequency measurement, turn Gating ON (Transform remains OFF). View the measurement without the effects of the two unwanted discontinuities. The blue trace is a measurement of the 6 dB attenuator with the unwanted discontinuities PHYSICALLY removed. The difference between the two traces in window 4 is the effect of "masking".
Learn how to launch the Transform dialog box

**Transform**

- **Category:** Gating
- **Start:** -10.000 nsec
- **Stop:** 10.000 nsec
- **Center:** 0 psec
- **Span:** 20.000 nsec

**Gating**
- Turns Gating ON and OFF.

**Coupling Settings**
- Launches the Setup Trace Coupling dialog box.
- **Start** Specifies the start time for the gate.
- **Stop** Specifies the stop time for the gate.
- **Center** Specifies the value at the center of the area that is affected by the gating function. This value can be anywhere in the analyzer range.
- **Span** Specifies the range to either side of the center value of area that is affected by the gating function.
- **Gate Type** Defines the type of filtering that will be performed for the gating function. The gate start and stop
flags on the display point toward the part of the trace you want to keep.

- **Bandpass** - **KEEPS** the responses within the gate span.
- **Notch** - **REMOVES** the responses with the gate span.

**Gate Shape** Defines the filter characteristics of the gate function. Choose from Minimum, Normal, Wide, Maximum

<table>
<thead>
<tr>
<th>Gate Shape</th>
<th>Passband Ripple</th>
<th>Sidelobe Levels</th>
<th>Cutoff Time</th>
<th>Minimum Gate Span</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum</td>
<td>±0.1 dB</td>
<td>-48 dB</td>
<td>1.4/Freq Span</td>
<td>2.8/Freq Span</td>
</tr>
<tr>
<td>Normal</td>
<td>±0.1 dB</td>
<td>-68 dB</td>
<td>2.8/Freq Span</td>
<td>5.6/Freq Span</td>
</tr>
<tr>
<td>Wide</td>
<td>±0.1 dB</td>
<td>-57 dB</td>
<td>4.4/Freq Span</td>
<td>8.8/Freq Span</td>
</tr>
<tr>
<td>Maximum</td>
<td>±0.01 dB</td>
<td>-70 dB</td>
<td>12.7/Freq Span</td>
<td>25.4/Freq Span</td>
</tr>
</tbody>
</table>

**Cutoff time** -- is the time between the stop time (-6 dB on the filter skirt) and the peak of the first sidelobe. The diagram below shows the overall gate shape and lists the characteristics for each gate shape.

- $T_1$ is the gate span, which is equal to the stop time minus the start time.
- $T_2$ is the time between the edge of the passband and the 6 dB point, representing the cutoff rate of the filter.
- $T_3$ is the time between the 6 dB point and the edge of the gate stopband.
- For all filter shapes $T_2$ is equal to $T_3$, and the filter is the same on both sides of the center time.

**Minimum gate span** -- is twice the cutoff time. Each gate shape has a minimum recommended gate span for proper operation. This is a consequence of the finite cutoff rate of the gate. If you specify a gate span that is smaller than the minimum span, the response will show the following effects:

- distorted gate shape that has no passband
- distorted shape
- incorrect indications of start and stop times
• may have increased sidelobe levels

Window Settings

There are abrupt transitions in a frequency domain measurement at the start and stop frequencies, causing overshoot and ringing in a time domain response. The window feature is helpful in lessening the abruptness of the frequency domain transitions. This causes you to make a tradeoff in the time domain response. Choose between the following:

• **Minimum Window = Better Response Resolution** - the ability resolve between two closely spaced responses.

• **Maximum Window = Dynamic Range** - the ability to measure low-level responses.

![Minimum Window vs Maximum Window](image)

Learn how to launch the Transform dialog box

![Transform dialog box](image)
Transform - Window dialog box help

**Programming Commands**

**Coupling Settings** Launches the Setup Trace Coupling dialog box.

The window settings balance response resolution versus dynamic range.

- Minimum Window = Best Response Resolution
- Maximum Window = Best Dynamic Range

The following three methods all set window size. For best results, view the time domain response while making these settings.

- **Minimum - Maximum** Move the slider with a mouse to change the window size
- **Kaiser Beta** Changes window size using a Kaiser Beta value
- **Impulse Width** Changes window size using an Impulse Width value

Learn more about Windowing (top)

---

**How to make Trace Coupling Settings**

You can launch the Trace Coupling Settings dialog box from any of the following dialog boxes:

- **Transform**
- **Gating**
- **Window**

**Programming Commands**

Learn more about using the front panel interface
Trace Coupling Settings allows you to change time domain parameters on a measurement, and have the same changes occur for all other measurements in the channel.

For example:

If you are simultaneously viewing a frequency domain measurement and time domain measurement, and Coupling is enabled in this dialog box, and ALL Gating Parameters are checked in this dialog box, and on the time domain measurement you change the Gate Span parameter, then the frequency domain measurement will automatically change to reflect the time domain gated span.

**Coupling ON/OFF**  Check to enable coupling. All of the measurements in the active channel are coupled.

The following parameters are available for coupling:

**Transform Parameters**
- **Stimulus**  Start, Stop, Center, and Span TIME settings.
- **State** (On/Off)  Transform ON and OFF
- **Window**  Kaiser Beta / Impulse Width
- **Mode**  Low Pass Impulse, Low Pass Step, Band Pass

**Gating Parameters**
- **Stimulus**  Start, Stop, Center, and Span TIME settings.
- **State** (On/Off)  Gating ON and OFF
- **Shape**  Minimum, Normal, Wide, and Maximum
- **Type**  Bandpass and Notch
Distance Marker Settings  dialog box help

Programming Commands

To launch this dialog box, click Dist. Marker Settings on the Transform dialog box.

When markers are present on a time domain measurement, distance is automatically displayed on the marker readout, marker table, and print copy. To learn how to create markers on your measurement see marker settings.

This dialog box allows you to customize the time domain distance marker readings.

These settings affect the display of ALL markers for only the ACTIVE measurement (unless Distance Marker Unit is coupled on the Trace Coupling dialog box).

Marker Mode  Specifies the measurement type in order to determine the correct marker distance.

- Select Auto for S-Parameter measurements.
- Select Reflection or Transmission for arbitrary ratio or unratioed measurements.

Auto  If the active measurement is an S-Parameter, automatically chooses reflection or transmission. If the active measurement is a non S-Parameter, reflection is chosen.

Reflection  Displays the distance from the source to the receiver and back divided by two (to compensate for the return trip.)

Transmission  Displays the distance from the source to the receiver.

Units  Specifies the unit of measure for the display of marker distance values.

Velocity Factor  Specifies the velocity factor that applies to the medium of the device that was inserted after the measurement calibration. The value for a polyethylene dielectric cable is 0.66 and 0.7 for Teflon dielectric. 1.0 corresponds to the speed of light in a vacuum. This is useful in Time Domain for accurate display of time and distance markers.

This setting can also be made from the Electrical Delay and Port Extensions dialog boxes.

Last Modified:

11-Sep-2007     Edits to resolution and rage formulas
Front Panel Tour
PNA-L and Microwave Models

See the PNA-X Front Panel Tour

Click on the sections of the front panel for information.

Power Switch
Toggles the analyzer between the On and Hibernate conditions. This switch is not connected to the power supply. Learn more about powering the PNA ON and OFF.

Front-Panel Access Jumpers
Provides access to the measurement path. Learn more.

Test Ports
PNA test ports internally switch between source and receiver allowing measurement of your device in two directions. Two different lighting methods are used to indicate the source and receiver port:

For 9 GHz and below PNA models

- A green light indicates the source port.
- An orange light indicates the receiver port.

All other PNA models:

- An illuminated image next to the test port indicates the source.

See Input Damage Levels

USB
This Type A Universal Serial Bus (USB) connector allows you to connect a keyboard, mouse, ECAL module, or other USB device.
- Contact 1: Vcc; 4.75 to 5.25 VDC, 500 mA maximum.
- Protected with an automatically-resettable 1A fuse.
- Contact 2: –Data
- Contact 3: +Data
- Contact 4: Ground

See USB limitations.

3.5” Floppy Disk Drive
Installs files on the analyzer hard drive or stores data files from the analyzer. Access the disk drive by using Windows Explorer.

- Unformatted Data Capacity: 2MB
- Compatible with: High Density (2HD), and Normal Density (2DD)
- Transfer Rate: 500 kbits/second

Probe Power
The 3-pin (m) connectors supply power for active probes.

- Top Pin: Ground
- Left Pin: +15 VDC ±2%, 400 mA maximum; protected by PTC @0.5A
- Right Pin: –12.6 VDC ±5%, 300 mA maximum; protected by PTC @0.5A

(PTC=slow acting, automatically resettable fuse)
The PNA-X does NOT provide probe power.

Navigation Keys
These keys allow you to navigate through menus and dialog boxes and select choices from the active entry toolbar.

**Left / Right**
Moves left and right through menus
Moves tab-left and tab-right within dialog boxes

**Up / Down**
Moves up and down through menus
Behaves as follows in a dialog box:

- Modifies a numeric value
- Moves through items in a drop-down list
- Moves through options buttons in a group of option buttons

**Click**
Makes a selection just like a mouse click

**F1...F4 Keys**
Selects choices from the active entry toolbar. The color of the key corresponds to the active entry toolbar choice.
Learn more about Active Entry Keys.

**Entry Keys**
These keys allow you to enter values for measurement settings.

**Numeric Keys**
Selects values for measurement settings, then press Enter or G/n or M/u to complete the selection.

**Units Keys**
Completes the value selection, assigning a unit of measurement. Select either

- G/n (Giga/Nano) E12 or E-12
- M/u (Mega/micro) E6 or E-6

Then press Enter to complete the value entry.

**Decimal point**
Enter a decimal point to designate fractions of a whole number.

**Plus - Minus - Backspace Key**
Toggles between a positive and negative value entry if it is the first key pressed in the entry.
Backs up the cursor and deletes any previous selection.

**Enter**
Enter the values that you select for the measurement settings.

**Knob**
Rotate to increase or decrease the value of the active entry.

**Display Keys**
Controls window and trace configuration

**Trace**  
First press brings up the Trace Active Toolbar. Subsequent presses allow you to cycle through the measurement traces in a window, making each trace active in turn. You can make modifications only to active functions. This key also allows you to create or delete traces, using the function keys.

**Window**  
First press brings up the Window Active Toolbar, where you can create, select, and delete windows, using the function keys.

Subsequent presses of this key cycle through the windows that are currently set up, making each window active in turn. You can make size modifications only to an active window. A window must also be active to cycle through the traces in the window.

**Measure Setup**  
Allows you choose from four pre-configured measurement setups.

**Arrange**  
Allows you to choose from four window arrangements: Overlay, Stack 2, Split 3, Quad 4. [Learn More about Arranging the Display](#)

**Channel Setup Keys**

Controls channel settings.

**Start/Stop**  
Sets the frequency range of the channel.

**Center/Span**

**Power**  
Sets the source power level.

**Sweep Setup**  
Defines several sweep settings.

**Channel**  
Select an active channel, or delete the active channel. A channel must be active to modify any channel settings. [Learn more about Channels](#).

**Sweep Type**  
Sets the sweep type and associated settings.

**Trigger**  
Sets how the start of the measurement sweep is initiated.

**Avg**  
Applies measurement averages which reduces noise. The analyzer performs a complex exponential average of a number of sweeps that you specify. [Learn more about Averaging](#)

**Cal**  
Initiates a measurement calibration. The Calibration Wizard appears if you press Menu/Dialog, Cal. Otherwise, pressing the Cal key makes the calibration active entry toolbar appear. [Learn more about Calibration](#)

**Command Keys**
OK  Closes a dialog box and enters any values made in the dialog box.

Help  Launches the analyzer Help file.

Cancel  Closes a dialog box.

Menu/Dialog  Allows you to browse the menus with the Navigation keys. Also allows you to display dialog boxes by pressing Menu/Dialog and then a key in the Channel, Trace, or Utility blocks. Learn more.

Trace Setup Keys
Performs many trace settings. Learn more about Traces.

Measure  Allows you to select an S-parameter measurement. Through the dialog box you can also select an arbitrary ratio, or unratioed power measurement. Learn more.

Format  Allows you to select the format the analyzer uses to display the measurement data. Learn more.

Scale  Allows you to specify the value the analyzer uses to scale the displayed measurement response. You can also let the analyzer automatically set the Y axis scales to fit the entire measurement trace on the screen. Learn more.

Marker  Allows you to activate a marker and set the value. Markers provide numerical readout of measured data. Learn more.

Marker Table  Displays the table of values that allow you to view the data readout for all of the markers on the active trace. Learn more.

Limit Table  Displays the table of values that allow you to create pass / fail testing based on these limit segments. Learn more.

Marker Search  Provides access to the marker search functions. If there is no marker displayed, this key will activate a marker. Learn more.

Marker Function  Allows you to change measurement settings, based on the location of an active marker. If there is no marker displayed, this key will activate a marker. Learn more.

Math/Memory  Allows you to select math and memory functions that the analyzer performs on the measurement data. Learn more.
Save  Allows you to save instrument states, calibration data, and measurement data to a file. Learn more.

Changes the selected window to the full measurement screen size and then restores it to the previous window size. Restores the PNA application if minimized. Learn more.

Preset  Presets the PNA. Learn more.

Recall  Recalls a file from the hard drive. Learn more.

Print  Prints a displayed measurement. Learn more.

Macro/Local  When the analyzer is being controlled through automation, pressing this key allows the analyzer to respond to front panel key presses.

When the analyzer is in normal operation, pressing this key accesses a set of user macros that are in the form of executable files. You can title and store up to 12 macros. When you repeatedly press this key, the titles in the active entry toolbar rotate through three sets of four titles.

The executable files must already be on the hard drive and setup as a macro. Learn more.
PNA-X Front-Panel Tour

- 10.4" LCD Touchscreen
- Softkeys
- Knob

See Also

- The PNA-X Display area
- PNA-X Models / Options
- PNA-X Rear-Panel Tour

Familiar Hardkey layout, similar to Agilent 8720 and 8753 Network Analyzers

Back to the familiar layout, significantly different from legacy PNA models. Most measurement settings are made from the Stimulus Block and the Response Block.

Fully functional Hardkey/Softkey selections consistent with Menu (mouse) selections

Access ALL PNA settings from the front panel using hardkey/softkeys or from the Menu using a mouse. Both methods are consistent; learn the menu structure once, and it applies to both methods of UI navigation.

Power Switch

Used for choosing between power-on (|) and standby (O) state.
Learn to power ON and OFF the PNA.

**Test Ports**
The PNA-X is available with 2 or 4 test ports.
Learn about the Test port connectors.
Learn about the Input damage levels.

**Front panel Access Jumpers**
These connectors provide direct access to the PNA source and receivers. This allows you to make a wide variety of measurements and improve dynamic range. All PNA-X models have these same jumpers for each test port.
See the PNA-X front panel jumpers specifications

![Image of jumpers]

**USB Hub**
This USB hub contains four USB ports to power your PNA peripherals. There are also four USB ports on the rear panel.

**Limitation:** The total power consumption for all eight USB ports is limited to 4.0 amps. If this limit is exceeded, all USB ports are disabled until a device is removed and power consumption falls below the limit. When first connected, Agilent ECal modules 8509x and N4431 draw significantly more current than other modules.

**Ground terminal**
Connect a banana-type plug to this terminal for grounding to the PNA chassis.

**No probe power**
Probe power is NOT provided with the PNA-X.

**Hardkeys**

**TRACE/CHAN Keys**
Manages the Traces and Channels on the PNA display.
### Hard Key

<table>
<thead>
<tr>
<th>Invokes these Softkeys</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1, 2, 3, 4</strong></td>
</tr>
<tr>
<td>Makes the corresponding trace active.</td>
</tr>
</tbody>
</table>

| Invokes the Traces softkey menu which allows you to **create** a new trace, **select** a trace, **delete** a trace, or **maximize** the trace. |

<table>
<thead>
<tr>
<th>Channels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invokes the Channels softkey menu which allows you to manage channels.</td>
</tr>
</tbody>
</table>

### RESPONSE Keys

Performs operations on measurement traces after data is measured - not including Data Analysis operations.

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys - Click to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Meas</strong></td>
<td></td>
</tr>
<tr>
<td>Measurement selections</td>
<td></td>
</tr>
<tr>
<td>• S-Parameters</td>
<td></td>
</tr>
<tr>
<td>• Balanced Parameters</td>
<td></td>
</tr>
<tr>
<td>• Measurement Class</td>
<td></td>
</tr>
<tr>
<td>More Meas</td>
<td></td>
</tr>
<tr>
<td>• Receivers</td>
<td></td>
</tr>
<tr>
<td><strong>Format</strong></td>
<td></td>
</tr>
<tr>
<td>Format</td>
<td></td>
</tr>
<tr>
<td><strong>Scale</strong></td>
<td></td>
</tr>
<tr>
<td>Scale</td>
<td></td>
</tr>
<tr>
<td>Electrical Delay</td>
<td></td>
</tr>
<tr>
<td>Phase Offset</td>
<td></td>
</tr>
<tr>
<td>More</td>
<td></td>
</tr>
<tr>
<td>• Velocity Factor</td>
<td></td>
</tr>
<tr>
<td>• Media -Waveguide/coax</td>
<td></td>
</tr>
<tr>
<td>• Waveguide cutoff freq</td>
<td></td>
</tr>
<tr>
<td><strong>Display</strong></td>
<td></td>
</tr>
<tr>
<td>Display settings</td>
<td></td>
</tr>
<tr>
<td>Arrangements (Overlay...)</td>
<td></td>
</tr>
<tr>
<td>Windows (Managing)</td>
<td></td>
</tr>
<tr>
<td>Measurement Setups</td>
<td></td>
</tr>
<tr>
<td>Display Items</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Title</td>
</tr>
<tr>
<td>---</td>
<td>-------------</td>
</tr>
<tr>
<td>Avg</td>
<td>Averaging</td>
</tr>
<tr>
<td>Cal</td>
<td>Start Cal</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Properties (must have a Cal ON) no idea what this is
Port Ext Toolbar
Interpolation
Fixtures
• ON | Off
• Port matching
• lots more

Manage Cal Kit
Manage ECAL
System Z0
Velocity Factor

**MARKER/ANALYSIS Keys**
Control all aspects of Data Analysis including Markers and Math functions..

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys - Click to learn more</th>
</tr>
</thead>
</table>
| **Marker** | Markers  
Properties  
• Delta Markers  
• Discrete  
• Type  
• Coupled |
| **Search** | Marker Search |
| **Memory** | Data/ Memory Math  
8510 Mode |
| **Analysis** | Limit Lines  
• Limit Test  
• Global Pass/Fail |
| Trace Statistics  
Gating  
Transform  
• Windowing |
• Coupling
• Distance Marker

Equation Editor

**STIMULUS Keys**
Controls settings that determine what data (stimulus range), and how data (sweep type and triggering), is measured.

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys - Click to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freq</td>
<td>Frequency Range</td>
</tr>
<tr>
<td></td>
<td>Frequency Offset Mode</td>
</tr>
<tr>
<td>Power</td>
<td>RF Power level</td>
</tr>
<tr>
<td></td>
<td>Power Slope</td>
</tr>
<tr>
<td></td>
<td>Power and Attenuator settings</td>
</tr>
<tr>
<td>Sweep</td>
<td>Sweep Time</td>
</tr>
<tr>
<td></td>
<td>Number of Points</td>
</tr>
<tr>
<td></td>
<td>Sweep Type</td>
</tr>
<tr>
<td></td>
<td>Sweep Setup</td>
</tr>
<tr>
<td></td>
<td>Segment Table settings</td>
</tr>
<tr>
<td>Trigger</td>
<td>Trigger settings</td>
</tr>
</tbody>
</table>

**UTILITY Keys**
Performs global PNA operations.

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys - Click to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save</td>
<td>File Save</td>
</tr>
<tr>
<td></td>
<td>Manage Files</td>
</tr>
<tr>
<td></td>
<td>Define Data Saves</td>
</tr>
<tr>
<td></td>
<td>User Preset</td>
</tr>
<tr>
<td>Print</td>
<td>Print</td>
</tr>
<tr>
<td></td>
<td>Print to file</td>
</tr>
<tr>
<td></td>
<td>Page Setup</td>
</tr>
<tr>
<td>Macro/Local</td>
<td>Macro Setup</td>
</tr>
<tr>
<td></td>
<td>Run Macros</td>
</tr>
</tbody>
</table>
### Recall
- **File Recall**

### System
- **Security**
- **Configure**
  - SICL / GPIB
  - Control Panel (Windows)
  - System Z0
  - Power Meter Settings
  - Millimeter Module

### Service
- **Help**
  - Error Messages
  - About NA

### Preset
- **Preset**
- **User Preset**

### ENTRY Keys

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OK</strong></td>
<td>Closes a dialog box and enters any values made in the dialog box.</td>
</tr>
<tr>
<td><strong>Cancel</strong></td>
<td>Closes a dialog box.</td>
</tr>
<tr>
<td><strong>Help</strong></td>
<td>Launches this Help file.</td>
</tr>
<tr>
<td><strong>Bk Sp</strong></td>
<td>Back Space. Backs up the cursor and deletes any previous selection.</td>
</tr>
<tr>
<td><strong>1 to 9</strong></td>
<td>Selects values for measurement settings, then press Enter or G/n - M/u - k/m to complete the selection.</td>
</tr>
<tr>
<td><strong>G/n</strong></td>
<td>Completes the value selection, assigning a unit of measurement.</td>
</tr>
<tr>
<td><strong>M/u</strong></td>
<td></td>
</tr>
<tr>
<td><strong>k/m</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- G/n (Giga/Nano) E12 or E-12</td>
</tr>
<tr>
<td></td>
<td>- M/u (Mega/micro) E6 or E-6</td>
</tr>
</tbody>
</table>
• k/m (kilo/milli) E3 or E-3

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter</td>
<td>Enters the values that you select for the measurement settings.</td>
</tr>
<tr>
<td>Off</td>
<td></td>
</tr>
<tr>
<td>Decimal point</td>
<td>Enters a decimal point to designate fractions of a whole number.</td>
</tr>
<tr>
<td>+/-</td>
<td>Plus - Minus  Toggles between a positive and negative value entry if it is the first key pressed in the entry.</td>
</tr>
</tbody>
</table>

**Knob**

Rotate to increase or decrease the value of the active entry.

**Navigation Keys**

These keys allow you to navigate through menus and dialog boxes and select choices from the active entry toolbar.

<table>
<thead>
<tr>
<th>Hard Key</th>
<th>Invokes these Softkeys</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left / Right</td>
<td>Moves left and right through menus.</td>
</tr>
<tr>
<td></td>
<td>Moves tab-left and tab-right within dialog boxes.</td>
</tr>
<tr>
<td>Up / Down</td>
<td>Moves up and down through menus.</td>
</tr>
<tr>
<td></td>
<td>Behaves as follows in a dialog box:</td>
</tr>
<tr>
<td></td>
<td>• Modifies a numeric value</td>
</tr>
<tr>
<td></td>
<td>• Moves through items in a drop-down list</td>
</tr>
<tr>
<td></td>
<td>• Moves through options buttons in a group of option buttons</td>
</tr>
<tr>
<td>Click</td>
<td>Makes a selection just like a mouse click.</td>
</tr>
</tbody>
</table>

Last Modified:

23-Aug-2007    Added front panel jumpers image
Rear Panel Tour

PNA-L and Microwave Models

See the PNA-X Rear Panel Tour

This image includes ALL rear-panel features. Your PNA may not have this capability or look. Click on a connector for detailed information.

See this rear-panel with a 1.1 GHz CPU Board.
10 MHz Reference IN/OUT

10 MHz Reference Input  When a 10 MHz external reference signal is detected at this port, it will be used as the instrument frequency reference instead of the internal frequency reference.

10 MHz Reference Output  This BNC(f) connector outputs a frequency reference signal for use by other test equipment.

See specifications for these ports.

VGA Connector  Learn more

USB Hub

This USB hub contains four USB ports to power your PNA peripherals. There are also four USB ports on the front panel.

Limitation: The total power consumption for all eight USB ports is limited to 4.0 amps. If this limit is exceeded, all USB ports are disabled until a device is removed and power consumption falls below the limit. When first connected, Agilent ECal modules 8509x and N4431 draw significantly more current than other modules. See specifications.

USB Device  Learn more
LAN Connector
This 10/100BaseT Ethernet connection has a standard 8-pin configuration and auto selects between the two data rates.

Line Power
See specifications

GPIB Controller and Talker/Listener Ports
The PNA-X can be a GPIB Controller and Talker/Listener. Learn more.

RF Path Access
These connectors allow RF Path Configuration.

IF Path Inputs
Option 020 adds these connectors, which allow access to the PNA Receiver / IF paths.

These are labeled A, B, C/R1, D/R2, R.

- For 2-port PNA-X models, use A, B, R1, R2.
- For 4-port PNA-X models, use A, B, C, D, R.

See IF Path configuration block diagram.
Settings are made using SCPI and COM ONLY.

Power I/O
Has some of the AUX I/O connector functionality on the PNA-L and E836xB models.
See Details

28 V (BNC output)
Used to power a noise source for the Noise Figure App.

External and AUX Trigger I/O
**MEAS TRIG IN** - When enabled, PNA is triggered by signals on this connector. [Learn more.](#)

**MEAS TRIG RDY** When enabled, PNA outputs a 'READY' signal on this connector to other devices. [Learn more.](#)

**AUX TRIG 1&2 IN** When enabled, PNA accepts signals on these connectors which indicates that the external devices is ready to be triggered. [Learn more.](#)

**AUX TRIG 1&2 OUT** When enabled, PNA outputs signals on these connectors either before or after a measurement. [Learn more.](#)

**Test Set I/O**
[See Details](#)

**Bias IN and Fuses**

Apply Bias to the PNA ports through these BNC connectors. [See specifications](#)

**Material Handler I/O**
[See details](#)

**Pulse I/O**
[See Details](#)

**RF and LO OUT**

**Caution:** LO OUT has more power than previous PNA models.
See Specifications

1.6 GHz CPU  See CPU Speed / Performance

Last modified:

4-Sep-2007  Added 28V image

June 6, 2007  Added RF and IF connector images

January 11, 2007  MX New topic
Powering the PNA ON and OFF

The following is described in this topic:

- **How to...**
- Hibernate
- **ON**
- Shutdown
- **Turn OFF Autostart**

**Notes**

During boot up of Windows or of the Network Analyzer application program, do **NOT** press keys on the front panel, rotate the RPG knob, or connect a USB device. Doing so MAY lead to a front panel lockup state.

If the PNA front-panel keypad or USB ports are not responding, SHUTDOWN or RESTART the PNA; do **NOT** Hibernate. This causes the PNA drivers to awaken from hibernation in the same corrupt state.

---

**How to Log off, Shut down, Restart, or Hibernate the PNA.**

**WITH a Mouse**

1. On the PNA **System** menu, click **Windows Taskbar**
2. On the Windows Taskbar, click **Shutdown**
3. In the **What do you want the computer to do?** list, choose an action:
   - Log off (closes programs and disconnects from the network)
   - **Shut down**
   - Restart (shutdown and start)
   - **Hibernate**
4. Click **OK** to perform the action

**WITHOUT a Mouse**

- To Hibernate, BRIEFLY press the front-panel PNA power button.
- To Shutdown - ONLY if the PNA is locked and you cannot operate the mouse or keypad - Press and hold the power button for at least four seconds. **This practice should be avoided!** Repeated shutdowns in this manner WILL damage the hard drive. [Learn more about damaging the PNA hard drive.](#)
- **Recommended** - To SAFELY shutdown the PNA without a mouse, configure the PNA so you can choose
what to do when the power button is briefly pressed (as in Step 3 above). PNAs shipped after June 2005 are already configured this way:

1. From Windows Control Panel, select **Power Options**
2. Click **Advanced** Tab
3. Under **Power buttons**, select **Ask me what to do**.
4. Click **OK** to end configuration.

The next time the power button is pressed, a dialog box will ask **What do you want the computer to do?** Use the PNA front panel **Tab** and **Enter** keys to choose an action.

**Tip:** If it is not already running, press the **Preset** button (on the PNA front-panel) to start the PNA application.

### Hibernate Mode

- In hibernate mode the current **instrument state** is automatically saved to the hard disk before the PNA is powered OFF.
- When the PNA is powered ON, this instrument state is loaded, thus saving time over a full system boot-up.
- A password is NOT required to resume PNA operation after Hibernate mode.
- The hibernation state is the normal OFF state. A small amount of standby power is supplied to the PNA when it is in the hibernation mode. This standby power only supplies the power switch circuits and the 10 MHz reference oscillator; no other CPU-related circuits are powered during hibernation. To guarantee that your measurements meet the PNA specified performance, allow the PNA to **warm-up for 90 minutes** after the power button light has changed from yellow back to green.

### ON Mode

- To turn ON the PNA press the yellow power button.
- The power button will change to green when power is ON.

### Turn OFF PNA Autostart

The PNA application (835x.exe) always starts automatically when power is turned ON. To cause the PNA to NOT Autostart, do the following:

1. Minimize the PNA application.
2. From Windows Explorer, navigate to and double-click the following file: C:/Program Files/Agilent/Network Analyzer/Service/Toggle_PNA_Autostart.
The script toggles the PNA Autostart mode ON and OFF.

**Shutdown Mode**

- In shut down mode the current instrument state is NOT automatically saved before the PNA is powered OFF.
- When the PNA is again powered ON, a full system boot-up is performed and the PNA powers-up in the **preset settings**.
- A password is required to resume PNA operation after being in Shutdown mode.
- To guarantee that your measurements meet the PNA specified performance, allow the PNA to **warm-up for 90 minutes** after the power button light has changed from yellow back to green.
- The PNA should only be shut down for service or to provide security via password protection.
- The power button will change to yellow when power is OFF.

**Note:** If the PNA is locked and you cannot operate the mouse or keypad, shut down the PNA by pressing and holding the power button for at least four seconds. **This practice should be avoided!** Repeated shutdowns in this manner WILL damage the hard drive. Learn more about damaging the PNA hard drive.

**Unplugging the PNA**

- Remove the power cord from the PNA ONLY when the power button is yellow, in either Hibernate or Shutdown mode. If the power cord is removed while the power button is green (PNA ON), damage to the hard drive is **likely**.
- The button will remain yellow for several seconds after the power cord has been removed.
- When plugged back in and the power button is pressed to ON, the PNA starts in the mode it was in when the power cord was unplugged, either Hibernate or Shutdown.
Front Panel Interface

All PNA models except PNA-X

There are three ways to use the front panel keys:

- **Active Entry Toolbar** (quickest)
- **Launch Dialog Boxes**
- **Navigate Menus** (most comprehensive)

**Other Quick Start topics**

**Active Entry Toolbar**

Not all settings can be made this way. For making ALL settings use **Menus**.

You can make settings quickly using this four step procedure.

1. Press a key
2. View active entry
3. Select a function
4. Enter a value (if necessary)

**Launch Dialog Boxes**

To quickly launch MOST dialog boxes:
Navigate Menus

You can access ALL PNA functions using Menus:

(1) Press the Menu/Dialog Key

(2) Use the direction keys to navigate through the Menus. Use the "Click" key to make a selection.

(3) Other Command keys are available for cancelling or seeking Help (if necessary)
Traces, Channels, and Windows on the PNA

It is critical to understand the meaning of the following terms as they are used on the PNA.

- **Traces - Managing**
- **Channels - Managing**
- **Windows - Managing**

**Note:** You may experience a significant decrease in computer processing speed with combinations of the following: increased number of points, number of traces, and calibration error terms (full 2-port or 3-port). If this becomes a problem, you can increase the amount of RAM with PNA **Option 022**. To monitor the amount of PNA memory usage, press **Ctrl Alt Delete**, select **Task Manager**, then click on the **Performance** tab.

**Other Quick Start topics**

**Traces** are a series of measured data points. There is no theoretical limit to the number of traces. However, the practical limit is the maximum number of windows * the maximum number of traces per window (8).

In addition, one memory trace can be stored and displayed for every data trace. **Learn more about Math / Memory traces.**

Trace settings affect the presentation and mathematical operations of the measured data. The following are Trace settings.

- **Parameter**
- **Format and Scale**
- **Smoothing**
- **Correction ON / OFF**
- **Electrical Delay**
- **Phase Offset**
- **Trace Math**
- **Markers**
- **Time Domain** (Opt 010)

**Managing Traces**

- **How to Select** a trace
- **How to Delete** a trace
- **How to Move a trace**
- **How to Maximize a trace**
- **How to Create a new trace**
- **How to Change the trace parameter.**
- **How to display a custom trace title.**

### How to Select a Trace

A trace must be selected (active) before its trace settings can be changed.

#### How to know which trace is Active?

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press TRACE repeatedly</td>
<td>1. Click the Trace Status button.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>For <strong>PNA-X</strong> models</th>
<th><strong>Programming Commands</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. For Traces 1-4, press the corresponding Hard Key</td>
<td>1. Click the Trace Status label or trace.</td>
</tr>
<tr>
<td>2. For other trace numbers, press TRACES</td>
<td></td>
</tr>
<tr>
<td>3. then [Select Traces]</td>
<td></td>
</tr>
<tr>
<td>4. Select a trace number in the Entry toolbar</td>
<td></td>
</tr>
</tbody>
</table>
### How to Delete a Trace

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press TRACE</td>
<td>1. Right-click the Trace Status button, then click Delete.</td>
</tr>
<tr>
<td><strong>For PNA-X models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. For Traces 1-4, press the corresponding Hard Key</td>
<td>1. Right-click the Trace Status label, then click Delete.</td>
</tr>
<tr>
<td>2. For other trace numbers, press TRACES</td>
<td></td>
</tr>
<tr>
<td>3. then [Select Traces]</td>
<td></td>
</tr>
<tr>
<td>4. Select a trace number in the Entry toolbar.</td>
<td></td>
</tr>
</tbody>
</table>

### How to Move a trace to a different window

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Not available</td>
<td>1. Not available</td>
</tr>
<tr>
<td><strong>For PNA-X models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Select the trace to move.</td>
<td>1. Right-click the Trace Status label, then click Move Trace.</td>
</tr>
<tr>
<td>2. Press TRACES</td>
<td></td>
</tr>
<tr>
<td>3. then [Move Trace]</td>
<td></td>
</tr>
<tr>
<td>4. Select a window number in the following dialog.</td>
<td></td>
</tr>
</tbody>
</table>

PNA-X ONLY
This dialog is launched by clicking **Trace/Chan**, then **Delete Trace**

The Select Trace dialog is launched by clicking **Trace/Chan**, then **Select Trace**

<table>
<thead>
<tr>
<th>Select, Delete, Move Traces dialog box help</th>
</tr>
</thead>
<tbody>
<tr>
<td>Both the <strong>Select Trace</strong> and <strong>Delete Trace</strong> dialogs work the same.</td>
</tr>
<tr>
<td>Select a trace, then click <strong>OK</strong>.</td>
</tr>
<tr>
<td>Only ONE trace can be Selected or Deleted.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
</tr>
<tr>
<td>To EASILY select a trace, click the <strong>Trace Status</strong> label.</td>
</tr>
<tr>
<td>To EASILY delete a trace, right-click the Trace Status label, then click <strong>Delete</strong>.</td>
</tr>
</tbody>
</table>

**Trace Max - PNA-X ONLY**

Makes the active trace the ONLY trace on the display. All other traces are hidden.

**How to do Trace Max**

- Select **Trace**, then **Trace Max**.
- With Trace Max ON, select a different trace from the **Traces** softkeys to make that trace visible.
- To make all traces visible again select **Trace Max OFF**

**Trace Title**

A Trace Title overwrites the Measurement Parameter in the **Trace Status** area, the **Status Bar**, and **hardcopy prints**.

- This title has priority over **Equation Editor** titles.
- The practical limit is about 70 characters if there is only one trace.
- Spaces are accepted but not displayed; use underscores.
- The title is annotated as follows:
How to enter a Trace Title

1. Click the Trace Status label to select a trace.

2. Click Trace/Chan, then Trace, then Trace Title.

3. Click Enable, then type the trace title. Click Keyboard to type with a mouse.

4. To remove the trace title, clear the Enable checkbox, or delete the text from the dialog entry.

Channels contain traces. The PNA can have up to 32 independent channels.

Channel settings determine how the trace data is measured. All traces that are assigned to a channel share the same channel settings. A channel must be selected (active) to modify its settings. To select a channel, click the Trace Status button of a Trace in that channel. The following are channel settings:

- Frequency range
- Power level
- Calibration
- IF Bandwidth
- Number of Points
- Sweep Settings
- Average
- Trigger (some settings are global)

Managing Channels

How to Select a Channel

A channel must be selected (active) before its settings can be changed.

To make a channel active, select a trace in that channel.
How to Turn ON or OFF a Channel

Click **Trace/Chan**, then **Channel**, then **Turn On / Off Channel**.

![Turn Off Channel](image)

**Turn ON | OFF Channel** dialog box help

Both the Turn **ON** and Turn **OFF** dialogs work the same.
Select a channel, then click **OK**. Only ONE channel can be selected.

When turning ON a channel, the new channel is always the Standard **Measurement Class** with an S11 trace.

**Note:** To create more than one trace in a new channel, click Trace, then New Trace.

**Windows** are used for viewing traces.

- The PNA can show an **UNLIMITED** number of windows on the screen (16 windows previous to PNA release 6.2) with the following limitations:
  - The COM property **MaximumNumberOfWindows** returns 1000 ('unlimited' is not a number).
  - The **SCPI status register** can only track the status of up to 576 traces.

- Each window can contain up to **8 traces** (4 traces previous to PNA release 5.2).
- Windows are completely independent of channels.
- Learn to **create and manage windows**.

The following is a window containing two traces. Both traces use the same channel 1 settings as indicated by the annotation at the bottom of the window.
PNA-X shows the window number in the lower-left corner of the window. The following shows window 5.

### Managing Windows

**How to make various window settings**

New, Close, Tile, Cascade, Minimize, Maximize

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>Window</strong></td>
<td>1. Click <strong>Window</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>For PNA-X models</strong></th>
<th><strong>Programming Commands</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>RESPONSE</strong></td>
<td>1. Click <strong>Response</strong></td>
</tr>
<tr>
<td>2. then <strong>[Display]</strong></td>
<td>2. then <strong>Display</strong></td>
</tr>
<tr>
<td>3. then <strong>[Windows]</strong></td>
<td>3. then <strong>Windows</strong></td>
</tr>
</tbody>
</table>
**Close Window** dialog box help

Select a window, then click **OK**. The remaining windows are tiled.

Only ONE window can be selected.

Traces contained in a closed window are deleted.

**Note:** To EASILY close a window, click the X in the upper right corner of a window. The X is only visible when **Title Bars** are enabled. The remaining windows are NOT tiled.

See [Customize the PNA screen](#) to learn how to make other window settings

Last modified:

- 9/19/06  MQ Modified for unlimited number of windows
Basic Measurement Sequence

The following process can be used to setup all PNA measurements:

**Step 1. Set Up Measurements**
Reset the analyzer, create a measurement state, and adjust the display.

**Step 2. Optimize Measurements**
Improve measurement accuracy and throughput using techniques and functions.

**Step 3. Perform a Measurement Calibration**
Reduce the measurement errors by performing a calibration.

**Step 4. Analyze Data**
Analyze the measurement results using markers, math operations, and limit tests.

**Step 5. Print, Save or Recall Data**
Save or print the measurement data.
Frequency Blanking

Note: To learn how to erase memory before moving your PNA out of a secure area, see http://na.tm.agilent.com/pna/security.html.

For security reasons, you can prevent frequency information from appearing on the PNA screen and printouts.

How to set Frequency Blanking

Using front-panel HARDKEY [softkey] buttons

Using a mouse with PNA Menus

For **N5230A** and **E836xA/B** models

1. Navigate using **MENU/DIALOG**

For **PNA-X** and **'C'** models

1. Press **UTILITY**
2. then **[System]**
3. then **[Security]**

Security Setting

Security level of None allows frequency information to be displayed on PNA application dialogs and printouts.

Options:

- None

Buttons:

- OK
- Cancel
- Help
Security Setting dialog box help

**None**  All frequency information is displayed on the screen and printouts.

**Low security level**
Frequency information is blanked from the following:

- Display annotation
- Calibration properties
- All tables
- All toolbars
- All printouts

To re-display frequency information, revisit this dialog box and select None.

**High security level**
Low security level settings PLUS:

- **GPIB console** is inactive

To re-display frequency information, perform an instrument preset, or recall an instrument state with security level of None.

**Extra security level**
High security level settings PLUS:

- All ASCII data saving capability (.snp, .prn, .cti) is DISABLED.

To re-display frequency information, perform an instrument preset, or recall an instrument state with security level of None.

**For ALL security levels:**
Frequency information is **NOT** blanked from the following:

- The Frequency Converter Application (opt 083) dialog box information or printouts.
- Service Adjustment Programs
- Your remote COM or SCPI programs.

---

**Last Modified:**

17-Jul-2007  Added Extra setting
Internal Second Source

The following PNA models include an internal second source.

<table>
<thead>
<tr>
<th>Model (click to see block diagram)</th>
<th>Total # of Ports</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5230A Opt 146</td>
<td>4</td>
<td>300 KHz - 13.5 GHz</td>
</tr>
<tr>
<td>N5230A Opt 246</td>
<td>4</td>
<td>300 KHz - 20 GHz</td>
</tr>
<tr>
<td>N5242A Opt 224</td>
<td>2*</td>
<td>10 MHz to 26.5 GHz</td>
</tr>
<tr>
<td>N5242A Opt 400</td>
<td>4</td>
<td>10 MHz to 26.5 GHz</td>
</tr>
</tbody>
</table>

How to use the second source

- Set frequency using the Frequency Offset Opt 080 dialog.
- Set power using the Advanced Power dialog.
- Source power calibration of the second source is performed as usual.
- Using FCA, click the LO button to set frequency and power.
- The specifications of the second source are the same as source 1.

Benefits / Uses of the second source

- Up to five times faster than stepping an external source.
- Measure Mixers with internal swept or fixed LO.
- Measure TOI or Intermodulation distortion.

Internal Second Source Restrictions

Source 1 and Source 2 are available at specific ports as follows:

**N5230A (PNA-L) models:**

- Source 1 power available at Port 1 OR Port 2; NOT at both ports simultaneously.
- Source 2 power available at Port 3 OR Port 4; NOT at both ports simultaneously.

**N5242A Opt 224 (PNA-X 2-port model):**
- Source 1 power available at Port 1 OR Port 2; NOT at both ports simultaneously.
- Source 2 (SRC 2) power available at Out 1 AND Out 2; BOTH ports simultaneously.

**N5242A Opt 400, 419, 423 (PNA-X 4-port models):**

- Source 1 power available at Port 1 OR Port 2; NOT at both ports simultaneously.
- Source 2 power available at Port 3 AND Port 4; BOTH ports simultaneously.
- Opt 423 ONLY - Source 2 power is available at Port 4 AND either Port 3 OR Port 1 (through the combiner as "Port 1 Src2"). [See block diagram for N5242A Opt 423](#)

**Remotely Accessing the Internal Second Source**

See

---

**Last modified:**

- 23-Jul-2007 Added remote section
- 1123-Jul-2007 MX Added PNA-X models
- 10/02/06 MQQ New topic
Networking and Connecting the PNA

The PNA as a PC

- **PNA User Accounts and Passwords**
- **Drive Mapping**
- **Connecting the PNA to a PC**
- **Easy versus Secure Configuration**
- **Changing Network Client**
- **Using VNC to Control the PNA User Interface**

GPIB / COM Programming

- **Configure for COM/DCOM Programming**
- **82357A USB to GPIB Interface**
- **The PNA as GPIB System Controller**
- **How to Configure for GPIB, SCPI, and SICL**

Controlling External Devices

- **E5091 TestSet Control**
- **External Testset Control**
- **Interface Control Feature**
- **TestSetIO Connector**
- **Handler IO Connector**
- **AuxIO Connector**
## PNA Preferences

The following is a list of PNA preferences. Most of these are set using SCPI or COM commands. SCPI commands can be easily set from the PNA front panel. For more information, click the links below.

### Calibration

<table>
<thead>
<tr>
<th>Preference</th>
<th>UI Setting</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto-save User Cal Set</td>
<td>None</td>
<td>SENS:CORR:PREF:CSET:SAVE</td>
<td>None</td>
</tr>
<tr>
<td>For Guided Cal, set external trigger.</td>
<td>None</td>
<td>SENS:CORR:PREF:TRIG:FREE</td>
<td>PreferInternalTriggerOnChannelSingle</td>
</tr>
<tr>
<td>For Unguided Cal, set external trigger.</td>
<td>None</td>
<td>SENS:CORR:PREF:TRIG:FREE</td>
<td>PreferInternalTriggerOnUnguidedCal</td>
</tr>
<tr>
<td>Sets behavior for simulated cal</td>
<td>None</td>
<td>SENS:CORR:PREF:SIMCal</td>
<td>None</td>
</tr>
<tr>
<td>Show or not, the first 'Method' Page of the Cal Wizard.</td>
<td>Cal Preferences</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Set and order default Cal Types</td>
<td>Cal Preferences</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Specify ECal port mapping when orientation is OFF</td>
<td>ECal Wizard</td>
<td>SENS:CORR:PREF:ECAL:PMAP</td>
<td>ECALPortMapEx</td>
</tr>
</tbody>
</table>

### File Save

272
<table>
<thead>
<tr>
<th>Preference</th>
<th>UI Setting</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the contents of subsequent citifile saves.</td>
<td>Data Define</td>
<td>MMEM:STOR:TRAC:CONT:CIT</td>
<td>CitiContents</td>
</tr>
<tr>
<td>Specifies the format of subsequent citifile saves.</td>
<td>Data Define</td>
<td>MMEM:STOR:TRAC:FORM:CIT</td>
<td>CitiFormat</td>
</tr>
</tbody>
</table>

**Measurements**

<table>
<thead>
<tr>
<th>Preference</th>
<th>UI Setting</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mathematically offset for receiver attenuation.</td>
<td>None</td>
<td>SYST:PREF:ITEM:OFFS:RCV</td>
<td>OffsetReceiverAttenuator</td>
</tr>
<tr>
<td>Mathematically offset for source attenuation.</td>
<td>None</td>
<td>SYST:PREF:ITEM:OFFS:SRC</td>
<td>OffsetSourceAttenuator</td>
</tr>
<tr>
<td>Turn RF power ON or OFF during a frequency sweep retrace.</td>
<td>None</td>
<td>SYST:PREF:ITEM:RETR:POW</td>
<td>PowerOnDuringRetraceMode</td>
</tr>
<tr>
<td>For power sweep, maintain source power at the start or stop power level.</td>
<td>None</td>
<td>SYST:PREF:ITEM:PSRT</td>
<td>PowerSweepRetracePowerMode</td>
</tr>
<tr>
<td>Sets the External Trigger OUT behavior to have either Global or Channel scope.</td>
<td>None</td>
<td>TRIG:PREF:AIGL</td>
<td>AuxTriggerIsGlobal</td>
</tr>
</tbody>
</table>

**Errors**

<table>
<thead>
<tr>
<th>Preference</th>
<th>UI Setting</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report source unleveled events as errors?</td>
<td>None</td>
<td>SYST:ERR:REP:SUNL</td>
<td>EnableSourceUnleveledEvents</td>
</tr>
<tr>
<td>Display Error Messages?</td>
<td>Error Preferences</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
Using VNC to Control the PNA User Interface

VNC (Virtual Network Computing) allows you to control the User Interface of a PNA from any PC. The PNA display appears on the connected PC display. Mouse and keyboard control can occur from both the PNA and PC, although not simultaneously.

Both the PNA and PC must be connected to the internet. The responsiveness of the PNA while using VNC is dependent of the speed of your internet connection.

Every PNA is shipped with VNC installed. However, you must download and install the VNC software onto the PC from http://www.realvnc.com/.

Once installed, the following procedure will help you configure VNC to view and control the PNA application from your PC.

On the PNA, run VNC Server

To do this:

1. Click View, then Minimize Application.

2. Click Start, Programs, RealVNC, Run VNC Server.

   - The first time you run VNC Server, you will set a password to control access from remote PCs.

   - To automatically start VNC when the PNA computer boots, drag a Run VNC Server shortcut to your User "startup" folder. The following is the Administrator folder: C:\Documents and Settings\Administrator\Start Menu\Programs\Startup

On the PC, run VNC Viewer

To do this:

1. Download (http://www.realvnc.com/) and install VNC on the PC.

2. From the PC Desktop, click Start, Programs, RealVNC, Run VNC Viewer

3. When prompted for the Hostname, type the full computer name or IP address of the PNA.

4. When prompted for the password, type the password you set when configuring VNC on the PNA.
Using Help

This topic discusses the following:

- **PNA Documentation**
- **Printing Help**
- **Copying Help to your PC**
- **Launching Help**
- **Navigating Help**
- **Help Languages**
- **Glossary**
- **Dialog Boxes**
- **About Network Analyzer**
- **Documentation Warranty**
- **Suggestions Please**

### Other Quick Start Topics

**PNA Documentation**

This Help file, which is embedded in the PNA, is the *Users Guide and Programming Manual for the PNA*. The help file is automatically updated on the PNA when firmware is updated. Only the PNA [Installation and Quick Start Guide](http://na.tm.agilent.com/pna/help/index.html) is shipped with new PNA instruments.

Hardcopy manuals are no longer available for purchase with the PNA.

All PNA documentation, including the latest online Web Help version of this Help file, and a printable .PDF version of the Help file, are available at [http://na.tm.agilent.com/pna/help/index.html](http://na.tm.agilent.com/pna/help/index.html).

**Printing Help**


**Copying Help to your PC**
With the Help system on your PC, you can read about the PNA while away from it. You can also Copy and Paste programming code from this Help system directly into your programming environment.

The Help file is located on your PNA hard-drive at `C:\ Winnt\ Help\ PNAHelp.chm`. If both the PNA and PC are connected to LAN, you can map a drive and copy the file directly.

The Help file can also be downloaded from `http://na.tm.agilent.com/pna/help/index.html`.

**Launching Help**

The Help system can be launched in three ways:

1. From the front panel Help button.
2. From the Help drop-down menu
3. From Dialog Box Help

**Navigating Help**

The Help Window contains 3 panes (regions):

1. **Toolbar** Pane
2. **Topic** Pane
3. **Navigation** Pane

---

**Toolbar Pane**

The Toolbar is at the top of all Help windows. It allows you to resize the window, browse and print the selected topic.
1. Hide or show the navigation pane
2. Locate the topic in the table of contents
3. Back to topic visited previously
4. Forward again if Back was clicked
5. Go to the Home page.
6. Print the topic pane.

Navigation Pane
Click the following tabs in the Navigation Pane to access information in the Help system:

- **Table of Contents Tab**
- **Index Tab**
- **Search Tab**
- **Favorites Tab**

(Table of Contents Tab)
1. Click tab to select Table of Contents.
2. Click a book to access related topics.
3. Click to display a topic.
4. Right click to access menu.
5. Click to display specifications
6. Click to display glossary

**Index Tab**

The index tab allows you to type a keyword and go to only the most applicable topics.

1. Click tab to select index.
2. Type keyword to find topics of interest.
3. View suggested topics. (Double-click to display topic.)
4. Click to display topic.

**Search Tab**

**TIP:** To Search any topic for a keyword, press **Ctrl** and **F**.

The following rules apply for using full-text search:

- Searches are not case-sensitive.
- You can search for any combination of letters (a-z) and numbers (0-9).
- Punctuation marks (period, colon, semicolon, comma, and hyphen) are ignored during a search.
- You can group the words of your search using double quotes or parentheses. Examples: "response calibration" or (response calibration). This requirement makes it impossible to search for quotation marks.
- Use Wildcard expressions:
  - To search for one undefined character use a question mark (?). For example, searching for cal? will find **calc** and **calf**.
  - To search for more than one undefined character use an asterisk (*). Searching for Cal* will find **calibration** and **calculate**.
- Use Boolean operators to define a relationship between two or more search words.
<table>
<thead>
<tr>
<th>Search for</th>
<th>Example</th>
<th>Results will show topics containing:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two words in the same topic</td>
<td>response AND</td>
<td>Both the words &quot;response&quot; and &quot;calibration&quot;.</td>
</tr>
<tr>
<td></td>
<td>calibration</td>
<td></td>
</tr>
<tr>
<td>Either of two words in a topic</td>
<td>response OR</td>
<td>Either the word &quot;response&quot; or the word &quot;calibration&quot; or both.</td>
</tr>
<tr>
<td></td>
<td>calibration</td>
<td></td>
</tr>
<tr>
<td>The first word without the second word in a</td>
<td>response NOT</td>
<td>The word &quot;response&quot; but not the word &quot;calibration&quot;.</td>
</tr>
<tr>
<td>topic</td>
<td>calibration</td>
<td></td>
</tr>
<tr>
<td>Both words in the same topic, close together.</td>
<td>response NEAR</td>
<td>The word &quot;response&quot; within eight words of the word &quot;calibration&quot;.</td>
</tr>
<tr>
<td></td>
<td>calibration</td>
<td></td>
</tr>
</tbody>
</table>

**Favorites Tab**

The favorites tab allows you to store (bookmark) the topics you refer to most often so that they can be recalled easily.

1. Click tab to view stored topics in Favorites.
2. Remove selected topic.
3. Display selected topic.
4. Add (store) current topic.

**Topic Pane**

The Topic pane allows you to view the contents of the selected topic.
Help Languages
Beginning with PNA Rev A.08.00, PNA Help is offered in English ONLY.

Glossary
The Glossary holds definitions of words, in alphabetical order.

Note: Click on a word in green text throughout Help to see the glossary definition.

Dialog Boxes
About Network Analyzer

To learn the following about the PNA, click Help, then About Network Analyzer:

- Model number (see list of PNA models)
- Frequency range
- Serial number
- Installed options
- Application Code (firmware) version
- Version of hard drive in the analyzer

Documentation Warranty

THE MATERIAL CONTAINED IN THIS DOCUMENT IS PROVIDED “AS IS,” AND IS SUBJECT TO BEING CHANGED, WITHOUT NOTICE, IN FUTURE EDITIONS. FURTHER, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, AGILENT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED WITH REGARD TO THIS MANUAL AND ANY INFORMATION CONTAINED HEREIN, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. AGILENT SHALL NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR ANY INFORMATION CONTAINED HEREIN. SHOULD AGILENT AND THE USER HAVE A SEPARATE WRITTEN AGREEMENT WITH WARRANTY TERMS COVERING THE MATERIAL IN THIS DOCUMENT THAT CONFLICT WITH THESE TERMS, THE WARRANTY TERMS IN THE SEPARATE AGREEMENT WILL CONTROL.

Suggestions Please!
Please let us know about your experience using PNA Help. Send your comments to: pna_help@am.exch.agilent.com. Comment about any aspect of the help system. Here are a few areas that you might consider:

- Does anything appear to be broken?
- Did you find what you were looking for?
- Was the information you found helpful?
- Any suggestions as to how we can improve the help system?

Your comments go directly to the help system authors. For help with technical questions, please refer to Technical Support.
Preset the PNA

When you Preset the PNA, it is set to known, or preset conditions. You can use the factory default preset conditions, or define your own User Preset conditions.

- **Preset (Default) Conditions**
- **User Preset Conditions**

### Preset Default Conditions

#### How to Preset the PNA

**Tip:** Press the Preset button to start the PNA application if it is not already running.

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>For N5230A and E836xA/B models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press Preset</td>
<td>1. Click System</td>
</tr>
<tr>
<td></td>
<td>2. then Preset</td>
</tr>
<tr>
<td>For PNA-X and 'C' models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press Preset</td>
<td>1. Click Utility</td>
</tr>
<tr>
<td></td>
<td>2. then Preset</td>
</tr>
</tbody>
</table>

Click to view the **factory preset conditions**.

- **Frequency Settings**
- **Power Settings**
- **Sweep Settings**
- **Segment Sweep Settings**
- **Trigger Settings**
- **Display Settings**
- **Response Settings**
- **Calibration Settings**
- **Marker Settings**
- **Limit Test Settings**
- **Time Domain Settings (Option 010)**
- **Global Display Settings**

### Frequency Settings:

<table>
<thead>
<tr>
<th>Measurement Parameter</th>
<th>S11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Frequency</td>
<td>Minimum frequency of the PNA</td>
</tr>
<tr>
<td>Stop Frequency</td>
<td>Maximum frequency of the PNA</td>
</tr>
<tr>
<td>CW Frequency</td>
<td>1 GHz</td>
</tr>
</tbody>
</table>

See the [PNA configurations](#) for the minimum and maximum frequency of your PNA.

### Power Settings:

<table>
<thead>
<tr>
<th>Test Port Power</th>
<th>0 dBm for E8356/7/8A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 dBm for E8801/2/3A</td>
</tr>
<tr>
<td></td>
<td>0 dBm for N3381/2/3A</td>
</tr>
<tr>
<td></td>
<td>-5 dBm for N5230A - 20 GHz</td>
</tr>
<tr>
<td></td>
<td>-10 dBm for N5230 - 40 GHz</td>
</tr>
<tr>
<td></td>
<td>-15 dBm for N5230 - 50 GHz</td>
</tr>
<tr>
<td></td>
<td>-12 dBm for E8362/3/4 A or B, standard</td>
</tr>
<tr>
<td></td>
<td>-15 dBm for E8361A</td>
</tr>
<tr>
<td></td>
<td>-17 dBm for E8362/3/4 A or B with option UNL or 014</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Power</th>
<th>On</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port Power Coupled</td>
<td>On</td>
</tr>
<tr>
<td>Auto Attenuation</td>
<td>On</td>
</tr>
<tr>
<td>Attenuator Value</td>
<td>0 dB</td>
</tr>
<tr>
<td>Power Slope</td>
<td>Off</td>
</tr>
<tr>
<td>Slope Value</td>
<td>0 dB/GHz</td>
</tr>
</tbody>
</table>

285
### Sweep Settings:

<table>
<thead>
<tr>
<th>Type</th>
<th>Linear Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode</td>
<td>Continuous</td>
</tr>
<tr>
<td>Generation</td>
<td>Analog</td>
</tr>
<tr>
<td>Auto Sweep Time</td>
<td>On</td>
</tr>
<tr>
<td>Number of Points</td>
<td>201</td>
</tr>
</tbody>
</table>

### Segment Sweep Settings:

<table>
<thead>
<tr>
<th>Active Segments</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Frequency</td>
<td>PNA start frequency</td>
</tr>
</tbody>
</table>
| Stop Frequency  | 1 MHz for E8356/7/8A  
                  | 1 MHz for E8801/2/3A  
                  | 1 MHz for N3381/2/3A  
                  | 1 GHz for E836xA/B    |
| Number of Points| 21 |
| Power           | PNA preset test port power |
| IF Bandwidth    | 50 KHz for N5230A  
                  | 35 kHz for all other models |
| Reduce IF BW at Low Frequencies | ON |
| Dwell Time      | 0  |

### Trigger Settings

<table>
<thead>
<tr>
<th>Source</th>
<th>Internal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode</td>
<td>Sweep</td>
</tr>
</tbody>
</table>

### Display Settings:

<table>
<thead>
<tr>
<th>Format</th>
<th>Log Mag</th>
</tr>
</thead>
</table>

These settings apply for formats when selected:
<table>
<thead>
<tr>
<th>Format</th>
<th>Scale</th>
<th>Reference Position</th>
<th>Reference Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log Mag</td>
<td>10 dB/</td>
<td>5</td>
<td>0 dB</td>
</tr>
<tr>
<td>Phase</td>
<td>45 degrees/</td>
<td>5</td>
<td>0 degrees</td>
</tr>
<tr>
<td>Group Delay</td>
<td>10 nsec/</td>
<td>5</td>
<td>0 s</td>
</tr>
<tr>
<td>Linear Mag</td>
<td>100 munits/</td>
<td>0</td>
<td>0 units</td>
</tr>
<tr>
<td>SWR</td>
<td>1 unit/</td>
<td>0</td>
<td>1 unit</td>
</tr>
<tr>
<td>Real</td>
<td>2 units/</td>
<td>5</td>
<td>0 units</td>
</tr>
<tr>
<td>Imaginary</td>
<td>2 units/</td>
<td>5</td>
<td>0 units</td>
</tr>
<tr>
<td>Polar</td>
<td>1 unit/</td>
<td>n/a</td>
<td>1 unit</td>
</tr>
<tr>
<td>Smith Chart</td>
<td>1 unit/</td>
<td>n/a</td>
<td>1 unit</td>
</tr>
</tbody>
</table>

**Response Settings:**

- **Channel Number**: 1
- **IF Bandwidth**: 50 KHz for N5230A, 35 kHz for all other models
- **Averaging**: Off
- **Averaging Factor**: 1
- **Smoothing**: Off
- **Smoothing Factor**: 1% of span
- **Electrical Delay**: 0 s
- **Velocity Factor**: 1.0
- **Phase Offset**: 0 degrees
- **Math/Memory Trace View**: Data
## Calibration Settings:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correction State</td>
<td>Off</td>
</tr>
<tr>
<td>Interpolation State</td>
<td>On</td>
</tr>
<tr>
<td>Calibration Type</td>
<td>None</td>
</tr>
<tr>
<td>Cal Kit Number</td>
<td>Current Cal Kit Number</td>
</tr>
<tr>
<td>System Z0</td>
<td>50 ohms</td>
</tr>
<tr>
<td>Port Extensions State</td>
<td>Off</td>
</tr>
<tr>
<td>Port Ext. Values</td>
<td>0</td>
</tr>
<tr>
<td>Input A, B</td>
<td></td>
</tr>
<tr>
<td>Port 1, 2</td>
<td></td>
</tr>
</tbody>
</table>

## Marker Settings:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Frequency</td>
<td>Current Center Frequency</td>
</tr>
<tr>
<td>Reference</td>
<td>None</td>
</tr>
<tr>
<td>Interpolation</td>
<td>On</td>
</tr>
<tr>
<td>Format</td>
<td>Trace Default</td>
</tr>
<tr>
<td>Type</td>
<td>Normal</td>
</tr>
<tr>
<td>Function</td>
<td>Max Value</td>
</tr>
<tr>
<td>Domain</td>
<td>Full Span</td>
</tr>
<tr>
<td>Table</td>
<td>Empty</td>
</tr>
<tr>
<td>Coupling</td>
<td>Always uncoupled</td>
</tr>
</tbody>
</table>

## Limit Test Settings:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limit Testing</td>
<td>Off</td>
</tr>
<tr>
<td>Line Display</td>
<td>ON</td>
</tr>
<tr>
<td>Sound on Fail</td>
<td>Off</td>
</tr>
</tbody>
</table>
### Limit List Settings:

<table>
<thead>
<tr>
<th>Type (OFF, MAX, MIN)</th>
<th>OFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Begin Stimulus</td>
<td>0</td>
</tr>
<tr>
<td>End Stimulus</td>
<td>0</td>
</tr>
<tr>
<td>Begin Response</td>
<td>0</td>
</tr>
<tr>
<td>End Response</td>
<td>0</td>
</tr>
</tbody>
</table>

### Time Domain Settings:

<table>
<thead>
<tr>
<th>Transform State</th>
<th>Off</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transform Mode</td>
<td>Band Pass</td>
</tr>
<tr>
<td>Transform Start</td>
<td>-10 ns</td>
</tr>
<tr>
<td>Transform Stop</td>
<td>10 ns</td>
</tr>
<tr>
<td>Window</td>
<td>6.0 (Kaiser-Bessel factor)</td>
</tr>
<tr>
<td>Gating State</td>
<td>Off</td>
</tr>
<tr>
<td>Gating Start</td>
<td>-10 ns</td>
</tr>
<tr>
<td>Gating Stop</td>
<td>10 ns</td>
</tr>
<tr>
<td>Gate Type</td>
<td>Band Pass</td>
</tr>
<tr>
<td>Gate Shape</td>
<td>Normal</td>
</tr>
</tbody>
</table>

### Global Display Settings:

<table>
<thead>
<tr>
<th>Trace Status</th>
<th>On</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency/ Stimulus</td>
<td>Off</td>
</tr>
<tr>
<td>Marker Readout</td>
<td>On (when a marker is activated)</td>
</tr>
<tr>
<td>Toolbars Shown</td>
<td>Active Entry</td>
</tr>
<tr>
<td>Status Bar State</td>
<td>ON</td>
</tr>
</tbody>
</table>

### User Preset Conditions

The analyzer can be **preset** to either **factory default** conditions or **User Preset** conditions.
How to set User Preset

Using front-panel HARDKEY [softkey] buttons

PNA Menu using a mouse

For **N5230A** and **E836xA/B** models

1. Navigate using **MENU/DIALOG**

For **PNA-X** and **’C’ models**

1. Press **SAVE**
2. then **[User Preset]**

**User Preset** dialog box help

With a User Preset saved and enabled, when the PNA is Preset, the User Preset settings are recalled instead of the factory default settings. Calibration data is NOT recalled with a User Preset. [Learn more about instrument state settings.]

**User Preset Enable**
Check - The PNA is preset to **User Preset** conditions when the Preset button is pressed.
Clear - The PNA is preset to **Default** conditions when the Preset button is pressed.

**Save current state as User Preset** Click to store the current instrument state as the User Preset conditions. File is stored as C:/ Program Files/ Agilent/ Network Analyzer/ Documents/ UserPreset.sta.

**Load existing file as User Preset** Click to retrieve an instrument state to be used as the User Preset conditions.
Last modified:

9/27/06  MX Added UI

9/12/06  Added link to programming commands
Measurement Parameters

This topic contains the following information:

- **S-Parameters** (pre-selected ratios)
- **Ratioed** (choose your own ratio)
- **Unratioed Power** (absolute power)
- **How to Select a Measurement Parameter**

Learn about Balanced Measurements

---

### S-Parameters

S-parameters (scattering parameters) are used to describe the way a device modifies a signal. For a 2-port device, there are **four S-Parameters**. The syntax for each parameter is described by the following:

\[ S_{\text{out} - \text{in}} \]

- \( \text{out} \) = PNA port number where the device signal output is measured (receiver)
- \( \text{in} \) = PNA port number where the signal is applied (incident) to the device (source)

Move the mouse over each S-parameter to see the signal flow:

For two-port devices:

- When the source goes into port 1, the measurement is said to be in the **forward** direction.
• When the source goes into port 2, the measurement is said to be in the reverse direction.

The analyzer automatically switches the source and receiver to make a forward or reverse measurement. Therefore, the analyzer can measure all four S-parameters for a two-port device with a single connection. See the block diagram (including receivers) of your PNA.

**Common Measurements with S-Parameters**

<table>
<thead>
<tr>
<th>Reflection Measurements (S11 and S22)</th>
<th>Transmission Measurements (S21 and S12)</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Return loss</td>
<td>• Insertion loss</td>
</tr>
<tr>
<td>• Standing wave ratio (SWR)</td>
<td>• Transmission coefficient</td>
</tr>
<tr>
<td>• Reflection coefficient</td>
<td>• Gain/Loss</td>
</tr>
<tr>
<td>• Impedance</td>
<td>• Group delay</td>
</tr>
<tr>
<td>• S11, S22</td>
<td>• Deviation from linear phase</td>
</tr>
<tr>
<td></td>
<td>• Electrical delay</td>
</tr>
<tr>
<td></td>
<td>• S21, S12</td>
</tr>
</tbody>
</table>

**Receiver Measurements**

A 2-port PNA typically has four receivers: A, B, R1, and R2. Your PNA may not have 2 reference and 2 test port receivers. See the block diagram of your PNA.

• R1 and R2 are reference receivers. They measure the PNA source signal as it leaves the PNA and is incident on the DUT.
  • R1 measures the signal out of Port 1
  • R2 measures the signal out of Port 2.

• A and B are test port receivers. They measure the signal out (or reflecting off) of the DUT.
  • A measures the signal into PNA Port 1
  • B measures the signal into PNA Port 2

You can specify measurements using one or two of the available receivers.

**Note:** Beginning with PNA Rev. 7.22, you can use the internal ADC (Analog-Digital Converters) as measurement receivers. Learn more.

**Ratioed Measurements**

Ratioed measurements allow you to choose your own ratio of any two receivers that are available in your PNA. S-parameters are actually predefined ratio measurements. For example S11 is A/R1.

The following are common uses of ratioed measurements:
Comparing the phase between two paths of a device. An example could be something simple like a power splitter or more complicated like a dual-channel receiver.

Measurements that require a higher dynamic range than the analyzer provides with S-parameters.

Your PNA MAY have front-panel jumper cables that go directly to measurement receivers. Learn about the front-panel jumpers on your PNA.

**Unratioed (Absolute Power) Measurements**

The unratioed power parameter allows you to look at the absolute power going into any of the measurement receivers that are available on your PNA.

The reference receivers are internally configured to measure the source power for a specific PNA port. Performing an absolute power measurement of a reference receiver using a different source port will measure very little power unless the front panel jumpers are removed and signal is applied directly to the receiver. An example of this would be an R1 measurement using port 2 as the source.

- Measuring phase using a single receiver yields meaningless data. Phase measurements must be a comparison of two signals.
- Averaging for Unratioed parameters is computed differently from ratioed parameters.

**How to create a NEW trace**

**PNA-X**

The only measurements that can be created are those in the same measurement class as is currently assigned to the active channel. To create a measurement other than these, first assign the appropriate measurement class to a new or existing channel. Learn how.

After that is done...

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>TRACE</strong></td>
<td>1. Click <strong>Trace</strong></td>
</tr>
<tr>
<td>2. then [Active Entry keys]</td>
<td>2. then <strong>New Trace</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>For PNA-X and 'C' models</strong></th>
<th><strong>Programming Commands</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>TRACE 1, 2, 3, OR 4</strong></td>
<td>1. Click <strong>Trace/Chan</strong></td>
</tr>
<tr>
<td></td>
<td>2. then <strong>New Trace</strong></td>
</tr>
</tbody>
</table>
## How to CHANGE the active trace

### For N5230A and E836x A/B models

1. Press **Measure**
2. then [Active Entry keys]

### For PNA-X and ‘C’ models

1. Press **MEAS**
2. Click **Trace** then **Measure**

### E836x and PNA-L models:

Click a tab to select the TYPE of measurement:

The tabs are populated ONLY with measurements and receivers that are available for your PNA configuration.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>S-Parameters</strong></td>
<td>Different measurements are available depending on the selected topology.</td>
</tr>
<tr>
<td><strong>Balanced</strong></td>
<td></td>
</tr>
</tbody>
</table>
Receivers

- Ratioed
- Unratioed

The internal ADCs (Analog-Digital Converters) can be used as measurement receivers. Learn more. Learn about new logical receiver notation.

Gain Compression

Noise Figure

Channel / Window Selections

New / Change Measurement dialog box help

Note: The only measurements that are available are those in the measurement class currently assigned to the active channel. Other measurements are NOT compatible.

To create a measurement other than these, first assign the appropriate measurement class to a new or existing channel. Learn how.

Click a tab to create or change measurements.

- When creating NEW measurements, you can choose more than one.
- When changing an EXISTING measurement, you can choose ONLY one.

Tabs

S-Parameter  Select a predefined ratioed measurements. Learn more about S-parameters.
Balanced  Select a balanced measurement type. (Multiport PNAs ONLY)
Change  Click to invoke the Balanced DUT Topology / Logical Port mappings dialog box. Learn more about Balanced Measurements.
Receivers  Select receivers to make Ratioed and Unratioed (absolute power) measurements. Learn more about receiver measurements.
**Logical Receiver Notation**

With PNA Rev 6.2, Ratioed and Unratioed measurements can also use **logical receiver notation** to refer to receivers. This notation makes it easy to refer to receivers with an **External Test Set** connected to the PNA. You do not need to know which physical receiver is used for each test port.

- aN - Reference receiver for logical port N
- bN - Test port receiver for logical port N

For example:

- **Ratioed**: "b12/a1" refers to the logical test port 12 receiver / the logical port 1 reference receiver.
- **Unratioed**: "b10" refers to the logical test port 10 receiver.

The old style notation (A, B, R1 and so forth) can still be used to refer to the PNA **physical** receivers. However, ratioed measurements MUST use the same notation to refer to both receivers; either the physical receiver notation (A, R1) or the logical receiver notation (aN, bN). For example, the following mixed notation is NOT allowed: A/b3 and a5/R2.

**Programming**

When entering receiver letters using programming commands, neither logical or physical receiver notation are case sensitive.

**Ratioed**  Check **Activate** to create or change a measurement. Select a receiver for the Numerator, select another receiver for the Denominator, then select a source port for the measurement.

The **Source port** is ALWAYS interpreted as a logical port number.

For convenience, the table is populated with common choices.

- Learn about **External Test Sets and Ratioed Measurements**
- Learn more about **Ratioed Measurements**.

**Unratioed**  Same as Ratioed, but select **1** as the Denominator.

- Learn More about **Unratioed Measurements**.
- See the **block diagram** of receivers in YOUR PNA.
- The internal ADCs (Analog-Digital Converters) can be used as measurement receivers. Learn more.

**Channel / Window Selections**

These selections are NOT AVAILABLE when changing an EXISTING measurement. Learn how to change a measurement.
**Channel Number**  Select the channel for the new traces.

**Create in New Window**

- Check to create new traces in a new window.
- Clear to create new traces in the active window. When the PNA traces per window limitation has been reached, no more traces are added.

**Auto-Create Windows**  Check to create new traces in as many windows as necessary. See PNA number of windows limitation.

*About Measurement Parameters* (top of page)

---

![Balanced DUT Topology / Logical Port mappings dialog box help](image)

**Balanced DUT Topology / Logical Port mappings** dialog box help

**New**  Check out the True Mode Stimulus Application being offered at [www.agilent.com/find/balanced](http://www.agilent.com/find/balanced). Create or edit DUT Topology and Logical Port Mapping.

Create or edit DUT Topology and Logical Port Mapping.

A Logical Port is a term used to describe a physical PNA test port that has been remapped to a new port number.

- Any **Two** physical PNA ports are mapped to **One Balanced** Logical port
- Any **One** PNA physical port is mapped to **One Single-Ended** Logical port

**Note:** These selections apply to ALL measurements in the channel. If the device topology is changed, any existing measurements in the channel that are incompatible with the new topology will be automatically changed to one that is compatible.

**Topology:**  Describes your DUT as you would like it tested. The following device topologies can be measured by a multiport PNA.

- **Balanced / Balanced**
  (2 logical ports - <4 actual ports>)

- **Single-ended / Balanced**
  (2 logical ports - <3 actual ports)>
• **Single-ended - Single-ended / Balanced**  
  (3 logical ports - <4 actual ports>)

These topologies can be used in the reverse (⇐⇒) direction to measure:

- **Balanced / Single-ended** topology
- **Balanced / Single-ended - Single-ended** topology

For example, to measure a **Balanced / Single-ended** topology, measure the S12 (reverse direction) of a **Single-ended / Balanced** topology.

Learn about [Logical Port mapping when using an External Test Set](#).

Learn more about [Balanced Measurements](#).

---

Last modified:

- 10/11/06  Added new UI
- 9/19/06  MQ Added logical receiver notation and Multiport meas toolbar.
- 9/12/06  Added link to programming commands
Measurement Classes

Measurement Classes are categories of measurements that can coexist on a channel.

**What are Measurement Classes**

**How to assign a Measurement Class to a Channel**

**Measurement Class Dialog Box Help**

---

See other 'Setup Measurements' topics

---

**What are Measurement Classes**

The following table shows the three Measurement Classes currently available for the PNA. Within each of these classes there are a number of measurements.

<table>
<thead>
<tr>
<th>Measurement Class</th>
<th>Examples of Measurements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard S-Parameters</td>
<td>S11, S22, R, A/R1</td>
</tr>
<tr>
<td>Scalar Mixer Measurements</td>
<td>SC21, S11, RevOPwr</td>
</tr>
<tr>
<td>Vector Mixer Measurements</td>
<td>VC21, S11, B</td>
</tr>
</tbody>
</table>

Measurement Classes are categories of measurements that can coexist on a channel. A measurement from one class can NOT reside in a channel with a measurement from another class. For example, a VC21 measurement can NOT reside in a channel that is currently hosting Scalar Mixer Measurements.

The Measurement Class dialog is accessed in the following ways:

---

**How to assign a Measurement Class to a Channel**

**Using front-panel HARDKEY [softkey] buttons**

For **N5230A** and **E836xA/B** models

1. Not Available

For **PNA-X** and **'C'** models

1. Press **MEAS**
2. then [Measurement Class]

**PNA Menu using a mouse**

1. Not Available

1. Click **Trace/Chan**
2. then **Measurement Class**
Measurements in a measurement class can NOT coexist in a channel with a measurement of a different measurement class. Select a measurement class for the active measurement channel.

Title Bar  Indicates the active channel to which the measurement class will be assigned.

Measurement Class  Choose the measurement class.

Note: The list of measurements is provided for display only. If you choose to create the measurement class in a new channel, a default measurement (usually S11) will be created. To change the measurement, click Trace, then select a new measurement.

Next  Click to invoke the following dialog. NOT available when the selected measurement class is the same as the active channel.

Choose to do the following:

- OK  - Delete the existing measurements in the active channel. Create the new measurement class, and default measurement, in that channel.

- Cancel  - Do not create the new measurement class. Leave the old measurements (and class) in that channel and return to the Measurement Class dialog box.
Frequency Range

Frequency range is the span of frequencies you specify for making a device measurement.

- **How to Set Frequency Range**
- **Zoom**
- **CW Frequencies**
- **Frequency Resolution**
- **Frequency Band Crossings**

See other 'Setup Measurements' topics

**How to set Frequency Range**

There are two ways to set the frequency range:

- A. Specify the **Start** and **Stop** frequencies of the range.
- B. Specify the **Center** frequency and desired **Span** of the range.

See the frequency ranges of all PNA models

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>START/STOP</strong> or <strong>CENTER/SPAN</strong></td>
<td>1. Click <strong>Channel</strong></td>
</tr>
<tr>
<td></td>
<td>2. then <strong>Start/Stop</strong> or <strong>Center/Span</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>For PNA-X and 'C' models</strong></th>
<th><strong>Programming Commands</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>FREQUENCY</strong></td>
<td>1. Click <strong>Stimulus</strong></td>
</tr>
<tr>
<td></td>
<td>2. then <strong>Frequency</strong></td>
</tr>
</tbody>
</table>
**Frequency Start/Stop** dialog box help

- **Start** Specifies the beginning frequency of the swept measurement range.
- **Stop** Specifies the end frequency of the swept measurement range.

**Frequency Center/Span** dialog box help

- **Center** Specifies the value at the center of the frequency sweep. This value can be anywhere in the analyzer range.
- **Span** Specifies the span of frequency values measured to either side of the center frequency.

**Zoom - PNA-X ONLY**

Zoom allows you to easily change the start and stop frequencies or start and stop power levels in a power sweep. Zoom operates on the Active Trace and all traces in the same channel as the active trace, regardless of the window in which they appear.
How to Zoom in a measurement window

1. Left-click the mouse or use a finger, then drag across a portion of a trace.
2. Release the mouse or lift the finger and the following menu appears:
3. Select from the following:

   Zoom - changes the channel stimulus settings to the left and right border values of the Zoom selection

   Zoom xy - changes the channel stimulus settings as above. In addition, the Y-axis scale of the active trace changes to the approximate scale of the Zoom selection.

   Zoom Full Out - changes the channel stimulus settings to the full span of the current calibration. If no calibration is ON, then the stimulus settings are changed to the full span of the PNA model.

Notes

- The stimulus settings are changed for ALL traces in the active channel, regardless of the window in which they appear.
- If markers are in the selected area, they remain in place.
- If markers are in the unselected area, they are moved to the right or left edge of the new span. When Zoom Full Out is selected, the markers are moved back to their original location.

Zoom is NOT available for the following:

- Smith Chart or Polar display formats
- CW Time and Segment sweep type
- Frequency Offset Measurements
- FCA Opt 083 Measurements

CW Frequencies

Measurements with a CW Time sweep or Power sweep are made at a single frequency rather than over a range of frequencies.
How to set CW Frequency

1. Set **Sweep Type** to **CW Time** or **Power**.

You can also set CW frequency from within the Sweep Type dialog box.

### Using front-panel HARDKEY [softkey] buttons

<table>
<thead>
<tr>
<th>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2. Press</strong> START/STOP <strong>or</strong> CENTER/SPAN</td>
<td><strong>1. Click Channel</strong></td>
</tr>
<tr>
<td><strong>2. Press</strong> FREQ <strong>then</strong> [CW]</td>
<td><strong>2. then CW Frequency</strong></td>
</tr>
<tr>
<td><strong>3. then [CW]</strong></td>
<td><strong>3. then Frequency</strong></td>
</tr>
<tr>
<td><strong>4. then CW Frequency</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
</tbody>
</table>

### CW Frequency dialog box help

**CW** Type a value and the first letter of the suffix (k,m,or g) or use the up and down arrows to select any value within the range of the PNA.

**Frequency Resolution**

The resolution for setting frequency is 1 Hz.

**Frequency Band Crossings**

The frequency range of the PNA covers several internal frequency bands. The higher the frequency range of the
PNA, the larger the number of bands. The source power to your DUT turns off as the stimulus frequency is swept through these band crossings. To learn more, see Power ON and OFF during Sweep and Retrace.

The listed frequencies in the following tables are the stop frequency of the specified band, and the start frequency of the following band.

Frequency band crossings are different for the following models:

- **3 GHz, 6 GHz, and 9 GHz Models**
- **E8362A/B, E8363A/B, E8364A/B**
- **E8361A**
- **N5230A (2-port models)**
- **N5230A (4-port models)**
- **N5242A**

**For 3 GHz, 6 GHz, and 9 GHz (discontinued) PNA models:**

<table>
<thead>
<tr>
<th>Band</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10 MHz</td>
</tr>
<tr>
<td>2</td>
<td>748 MHz</td>
</tr>
<tr>
<td>3</td>
<td>1500 MHz</td>
</tr>
<tr>
<td>4</td>
<td>3000 MHz</td>
</tr>
<tr>
<td>5</td>
<td>4500 MHz</td>
</tr>
<tr>
<td>6</td>
<td>6500 MHz</td>
</tr>
</tbody>
</table>

**For E8362 / 63 / 64 A/B**

(A models do not have band 0)
<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.045</td>
<td>1</td>
<td>0.748</td>
<td>9</td>
<td>7.60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>1.500</td>
<td>10</td>
<td>10.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>3.00</td>
<td>11</td>
<td>12.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>3.80</td>
<td>12</td>
<td>12.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>4.50</td>
<td>13</td>
<td>15.20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>4.80</td>
<td>14</td>
<td>16.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>6.00</td>
<td>15</td>
<td>20.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>6.40</td>
<td>16</td>
<td>22.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>17</td>
<td>7.60</td>
<td>18</td>
<td>25.60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19</td>
<td>10.00</td>
<td>20</td>
<td>30.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11</td>
<td>12.00</td>
<td>12</td>
<td>12.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>13</td>
<td>15.20</td>
<td>14</td>
<td>16.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15</td>
<td>20.00</td>
<td>16</td>
<td>22.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>17</td>
<td>30.00</td>
<td>18</td>
<td>32.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19</td>
<td>36.00</td>
<td>20</td>
<td>40.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>21</td>
<td>44.70</td>
<td>22</td>
<td>46.20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>23</td>
<td>51.20</td>
<td>24</td>
<td>60.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>25</td>
<td>64.00</td>
<td>1</td>
<td>0.748</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>1.500</td>
<td>10</td>
<td>10.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>3.00</td>
<td>11</td>
<td>12.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>3.80</td>
<td>12</td>
<td>15.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>4.00</td>
<td>13</td>
<td>16.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>4.50</td>
<td>14</td>
<td>20.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>6.00</td>
<td>15</td>
<td>24.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>7.70</td>
<td>16</td>
<td>25.60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>17</td>
<td>30.00</td>
<td>18</td>
<td>32.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19</td>
<td>36.00</td>
<td>20</td>
<td>40.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>21</td>
<td>44.70</td>
<td>22</td>
<td>46.20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>23</td>
<td>51.20</td>
<td>24</td>
<td>60.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>25</td>
<td>64.00</td>
<td>1</td>
<td>0.748</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>1.500</td>
<td>10</td>
<td>10.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>3.00</td>
<td>11</td>
<td>12.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>3.80</td>
<td>12</td>
<td>16.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>4.00</td>
<td>13</td>
<td>20.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>4.50</td>
<td>14</td>
<td>24.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>6.00</td>
<td>15</td>
<td>25.60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>7.70</td>
<td>16</td>
<td>30.00</td>
</tr>
</tbody>
</table>

For E8361A

<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.045</td>
<td>9</td>
<td>10.00</td>
<td>18</td>
<td>32.00</td>
</tr>
<tr>
<td>1</td>
<td>0.748</td>
<td>10</td>
<td>12.00</td>
<td>19</td>
<td>36.00</td>
</tr>
<tr>
<td>2</td>
<td>1.500</td>
<td>11</td>
<td>12.80</td>
<td>20</td>
<td>40.00</td>
</tr>
<tr>
<td>3</td>
<td>3.00</td>
<td>12</td>
<td>15.40</td>
<td>21</td>
<td>44.70</td>
</tr>
<tr>
<td>4</td>
<td>3.80</td>
<td>13</td>
<td>16.00</td>
<td>22</td>
<td>46.20</td>
</tr>
<tr>
<td>5</td>
<td>4.00</td>
<td>14</td>
<td>20.00</td>
<td>23</td>
<td>51.20</td>
</tr>
<tr>
<td>6</td>
<td>4.50</td>
<td>15</td>
<td>24.00</td>
<td>24</td>
<td>60.00</td>
</tr>
<tr>
<td>7</td>
<td>6.00</td>
<td>16</td>
<td>25.60</td>
<td>25</td>
<td>64.00</td>
</tr>
<tr>
<td>8</td>
<td>7.70</td>
<td>17</td>
<td>30.00</td>
<td>1</td>
<td>0.748</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>1.500</td>
<td>10</td>
<td>10.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>3.00</td>
<td>11</td>
<td>12.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>3.80</td>
<td>12</td>
<td>16.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>4.00</td>
<td>13</td>
<td>20.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>4.50</td>
<td>14</td>
<td>24.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>6.00</td>
<td>15</td>
<td>25.60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>7.70</td>
<td>16</td>
<td>30.00</td>
</tr>
</tbody>
</table>

For N5230A 2-port models
<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.045</td>
<td>11</td>
<td>10.500</td>
<td>21</td>
<td>28.600</td>
</tr>
<tr>
<td>2</td>
<td>0.748</td>
<td>12</td>
<td>12.500</td>
<td>22</td>
<td>31.250</td>
</tr>
<tr>
<td>3</td>
<td>1.5</td>
<td>13</td>
<td>15.750</td>
<td>23</td>
<td>31.500</td>
</tr>
<tr>
<td>4</td>
<td>3.125</td>
<td>14</td>
<td>16.667</td>
<td>24</td>
<td>33.333</td>
</tr>
<tr>
<td>5</td>
<td>4.167</td>
<td>15</td>
<td>18.750</td>
<td>25</td>
<td>37.000</td>
</tr>
<tr>
<td>6</td>
<td>5.250</td>
<td>16</td>
<td>21.000</td>
<td>26</td>
<td>40.500</td>
</tr>
<tr>
<td>7</td>
<td>6.250</td>
<td>17</td>
<td>22.500</td>
<td>27</td>
<td>41.667</td>
</tr>
<tr>
<td>8</td>
<td>7.875</td>
<td>18</td>
<td>25.000</td>
<td>28</td>
<td>42.000</td>
</tr>
<tr>
<td>9</td>
<td>8.333</td>
<td>19</td>
<td>26.250</td>
<td>29</td>
<td>46.800</td>
</tr>
<tr>
<td>10</td>
<td>9.375</td>
<td>20</td>
<td>26.500</td>
<td>30</td>
<td></td>
</tr>
</tbody>
</table>

For N5230A 4-port models

<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.00112</td>
<td>9</td>
<td>8.333</td>
</tr>
<tr>
<td>1</td>
<td>0.010</td>
<td>10</td>
<td>9.800</td>
</tr>
<tr>
<td>2</td>
<td>0.040</td>
<td>11</td>
<td>10.500</td>
</tr>
<tr>
<td>3</td>
<td>0.748</td>
<td>12</td>
<td>12.500</td>
</tr>
<tr>
<td>4</td>
<td>1.500</td>
<td>13</td>
<td>15.000</td>
</tr>
<tr>
<td>5</td>
<td>3.125</td>
<td>14</td>
<td>15.750</td>
</tr>
<tr>
<td>6</td>
<td>4.166</td>
<td>15</td>
<td>16.666</td>
</tr>
<tr>
<td>7</td>
<td>5.250</td>
<td>16</td>
<td>18.750</td>
</tr>
<tr>
<td>8</td>
<td>6.250</td>
<td>17</td>
<td>20.100</td>
</tr>
</tbody>
</table>

For N5242A
<table>
<thead>
<tr>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
<th>Band</th>
<th>Freq (GHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Reserved</td>
<td>12</td>
<td>.396</td>
<td>24</td>
<td>8.50</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
<td>13</td>
<td>.500</td>
<td>25</td>
<td>10.664</td>
</tr>
<tr>
<td>2</td>
<td>.014</td>
<td>14</td>
<td>.628</td>
<td>26</td>
<td>12.00</td>
</tr>
<tr>
<td>3</td>
<td>.019</td>
<td>15</td>
<td>1.00</td>
<td>27</td>
<td>12.80</td>
</tr>
<tr>
<td>4</td>
<td>.027</td>
<td>16</td>
<td>1.50</td>
<td>28</td>
<td>13.51</td>
</tr>
<tr>
<td>5</td>
<td>.038</td>
<td>17</td>
<td>2.00</td>
<td>29</td>
<td>15.40</td>
</tr>
<tr>
<td>6</td>
<td>.053</td>
<td>18</td>
<td>3.00</td>
<td>30</td>
<td>16.00</td>
</tr>
<tr>
<td>7</td>
<td>.075</td>
<td>19</td>
<td>3.20</td>
<td>31</td>
<td>18.00</td>
</tr>
<tr>
<td>8</td>
<td>.105</td>
<td>20</td>
<td>4.00</td>
<td>32</td>
<td>20.00</td>
</tr>
<tr>
<td>9</td>
<td>.146</td>
<td>21</td>
<td>5.32</td>
<td>33</td>
<td>21.328</td>
</tr>
<tr>
<td>10</td>
<td>.205</td>
<td>22</td>
<td>6.75</td>
<td>34</td>
<td>22.50</td>
</tr>
<tr>
<td>11</td>
<td>.250</td>
<td>23</td>
<td>8.00</td>
<td>35</td>
<td>24.00</td>
</tr>
</tbody>
</table>

Last modified:

10/23/06  MX Added new band crossings
10/16/06  Moved phase lock lost indicator
9/12/06   Added link to programming commands
9/27/06   MX Added UI
Power Level

Power level is the power of the PNA source at the test ports.

- **How to make Power Settings**
- **Power Dialog Help**
- **Power and Attenuation Dialog Help**
- **Source Unleveled**
- **Setting Independent Port Power**
- **Optimum Attenuation Value**
- **Receiver Attenuation**
- **Power ON and OFF during Sweep and Retrace**

See other 'Setup Measurements' topics

---

**Power Settings**

The test port output power is specified over frequency (See the Power Range and Frequency Range specifications for your PNA).

**How to make Power settings**

Use one of the following methods to set port power. Only the menu can be used to launch the Power and Attenuators dialog box.

### Using front-panel HARDKEY [softkey] buttons

For **N5230A** and **E836xA/B models**

1. Press **Power**
2. Then

For **PNA-X** and **'C' models**

1. Click **Channel**
2. Then **Power**

### Using a mouse with PNA Menus

- **Programming Commands**
1. Press **Stimulus**
2. then **[Power] or [Power and Attenuators]**

### Power dialog box help

Defines and controls the PNA source power and attenuation.

**Power On (All Channels)** Check to enable source power for all channels. Only turns power ON if channel power setting is ON or Auto. See Advanced Power.

#### Note: Power ON/OFF setting and Instrument State Save and Recall.

If power is OFF when an instrument state is saved, the power will be OFF when the state is recalled.

If power is ON when an instrument state is saved, then when recalled, the power setting will be the SAME as the current power setting. To protect your DUT, power will NOT be turned ON by an instrument state recall if the current power setting is OFF.

**User Preset** follows this instrument state save/recall behavior.

**Instrument Preset** always includes Power ON.

**Port 'n'** Active source port for which power is being set.

**Port Power** Sets the power level for the specified port.

- To accurately set the power level at any point after the test port, perform a Source Power Calibration.

- See the specified power range of your PNA model.

**Power Sweep**

**Start / Stop Power** Set the start and stop power values of a power sweep. These settings are only available when **Sweep Type** is set to Power Sweep. Sweep power can also be set from the Advanced Power dialog box.
Zoom - allows you to easily change the start and stop power levels in a power sweep. Learn how.
Learn more about Power Sweep.

Power Slope

Helps compensate for cable and test fixture power losses at increased frequency.

Slope  Select to set the power slope. Clear to set power slope OFF.

- With power slope enabled, the port output power increases (or decreases) as the sweep frequency increases.
- The units of power slope are dB/GHz.
- Power slope can only be set to values of 0.5, 1, 1.5, or 2 (positive or negative).

Power and Attenuators dialog box help

Defines and controls the PNA source power and attenuation for the active channel.

Beginning with PNA Rev. 7.21, external sources can be controlled from this dialog. Learn more.

Power On (All Channels)  Check to enable source power for all channels. Only turns power ON if channel power setting is ON or Auto.

Port Powers Coupled

- Coupled (checked)  The power levels are the same at each test port. Set power at any test port and all test ports change to the same power level.

- Uncoupled (cleared)  The power levels are set independently for each test port. Uncouple power, for example, if you want to measure the gain and reverse-isolation of a high-gain amplifier. The power required for the input port of the amplifier is much lower than the power required for the output port. Learn more about Setting Independent Port Power

Name  Lists the PNA test ports.

State  ON and OFF are valid ONLY on PNA models with an internal second source.
Auto  Source power is turned ON at the specified test port when required by the measurement. This is the most common (default) setting.

ON  Source power is ALWAYS ON, regardless of measurements that are in process. Use this setting to supply source power to a DUT port that always requires power, such as an LO port. This could turn OFF power at another test port. Learn about internal second source restrictions.

OFF  Source power is never ON, regardless of the measurement requirements. Use this setting to prevent damage to a sensitive DUT test port.

Port Power  Sets the power level at the output of the source.

- To accurately set the power level at any point after the test port, perform a Source Power Calibration.
- See the specified power range of your PNA model.

Start / Stop Power  Available ONLY when sweep type is set to Power Sweep. Set the start and stop power values of a power sweep. Learn more about Power Sweep.

In PNA release 6.04 you can specify whether to maintain source power at either the start power or stop power level at the end of a power sweep. To do this, send SYST:PREF:ITEM:PSRT from the GPIB Command Processor Console.

Auto Range  Check to allow the PNA to select the optimum attenuation value to achieve the specified test port power.

Clear to manually set the attenuation for each port. Type or select the attenuation value in the adjacent Attenuator Control box.

Attenuator Control  When Port Powers are Uncoupled, manual attenuator control allows you to set a wide range of power levels by setting the attenuation. See Setting Independent Port Power. Also use manual attenuation control when a measurement requires a very good impedance match with the source, such as with oscillators or conditionally unstable amplifiers. Choose an attenuation level of 10 dB or more to ensure the best source match.

When Port Powers are Coupled, changing one port Attenuation Control value changes all port values.

Attenuation is in between the Source and the test port. Power to the reference receiver is not attenuated and is therefore higher than at the test port by the amount of attenuation. This will make an uncalibrated measurement using a reference receiver appear as though there is added attenuation at the test device. See the PNA Block diagram.

Note: With PNA release 7.2, a preference can be set to mathematically offset (or NOT) the power at the reference receiver by the amount of source attenuation. Because the reference receiver is not in the attenuation path, there is more power at the reference receiver than at the test port by the amount of source attenuation.

By default, ALL PNA models currently offset the reference receivers. See Block diagram.

See the SCPI and COM commands used to set the preference.

Leveling Mode

Internal  Standard ALC leveling. Power level within an attenuator setting is limited to the ALC Range. See Source Unleveled.

Open Loop  (Used during pulse conditions with the internal source modulators). No leveling is used in setting
the source power. The lowest settable power, without attenuation, is limited to -30dBm. The source power level accuracy is very compromised. Use a source power calibration to make the source power somewhat more accurate.

Source Unleveled

When the power level that is required at a test port is higher than the PNA can supply, a Source Unleveled error message appears on the screen and the letters LVL appear on the status bar.

To perform a power sweep, the range of power is usually limited to the range of the Automatic Leveling Control (ALC) loop. (The PNA-X allows a very wide power range using Open Loop).

PNA specifications guarantee the ALC power range over which the PNA can supply power without an unleveled indication. However, the actual achievable power range on your PNA is probably greater than the specified range.

How to calculate the specified achievable power range

From the specifications for the N5230A Opt 245 for the frequency span from 15 GHz to 20 GHz:

- Max Leveled Power = -8 dBm
- Power Sweep Range (ALC) = -17 dB

For this frequency range the specified power range is calculated as:

- Max = -8 dBm
- Min = (-8)-(17) = -25 dBm

When using Source Attenuators:

- with 10dB of attenuation, this becomes -18 dBm to -35 dBm
- with 20dB of attenuation, this becomes -28 dBm to -45 dBm, and so forth.

See the output power specs for your PNA.

To resolve an unleveled condition, change either the Test Port Power or Attenuator setting. If an Unleveled condition exists within the specified power range, contact Technical Support.

Setting Independent Port Power

The PNA allows you to uncouple port power and specify different power levels at each test port. There are a few things to consider when setting independent port powers.

- Does your required high and low power levels fall within the specified Min and Max power range of the PNA? See Unleveled Indicator. If they do not, you may need to use the internal Source Attenuators.

- Does the PNA have source attenuators? If so, how many source attenuators? Some PNA models have one attenuator for each port. In most multiport PNA systems, the attenuators are shared by at least two test ports. See PNA Options to see the availability and range of source attenuation on your PNA.
Note: To prevent premature wear on source and receiver attenuators, the PNA does not allow attenuators (or other mechanical switches) to switch between settings when continuously triggering. Attenuator values are set for the entire channel.

When different channels are used and settings require an attenuator to switch value, only one channel is allowed to sweep continuously. All other channels are automatically put in Trigger Hold.

To override this condition, change the 'Hold' channel to Single trigger or Group trigger, which allows up to 2 million triggers. The attenuator will then be allowed to switch settings for each channel.

Optimum Attenuation Value

The attenuator has different positions, allowing a wide range of power levels. The number of power ranges available is determined by the source attenuation installed in your PNA. See PNA Options to see the availability and range of source attenuation on your PNA.

- Each range has a total specified span (25 dB in the following Attenuation Values graphic).
- The optimum setting is the middle of the range. This range provides the best accuracy and performance of the source leveling system. The optimum ranges are the blue regions in the following graphic.
- An attenuator setting can be selected manually or automatically. If automatic is selected, the blue optimum ranges (shown in the following graphic) are used.

(Attenuator ranges vary, this particular range is 70 dB)

Note: Error correction is fully accurate only for the power level at which a measurement calibration was performed. However, when changing power within the same attenuator range at which the measurement calibration was...
performed, ratioed measurements can be made with nearly full accuracy (non-ratioed measurements with less accuracy).

**Receiver Attenuators** dialog box help

Type or select independent attenuation values for each receiver.

Receiver Attenuation, available as option 016 on some PNA models, is used to attenuate the output signal from the device under test to avoid damaging the PNA receivers. The PNA receivers typically start to compress at around +10 dBm. This causes the power level at the receiver to be less than the power at the test port by the specified amount of attenuation.

**Note:** With PNA release 7.2, a preference can be set to mathematically offset (or NOT) the displayed trace by the amount of receiver attenuation. This causes the display to show the power at the test port.

By default:

- PNA-L and E836xB do **NOT** offset the display.
- The PNA-X **DOES** offset the display.

See the SCPI and COM commands used to set the preference.

When an external test set is connected, Receiver Attenuation control is only available for the physical receivers in the PNA. Switching receiver attenuation using logical receiver notation is NOT allowed.

**CAUTION!** You can damage the analyzer receivers if the power levels exceed the maximum values. See your analyzer’s Technical Specifications for the maximum input power to a receiver.

The receiver attenuator characteristics are:

- Range:
  - 0 to 50 dB (E8361A only)
- 0 to 35 dB (all other applicable PNA models)
- Resolution:
  - 10 dB (E8361A only)
  - 5 dB (all other applicable PNA models)

**Power ON and OFF during Sweep and Retrace**

The frequency range of the PNA covers several internal frequency bands. The higher the frequency range of the PNA, the larger the number of bands. For example, a 9 GHz PNA has 6 frequency bands, a 50 GHz PNA has 25 frequency bands. See the frequency band crossings.

Power to the DUT is turned OFF during band changes to avoid causing power spikes to the DUT.

Retrace occurs when the source gets to the end of your selected frequency span and moves back to the start frequency. Power to the DUT is again turned OFF when retraceing across frequency bands.

Therefore, the following occurs for various stimulus settings:

1. **Single band sweep** - The power to the DUT is always ON, even during retrace.
   In PNA release 6.04, a preference setting can turn power OFF during a retrace. Only available in single band frequency and segment sweeps.

2. **Multi-band sweep** - The power to the DUT is turned OFF while sweeping across a band crossing. It is turned OFF again during retrace.

3. **Power sweep** - Because power sweep is always done at a single frequency, the frequency is always within a single band and the source power is always ON. At the end of a power sweep, power is immediately set to the start power.
   In PNA release 6.04, this behavior can be changed with a preference setting.

4. **Single sweep**:
   - Manual trigger mode - At the end of a multiband sweep, power is turned OFF during retrace, and then power is turned back ON before arming for the next trigger.
   - Hold mode - Power can be ON or OFF depending on when and how Hold mode is entered. However, power can be immediately turned OFF manually or remotely.

**Caution**: Avoid expensive repairs to your PNA. Read Electrostatic Discharge Protection.
26-Mar-2007  Clarified retrace power OFF

11/16/06  Added new retrace features

10/23/06  Modified for new power diag

10/17/06  Clarified leveling

9/12/06  Added link to programming commands
Sweep Settings

A sweep is a series of consecutive data point measurements taken over a specified sequence of stimulus values. You can make the following sweep settings:

- **Sweep Type**
- **Sweep Time**
- **Sweep Setup**

See [Triggering](#) and other ‘Setup Measurements’ topics

### How to set Sweep Type

**Using front-panel HARDKEY [softkey] buttons**

**For N5230A and E836xA/B models**

1. Press **Sweep Type**
2. then **[Active Entry keys]**

**For PNA-X and ‘C’ models**

1. Press **Sweep**
2. then **[Sweep Type]**

**Programming Commands**

### Sweep Type dialog box help
**Note:** Sweep Settings are not applied until either **OK** or **Apply** is pressed.

**Channel** The active channel when Sweep Type was selected. Sweep settings will be applied to this channel.

### Sweep Type

**Linear Frequency** Sets a linear frequency sweep that is displayed on a standard grid with ten equal horizontal divisions.

- **Start** Sets the beginning value of the frequency sweep.
- **Stop** Sets the end value of the frequency sweep.
- **Points** Sets the number of data points that the PNA measures during a sweep. Range: 2 to 20001. (Default is 201).

**Log Frequency** The source is stepped in logarithmic increments and the data is displayed on a logarithmic x-axis. This is usually slower than a continuous sweep with the same number of points.

- **Start** Sets the beginning value of the frequency sweep.
- **Stop** Sets the end value of the frequency sweep.
- **Points** Sets the number of data points that the PNA measures during a sweep. Range: 2 to 20001. (Default is 201).

**Power Sweep** Activates a power sweep at a single frequency that you specify. [Learn about power sweep](#)

- **Start** Sets the beginning value of the power sweep.
- **Stop** Sets the end value of the power sweep.
- **CW Frequency** Sets the single frequency where the PNA remains during the measurement sweep.

**CW Time** Sets the PNA to a single frequency, and the data is displayed versus time.

- **CW Frequency** Sets the frequency where the PNA remains during the measurement.
- **Sweep Time** Sets the duration of the measurement, which is displayed on the X-axis.
- **Points** Sets the number of data points that the PNA measures during a sweep. Range: 2 to 20001. (Default is 201).

**Segment Sweep** Sets the PNA to sweep through user-defined sweep segments. [Learn how to make these settings](#)

- **Independent Power Levels** Check to set the source power level for each segment. [Test port uncoupling](#) is also allowed.
- **Independent IF Bandwidth** Check to set the IF bandwidth for each segment.
- **Independent Sweep Time** Check to set the duration of the measurement for each segment.
- **X-Axis Point Spacing** Check to scale the X-Axis to include only the segments. [Learn more](#).
- **Allow Arbitrary Segments** Check to allow arbitrary frequencies (overlapped or reverse sweeps). [Learn more](#).
- **Show Table** Shows the table that allows you to create and edit segments.
- **Hide Table** Hides the segment table from the screen.

**OK** Applies setting changes and closes the dialog box.

**Apply** Applies setting changes and leaves the dialog box open to make more setting changes.

**Cancel** Closes the dialog. Setting changes that have been made since the last Apply button click are NOT applied.

---

**Power Sweep**

A power sweep either increases or decreases source power in discrete steps. Power sweep is used to characterize power-sensitive circuits, with measurements such as gain compression or AGC (automatic gain control) slope.

In the Sweep Type dialog, specify Start power, Stop power, and CW Frequency. Power can be swept over any attainable range within the PNA ALC range.

**Note:** If the PNA has source attenuators, and the attenuation must be changed in order to achieve the requested start and stop power, click **Channel**, then **Power** to [set the power and attenuation](#).

The PNA does NOT allow a single power sweep over a range that requires attenuator switching. However, two power sweeps can be performed in different channels. The attenuators will not be allowed to switch continuously, but triggering can be performed using single or group triggering. [Learn more](#).

The remaining power settings apply in power sweep mode:

- Port Power is always coupled.
- Test Port Power setting is ignored.
- Attenuator Control is always Manual.
- Power Slope (dB/GHz) is ignored. The output frequency is CW.

Click **Sweep**, then **Number of Points** to change the step size of the power sweep.

Beginning with PNA release 6.04 you can specify whether to maintain source power at either the start power or stop power level at the end of a power sweep. To do this, send `SYST:PREF:ITEM:PSRT` from the GPIB Command Processor Console.

---

**Segment Sweep**

Segment Sweep activates a sweep which consists of frequency sub-sweeps, called segments. For each segment you can define independent power levels, IF bandwidth, and sweep time.
Once a measurement calibration is performed on the entire sweep or across all segments, you can make calibrated measurements for one or more segments.

In segment sweep type, the analyzer does the following:

- Sorts all the defined segments in order of increasing frequency
- Measures each point
- Displays a single trace that is a composite of all data taken

Restrictions for segment sweep:

- The frequency range of a segment is not allowed to overlap the frequency range of any other segment.
- The number of segments is limited only by the combined number of data points for all segments in a sweep.
- The combined number of data points for all segments in a sweep cannot exceed 20001.
- All segments are FORCED to have power levels within the same attenuator range to avoid premature wear of the mechanical step attenuator. See Power Level.

### How to make segment sweep settings

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press <strong>Sweep Type</strong></td>
<td>1. Click <strong>Sweep</strong></td>
</tr>
<tr>
<td>2. then [<strong>Active Entry keys</strong>]</td>
<td>2. then <strong>Segment Table</strong></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>For <strong>PNA-X</strong> and <strong>'C' models</strong></td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press <strong>Sweep</strong></td>
<td>1. Click <strong>Stimulus</strong></td>
</tr>
<tr>
<td>2. then [<strong>Sweep Type</strong>]</td>
<td>2. then <strong>Sweep</strong></td>
</tr>
<tr>
<td></td>
<td>3. then <strong>Sweep Type</strong></td>
</tr>
</tbody>
</table>
**Insert Segment** - adds a sweep segment before the selected segment. You can also click the "down" arrow on your keyboard to quickly add many segments.

**Delete Segment** - removes the selected segment.

**Delete All Segments** - removes all segments.

**Note:** At least ONE segment must be ON or **Sweep Type** is automatically set to **Linear**.

### To Modify an Existing Segment

To make the following menu settings available, you must first show the segment table.

Click **View**, point to **Tables**, then click **Segment Table**.

<table>
<thead>
<tr>
<th>STATE</th>
<th>START</th>
<th>STOP</th>
<th>F1</th>
<th>IFBW</th>
<th>P1 Pwr</th>
<th>P2 Pwr</th>
<th>TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>20.000000 GHz</td>
<td>1.000000 GHz</td>
<td>21</td>
<td>10.0 kHz</td>
<td>17.00 dBm</td>
<td>0.00 dBm</td>
<td>2.474 sec</td>
</tr>
<tr>
<td>ON</td>
<td>1.000000 GHz</td>
<td>4.000000 GHz</td>
<td>21</td>
<td>35.0 kHz</td>
<td>17.00 dBm</td>
<td>0.00 dBm</td>
<td>630,000 μsec</td>
</tr>
</tbody>
</table>

The above graphic shows the Segment table with all independent settings selected, including source power uncoupled (two power settings).

**STATE** Click the box on the segment to be modified. Then use the up / down arrow to turn the segment ON or OFF.

**START** Sets start frequency for the segment. Click the box and type a value and the first letter of a suffix (KHz, MHz, GHz). Or double-click the box to select a value.

**STOP** Sets stop frequency for the segment. Click the box and type a value and the first letter of a suffix (KHz, MHz, GHz). Or double-click the box to select a value.

**POINTS** Sets number of data points for this segment. Type a value or double-click the box to select a value.

### To set IFBW, Power, and Sweep Time independently for each segment:

1. On the **Sweep** menu, click **Sweep Type**, then **Segment Sweep**.
2. Check the appropriate **Sweep Properties** boxes
3. Then click the box and type a value or double-click the box and select a value.

**Note:** If the following are NOT set, the entire sweep uses the channel IFBW, Power, and Time settings.

**IFBW** Sets the **IF Bandwidth** for the segment.

**POWER** Sets the **Power level** for the segment. You can also UNCOUPLE the test port power. See **Power Coupling**.

**TIME** Sets the **Sweep time** for the segment.

### X-Axis Point Spacing - Segment Sweep ONLY

This feature affects how a segment trace is drawn on the screen.

**How to select X-Axis Point Spacing**

On the **Sweep Type** dialog box, click **Segment Sweep**

Then check **X-Axis Point Spacing**
- **Without X-axis point spacing**, a multi-segment sweep trace can sometimes result in squeezing many measurement points into a narrow portion of the x-axis.
- **With X-axis point spacing**, the x-axis position of each point is chosen so that all measurement points are evenly spaced along the x-axis.

For example, given the following two segments:

<table>
<thead>
<tr>
<th>STATE</th>
<th>START</th>
<th>STOP</th>
<th>POINTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ON 45.00000 MHz 50.00000 MHz</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ON 170.00000 MHz 180.00000 MHz</td>
<td>21</td>
<td></td>
</tr>
</tbody>
</table>

**Arbitrary Segment Sweep**

This feature allows arbitrary frequencies to be entered into the segment sweep table. With this capability, segments can have:

- overlapping frequencies.
• the stop frequency less than the start frequency (reverse sweep).

### How to enable Arbitrary Segment Sweep

1. On the **Sweep Type** dialog box, click **Segment Sweep**
2. Check **Allow Arbitrary Segment Sweep**

### Notes:

- Unusual results may occur when using arbitrary sweep segments with markers, display settings, limit lines, formatting, and some calibration features.
- When **Allow Arbitrary Segment** is checked, **X-axis point spacing** is automatically turned ON.

---

### Sweep Time

The PNA automatically maintains the fastest sweep time possible with the selected measurement settings. However, you can increase the sweep time to perform a slower sweep.

<table>
<thead>
<tr>
<th>How to set Sweep Time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Using front-panel HARDKEY [softkey] buttons</strong></td>
</tr>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
</tr>
<tr>
<td>1. Press <strong>Sweep Setup</strong></td>
</tr>
<tr>
<td>2. then [Active Entry keys]</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
</tr>
<tr>
<td>1. Press <strong>Stimulus</strong></td>
</tr>
<tr>
<td>2. then <strong>Sweep</strong></td>
</tr>
<tr>
<td>3. then <strong>Sweep Time</strong></td>
</tr>
</tbody>
</table>
Sweep Time dialog box help

Specifies the time the PNA takes to acquire data for a sweep. The maximum sweep time of the PNA is 86400 seconds or 1 day. Learn about other settings that affect sweep speed.

Note: If sweep time accuracy is critical, use ONLY the up and down arrows next to the sweep time entry box to select a value that has been calculated by the PNA. Do NOT type a sweep time value as it will probably be rounded up to the closest calculated value. This rounded value will not be updated in the dialog box.

- The actual sweep time includes this acquisition time plus some "overhead" time.
- The PNA automatically maintains the fastest sweep time possible with the selected measurement settings. However, you can increase the sweep time using this setting.
- Enter 0 seconds to return the analyzer to the fastest possible sweep time.
- The Sweep Time setting is applied to the active channel.
- The sweep time is per sweep. A full 2-port cal requires two sweeps, both using the specified sweep time. Learn more.
- A Sweep Indicator appears on the data trace when the Sweep Time is 0.3 seconds or greater, or if trigger is set to Point Sweep Mode. The indicator is located on the last data point that was measured by the receiver. If the indicator is stopped (point sweep mode) the source has already stepped to the next data point.

Sweep Setup

How to make Sweep Setup settings

Using front-panel HARDKEY [softkey] buttons

For N5230A and E836xA/B models

1. Press SWEEP SETUP

PNA Menu using a mouse

1. Click Sweep
2. then Sweep Setup

For PNA-X and 'C' models

1. Press STIMULUS
2. then [Sweep]
3. then [Sweep Setup]
**Sweep Setup** dialog box help

**Channel**  Specifies the channel that the settings apply to.

**Stepped Sweep**  When checked (Stepped Sweep) the PNA source is tuned, then waits the specified Dwell time, then takes response data, then tunes the source to the next frequency point. This is slower than Analog Sweep, but is more accurate when testing electrically-long devices.

When cleared (Analog Sweep) the PNA takes response data AS the source is sweeping. The sweep time is faster than Stepped, but could cause measurement errors when testing electrically-long devices.

When the dialog checkbox is cleared, the PNA could be in either Analog or Step mode. There is no display indication of whether the PNA is in Analog or Stepped Sweep. Step mode is automatically selected for a number of reasons. Here are some of the reasons:

- **IF Bandwidth** is at, or below, 1 kHz.
- **Source Power Correction** is ON unless doing CW measurement.
- When more than one source is turned ON ([multisource PNA models](#)).
- When step mode is a faster way to take the data.
- For all **FOM** and **FCA** measurements.
- For all **ADC measurements**.
- For all **MMwave measurements**.

**Dwell Time**  Specifies the time the source stays at each measurement point before the analyzer takes the data. Only applies to stepped sweep. The maximum dwell time is 100 seconds. See also [Electrically Long Devices](#).

**Alternate Sweeps**  When checked, the PNA measures only one receiver per sweep.

When cleared, the PNA measures both the A and B receivers (if used) each sweep. See also [Crosstalk](#).

**External ALC**  Available ONLY on 3 GHz, 6 GHz, and 9 GHz PNA models (now discontinued).

When checked, the analyzer is enabled to receive an external signal that you provide for leveling the source output. The external ALC signal input connector is the **External Detector Input** on the rear panel.
21-Jun-2007  Increased max data points

3-May-2007  Updated Step mode conditions

9/27/06  MX Added UI

9/12/06  Added link to programming commands
Trigger

A trigger is a signal that causes the PNA to make a measurement sweep. The PNA offers great flexibility in configuring the trigger function.

View the interactive Trigger Model animation to see how triggering works in the PNA.

- **How to Set Trigger**
- **Source**
- **Scope**
- **Channel Settings**
- **Restart**
- **External and Auxiliary Triggering** (separate topic)

### See other 'Setup Measurements' topics

---

**How to set Triggering**

**Using front-panel HARDKEY [softkey] buttons**

For **N5230A and E836xA/B models**

1. Press TRIGGER

For **PNA-X and 'C' models**

1. Press TRIGGER
2. then [Trigger...]

**PNA Menu using a mouse**

1. Click Sweep
2. then Trigger
3. then Trigger

**Programming Commands**

**Note:** The Continuous, Single, and Hold settings apply ONLY to the active channel. These settings are available from the Trigger menu, Active Entry keys, and softkeys.
**Trigger Setup** dialog box help

View the interactive Trigger Model animation to see how triggering works in the PNA.

**Trigger Source**

These settings determine where the trigger signals originate for all existing channels. A valid trigger signal can be generated only when the PNA is not sweeping.

**Internal**  Continuous trigger signals are sent by the PNA as soon as the previous measurement is complete.

**Manual**  One trigger signal is sent when invoked by the Trigger button, the active toolbar, or a programming command.

**External**  Trigger signals sent out or received from various connectors on the rear panel. Learn more about External and AUX Triggering.

**Manual Trigger!** - Manually sends one trigger signal to the PNA. Available ONLY when Manual trigger is selected.

**Trigger Scope**

These settings determine what is triggered.

**Global**  All channels not in Hold receive the trigger signal [Default setting]

**Channel**  Only the next channel that is not in Hold receives the trigger signal. This is not obvious or useful unless Trigger Source is set to Manual. This setting enables Point Sweep mode.

**Channel Trigger State**

These settings determine how many trigger signals the channel will accept.

**Continuous**  The channel accepts an infinite number of trigger signals.
Groups  The channel accepts only the number of trigger signals that is specified in the Number of Groups text box, then goes into Hold. Before selecting groups you must first increment the Number of Groups text box to greater than one.

Number of Groups  Specify the number of triggers the channel accepts before going into Hold. If in Point Sweep, an entire sweep is considered one group.

First increment to desired number, then select 'Groups'.

Single  The channel accepts ONE trigger signal, then goes into Hold.

Another way to trigger a single measurement is to set Trigger Source to Manual, then send a Manual trigger. However, ALL channels are single triggered.

Hold  The channel accepts NO trigger signals.

Trigger Mode

These settings determine what EACH signal will trigger.

Sweep and Point modes are available ONLY when both Trigger Source = MANUAL or EXTERNAL AND Trigger Scope = CHANNEL.

Channel  Each trigger signal causes ALL traces in that channel to be swept in the order specified below.

Sweep  Each Manual or External trigger signal causes ALL traces that share a source port to be swept in the order specified below. When in Groups or Single trigger, the count is decremented by one after ALL traces in ALL directions are swept.

When correction is ON which requires sweeps in more than one direction, traces on the screen will not update until all of the relevant directions have been swept. For example, with all four 2-port S-Parameters displayed:

- When correction is OFF, trigger 1 causes S11 and S21 to update; trigger 2 causes S22 and S12 to update.
- When Full 2-port correction is ON, trigger 1 causes NO traces to update; trigger 2 causes ALL S-Parameters to update. Learn more about sweeps with correction ON.

Point  Each Manual or External trigger signal causes one data point to be measured. Subsequent triggers go to the same trace until it is complete, then other traces in the same channel are swept in the order specified below. When in Groups or Single trigger, the count is decremented by one after ALL data points on ALL traces in the channel are measured. See Also, the (point) Sweep Indicator and SCPI Triggering example for use with External.
**Trace Sweep Order**

For ALL Trigger Modes, traces within each channel are always swept in the following order. Trigger signals continue in the same channel until all traces in that channel are complete. Triggering then continues to the next channel that is not in HOLD.

- Traces are swept sequentially in source-port order. For example, in a channel with all four 2-port S-parameters, first the source port 1 traces (S11 and S21) are swept simultaneously. Then the source port 2 traces (S22 and S12) are swept simultaneously.

- In addition, when **Alternate sweep** is selected, traces are swept sequentially in source-port / receiver-port order. In the above example, first the S11 trace is swept, then S21, then S12, then S22.

**Restart** (Available only from the Trigger menu) Channels in Hold are set to single trigger (the channel accepts a single trigger signal). All other settings are unaffected, including decrementing trigger Groups.

**See Also**

- [External and AUX Triggering](#)
- Interactive [Trigger Model](#) animation

---

Last modified:

- 26-Oct-2007 Added Trigger Mode
- 15-Dec-2006 Added MX capability
- 9/12/06 Added link to programming commands
External and Auxiliary Triggering

External and Auxiliary triggering are both used to synchronize the triggering of the PNA with other equipment or events.

- **Overview**
- **Capability Summary for each PNA Model**
- **Dialogs**
  - Auxiliary Triggering (PNA-X only)
  - External Trig (IN) Dialog (All models)
  - I/O2 Trig Out Dialog (PNA-L and E836x)

See Also
- Controlling a Handler
- Synchronizing an External Source
- PNA Triggering

Overview

The manner in which External Triggering is performed has evolved throughout the PNA history, with each new model becoming more comprehensive and flexible. Unfortunately, our ability to update the older models is limited as a large part of external triggering is dependent on the PNA hardware. Where possible, we have updated the capability of the older models with software.

Ready Signals versus Trigger Signals

A 'Ready for Trigger' signal is different from a Trigger signal. The ready signal indicates that the instrument sending the signal is ready for measurement. The instrument receiving the ready signal would then send a trigger signal, indicating that the measurement will be made, or has been made. Usually the slower instrument sends the trigger signal. The following two scenarios illustrate when the PNA is faster, and slower than the external device:

- A material handler is very mechanical and takes a relatively long time to load and discharge parts. Therefore, the PNA sends a 'Ready' signal when it is setup to measure, and the handler sends a trigger signal to the PNA when it is ready for a measurement. Additional signals are available on the PNA Handler I/O to indicate that the PNA sweep has ended, and that the handler can setup for the next measurement. See a procedure.

- Alternatively, an external source usually sets up for the next measurement much faster than the PNA. This is because the PNA acquires data, and moves both source and receivers for the next measurement. In this case, the external source sends a 'Ready' signal. The PNA then begins the measurement and sends a trigger signal AFTER the measurement has been made. This indicates that the measurement is complete.
and that the source should setup for the next measurement. See a procedure. Beginning with A.07.22, the PNA can control an external source from within the firmware. Learn more.

**Capability Summary for each PNA Model**

The following describes the capabilities and recommended method of triggering for each PNA model.

<table>
<thead>
<tr>
<th>Signal Pair</th>
<th>Rear-Panel Connectors</th>
<th>Control Settings (click to learn more)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNA Ready for Trigger (OUT)</td>
<td>MEAS TRIG RDY and Handler I/O p21</td>
<td>Meas Trigger TAB</td>
</tr>
<tr>
<td>Trigger IN to PNA</td>
<td>MEAS TRIG IN and Handler I/O p18</td>
<td>Meas Trigger TAB</td>
</tr>
<tr>
<td>Trigger OUT of PNA</td>
<td>AUX TRIG OUT (1&amp;2)</td>
<td>AUX Trig TAB</td>
</tr>
<tr>
<td>Ext Device Ready (IN to PNA)</td>
<td>AUX TRIG IN (1&amp;2)</td>
<td>AUX Trig TAB</td>
</tr>
</tbody>
</table>

**PNA-L models**

The I/O (TRIG IN) and I/O TRIG OUT signal pair is the recommended signal pair to synchronize the PNA-L and external devices. Both signals result in triggering the other instrument; neither of these signals indicate a ‘Ready’ condition.

<table>
<thead>
<tr>
<th>Recommended Signal Pair</th>
<th>Rear-Panel Connectors</th>
<th>Control Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger IN to PNA</td>
<td>BNC IN</td>
<td>External TAB</td>
</tr>
<tr>
<td>Trigger OUT of PNA</td>
<td>BNC OUT</td>
<td>I/O Trig TAB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other Signals</th>
<th>Rear-Panel Connectors</th>
<th>Control Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNA Ready for Trigger (OUT)</td>
<td>AUX I/O p18</td>
<td>SCPI and COM Only</td>
</tr>
<tr>
<td></td>
<td>Handler I/O p21 (some PNA-L models)</td>
<td>SCPI and COM Only</td>
</tr>
<tr>
<td>Trigger IN to PNA</td>
<td>AUX I/O p19</td>
<td>SCPI and COM Only</td>
</tr>
<tr>
<td></td>
<td>Handler I/O p18 (some PNA-L models)</td>
<td>SCPI and COM Only</td>
</tr>
</tbody>
</table>
**E836xA/B/C**

The I/O (TRIG IN) and I/O TRIG OUT signal pair is the recommended signal pair to synchronize the E836x and external devices. Both signals result in triggering the other instrument; neither of these signals indicate a 'Ready' condition.

<table>
<thead>
<tr>
<th>Recommended Signal Pair</th>
<th>Rear-Panel Connectors</th>
<th>Control Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger IN to PNA</td>
<td>BNC IN</td>
<td>External TAB</td>
</tr>
<tr>
<td>Trigger OUT of PNA</td>
<td>BNC OUT</td>
<td>I/O Trig Out TAB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other Signals</th>
<th>Rear-Panel Connectors</th>
<th>Control Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNA Ready for Trigger (OUT)</td>
<td>AUX I/O p18</td>
<td>SCPI and COM Only</td>
</tr>
<tr>
<td>Trigger IN to PNA</td>
<td>AUX I/O p19</td>
<td>SCPI and COM Only</td>
</tr>
<tr>
<td>Ext Device Ready (IN to PNA)</td>
<td>None</td>
<td>N/A</td>
</tr>
</tbody>
</table>

See how to access the Trigger Dialog
Aux Trig 1 - Aux Trig 2 dialog box help

This reciprocal pair of signals on PNA-X models ONLY, offers high flexibility, and robust synchronization with external devices.

- When enabled, the PNA-X waits indefinitely for a 'Ready IN' signal on the AUX IN connector from an external device.
- When received, the PNA is triggered from the usual trigger sources (Internal, External, or Manual).
- The trigger output signal on the AUX OUT connector can be sent BEFORE or AFTER data acquisition.
- Each channel can be configured differently.
- Two pair of AUX TRIG connectors allow two external devices to be controlled simultaneously.

See Also

- See how to use these connectors to synchronize with External Sources.
- To use the opposite reciprocal pair, see Meas Trig IN and Ready OUT pair.

Dialog Settings

Note: The Aux Trig 1 and Aux Trig 2 tabs are identical.

Enable Check to use the Aux1 or Aux2 connectors to output signals to an external device.
**Channel** All settings on this dialog affect the specified channel ONLY.

**OUT (Trigger)**

After receiving the Aux Trig IN ‘Ready’ signal, the trigger signal comes from any of the following Trigger Sources:

- **Internal** - trigger occurs immediately.
- **Manual** - trigger occurs when the Trigger button is pressed.
- **External** - trigger occurs when Meas Trig In signal is received. This must be configured independently.

The following settings control the properties of the signals sent out the rear panel AUX TRIG OUT (1&2) connectors:

**Polarity**
- **Positive Pulse**  Outgoing pulse is positive.
- **Negative Pulse**  Outgoing pulse is negative.

**Position**
- **Before Acquisition**  Pulse is sent immediately before data acquisition begins.
- **After Acquisition**  Pulse is sent immediately after data acquisition is complete.

**Per Point** Check to cause a trigger output to be sent for each data point. Clear to send a trigger output for each sweep. This setting controls the trigger output signal regardless of the channel Point trigger setting, which causes the PNA channel to trigger per point. For example, to trigger the PNA channel per point, and output a trigger signal per point, both this, and the channel setting must be checked ON.

**Pulse Duration** Specifies the duration of the positive or negative output trigger pulse.

**Ready for Trigger Handshake**

When checked, the PNA waits indefinitely for the input line at the rear panel AUX TRIG OUT (1&2) connectors to change to the specified level before acquiring data. This signal indicates that the external device is ready for PNA data acquisition. If the signal arrives before the PNA is ready to acquire data, it is latched (remembered).

When NOT checked, the PNA-X does not wait, but outputs trigger signals when the PNA-X is ready.

This signal does NOT trigger the PNA-X. The trigger signal is generated from any of the usual sources: Internal, Manual, or External.

**IN (READY)**
- **Ready High**  PNA responds to the leading edge of a pulse on the Aux1 or Aux2 In connector.
- **Ready Low**  PNA responds to the trailing edge of a pulse on the Aux1 or Aux2 In connector.

**Delay**  Time that the PNA waits after receiving the Handshake input before data acquisition.

See how to access the Trigger Dialog
Meas (External) Trigger dialog box help

Learn how to External Trigger during Calibration

Main Trigger Input

Global / Channel Trigger Delay  After an external trigger is received, the start of the sweep is held off for this specified amount of time plus any inherent latency.

When Trigger Scope = Channel, the delay value is applied to the specified channel.

When Trigger Scope = Global, the same delay value is applied to ALL channels.

Source  The PNA accepts Trigger IN signals through the following rear-panel connectors:

- Meas Trig IN BNC  (PNA-X ONLY)
- Handler I/O Pin 18  (PNA-L and PNA-X ONLY)
- I/O 1 (TRIG IN) BNC  (PNA-L and E836x ONLY)
- Aux I/O - pin 19  (PNA-L and E836x ONLY)

Level / Edge

High Level  The PNA is triggered when it is armed (ready for trigger) and the TTL signal at the select input is HIGH.

Low Level  The PNA is triggered when it is armed (ready for trigger) and the TTL signal at the select input...
is LOW.

**Positive Edge**  After the PNA arms, it will trigger on the next positive edge. If **Accept Trigger Before Armed** is set, PNA will trigger as soon as it arms if a positive edge was received since the last data was taken.

**Negative Edge**  After the PNA arms, it will trigger on the next negative edge. If **Accept Trigger Before Armed** is set, PNA will trigger as soon as it arms if a negative edge was received since the last data was taken.

**Note:** Edge triggering is NOT available on the following PNA models: E835xA, E880xA, N338xA, E8362A, E8363A, E8364A.

**Accept Trigger Before Armed**  When checked, as the PNA becomes armed (ready to be triggered), the PNA will immediately trigger if any triggers were received since the last taking of data. The PNA remembers only one trigger signal. All others are ignored.

- When this checkbox is cleared, any trigger signal received before PNA is armed is ignored.
- This feature is only available when positive or negative EDGE triggering is selected.
- Configure this setting remotely using **CONTrol:SIGNal** (SCPI) or **ExternalTriggerConnectionBehavior** (COM).

**Ready for Trigger Indicator**

Connector to send the PNA 'Ready' OUT signal.

On the PNA-X, when **Meas Trig IN** is enabled, then **Meas Trig Ready (OUT)** is also enabled.

Choose from:

- **Handler I/O p21**
- **AUX I/O p18**

[See how to access the Trigger Dialog](#)
This TAB appears ONLY on E836X and PNA-L models with a Trig I/O rear-panel connector.

Enable  When checked, the PNA sends synchronized trigger signals out the rear-panel I/O (TRIG OUT) BNC connector.

Channel

- **Global** - Trigger output properties apply for ALL channels. This is the default setting and is consistent with pre-7.2 release behavior. In this mode, the Per Point setting (below) is not allowed, but is coupled to the channel **Point trigger** property. In other words, when a channel is in point sweep mode, the trigger output will be sent per point.

- **Channel** Trigger output properties are channel dependent. To output trigger signals for each point, check Per Point (see below).

**Note:** This Channel / Global setting can be changed ONLY by using the following Preference commands:

- SCPI  `Trig:Pref:AIGlobal`
- COM - `AuxTriggerIsGlobal Property`

The current setting is annotated at the bottom of the dialog as:

Compatibility Mode on: Aux Trigger Scope = global
### AUX (I/O) TRIG OUT (To Device)

**Polarity**  The trigger pulse output from the PNA is either in the Positive or Negative direction.

**Position**  The trigger pulse output is sent either BEFORE or AFTER data is acquired.

**Per Point**  *(Channel mode only)* Check to cause a trigger output to be sent for each data point. Clear to send a trigger output for each sweep. This setting controls the trigger output signal regardless of the channel Point trigger setting, which causes the PNA channel to trigger per point. For example, to trigger the PNA channel per point, and output a trigger signal per point, both this, and the channel setting must be checked ON.

**Pulse Duration**  Specifies the duration of the positive or negative output trigger pulse.

Learn how to calibrate while in External Trigger

### Note:
Beginning with PNA Rev 6.0, Guided and Unguided Calibration CAN be performed in External Trigger mode. With this optional behavior, while Trigger Source is set to External, trigger signals must be sent for Calibration sweeps. This behavior does not apply to FCA calibrations.

To revert to pre-6.0 behavior, (the PNA calibrates using Internal trigger signals while Trigger Source is set to External), send these SCPI or COM commands. You can send SCPI commands using the GPIB console.

The following dialog box appears on the PNA screen while the PNA is waiting for an External trigger signal.

Click **Abort** to cancel the wait for a trigger signal.

---

Last Modified:

- 3-Mar-2008  Many edits
- 25-Jan-2007  MX New topic
About the trigger model

Read [Text description](#) of triggering behaviors.

This model does not include the new [Sweep trigger mode](#).
Data Format and Scale

A data format is the way the PNA presents measurement data graphically. Pick a data format appropriate to the information you want to learn about the test device.

- How to set Format
- Rectangular (Cartesian) Display Formats
- Polar
- Smith Chart
- Scale, Reference Level and Position
- Magnitude Offset

See other 'Setup Measurements' topics

How to set the Display Format

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press FORMAT</td>
<td>1. Click Trace</td>
</tr>
<tr>
<td>2. then Active Entry keys</td>
<td>2. then Format</td>
</tr>
<tr>
<td>For <strong>PNA-X</strong> and <strong>'C'</strong> models</td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press FORMAT</td>
<td>1. Click Response</td>
</tr>
<tr>
<td></td>
<td>2. then Format</td>
</tr>
</tbody>
</table>
Rectangular Display Formats

Seven of the nine available data formats use a rectangular display to present measurement data. This display is also known as Cartesian, X/Y, or rectilinear. The rectangular display is especially useful for clearly displaying frequency response information of your test device.

- Stimulus data (frequency, power, or time) appears on the X-axis, scaled linearly.
- Measured data appears on the Y-Axis.

Log Mag (Logarithmic Magnitude) Format

- Displays Magnitude (no phase)
- Y-axis: dB
- Typical measurements:
  - Return Loss
Insertion Loss or Gain

**Phase Format**

- Displays Phase (no magnitude)
- Y-axis: Phase (degrees)
- The trace 'wraps' every 180 degrees for easier scaling.
- Typical Measurements:
  - Deviation from Linear Phase

**Unwrapped Phase**

- Same as Phase, but without 180 degree wrapping.

**Group Delay Format**

- Displays signal transmission (propagation) time through a device
- Y-axis: Time (seconds)
- Typical Measurements:
  - Group Delay

See also:

- Comparing the PNA Delay Functions.
- Phase Measurement Accuracy

**Linear Magnitude Format**

- Displays positive values only
- Y-axis: Unitless (U) for ratioed measurements
  Watts (W) for unratioed measurements.
- Typical Measurements:
  - reflection and transmission coefficients (magnitude)
  - time domain transfer

**SWR Format**

- Displays reflection measurement data calculated from the formula \((1+ r)/(1- r)\) where \(r\) is reflection coefficient.
- Valid only for reflection measurements.
- Y axis: Unitless
Typical Measurements:
  - SWR

**Real Format**

- Displays only the real (resistive) portion of the measured complex data.
- Can show both positive and negative values.
- Y axis: Unitless
- Typical Measurements:
  - time domain
  - auxiliary input voltage signal for service purposes

**Imaginary Format**

- Displays only the imaginary (reactive) portion of the measured data.
- Y-axis: Unitless
- Typical Measurements:
  - impedance for designing matching network

**Polar Format**

Polar format is used to view the magnitude and phase of the reflection coefficient ($r$) from your $S_{11}$ or $S_{22}$ measurement.

You can use Markers to display the following:

- Linear magnitude (in units) or log magnitude (in dB)
- Phase (in degrees)
- The dashed circles represent reflection coefficient. The outermost circle represents a reflection coefficient (G) of 1, or total reflected signal. The center of the circle represents a reflection coefficient (G) of 0, or no reflected signal.

- The radial lines show the phase angle of reflected signal. The right-most position corresponds to zero phase angle, (that is, the reflected signal is at the same phase as the incident signal). Phase differences of 90°, ±180°, and -90° correspond to the top, left-most, and bottom positions on the polar display, respectively.

**Smith Chart Format**

The Smith chart is a tool that maps the complex reflection coefficient (G) to the test device's impedance.

In a Smith chart, the rectilinear impedance plane is reshaped to form a circular grid, from which the series resistance and reactance can be read (R + jX).

You can use Markers to display the following:

- Resistance (in units of ohms)
- Reactance as an equivalent capacitance (in units of farads) or inductance (in units of henrys)

**Inverse Smith Chart** *(also known as Admittance)*

Same as standard Smith Chart, except:

- The plot graticule is reversed right-to-left.
Admittance (in units of siemens) instead of resistance.

Interpreting the Smith Chart

- Every point on the Smith Chart represents a complex impedance made up of a real resistance (r) and an imaginary reactance (r+\text{j}X)
- The horizontal axis (the solid line) is the real portion of the impedance - the resistance. The center of the horizontal axis always represents the system impedance. To the far right, the value is infinite ohms (open). To the far left, the value is zero ohms (short)
- The dashed circles that intersect the horizontal axis represent constant resistance.
- The dashed arcs that are tangent to the horizontal axis represent constant reactance.
- The upper half of the Smith chart is the area where the reactive component is positive and therefore inductive.
- The lower half is the area where the reactive component is negative and therefore capacitive.

Scale, Reference Level and Position

The Scale, Reference Level and Reference Position settings (along with format) determine how the data trace appears on the PNA screen.
# How to set Scale, Reference Level, and Position

## Using front-panel HARDKEY [softkey] buttons

<table>
<thead>
<tr>
<th>N5230A and E836xA/B models</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>SCALE</strong></td>
<td>1. Click <strong>Scale</strong></td>
</tr>
<tr>
<td>2. then <strong>Active Entry</strong> keys</td>
<td>2. then <strong>Scale</strong></td>
</tr>
</tbody>
</table>

## For PNA-X and 'C' models

<table>
<thead>
<tr>
<th></th>
<th>Programming Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>SCALE</strong></td>
<td></td>
</tr>
<tr>
<td>2. then <strong>Scale</strong></td>
<td></td>
</tr>
<tr>
<td>3. then <strong>Scale</strong></td>
<td></td>
</tr>
</tbody>
</table>

## Scale

![Scale Screen](image)

- **Per Division**: 10.000 dB
- **Reference Level**: 0.000 dB
- **Position**: 5.00 Div

**OK**  **Cancel**  **Help**
### Scale dialog box help

#### Scale

- **Per Division**  Sets the value of the vertical divisions of a rectangular display format. In Polar and Smith Chart formats, scale sets the value of the outer circumference. Range: 0.001 dB/div to 500 dB/div

- **Autoscale** - Automatically sets value of the vertical divisions and reference value to fit the ACTIVE data trace within the grid area of the screen. The stimulus values and reference position are not affected.
  
  The analyzer determines the smallest possible scale factor that will allow all the displayed data to fit onto 80 percent of the vertical grid.

  The reference value is chosen to center the trace on the screen.

- **Autoscale All**  Automatically scales ALL data traces in the ACTIVE WINDOW to fit vertically within the grid area of the screen.

#### Reference

- **Level**  In rectangular formats, sets the value of the reference line, denoted by \( 0 \) on the PNA screen. Range: -500 dB to 500 dB.

  In Polar and Smith chart formats, reference level is not applicable.

- **Position**  In rectangular formats, sets the position of the reference line. Zero is the bottom line of the screen and ten is the top line. Default position is five (middle).

  In Polar and Smith chart formats, reference position is not applicable.

### Magnitude Offset

Magnitude Offset allows you to offset the magnitude (not phase) data by a fixed and / or sloped value in dB. If the display format is Linear Magnitude or Real (unitless), the conversion from dB is performed and the correct amount of offset is implemented.

#### How to set Magnitude Offset

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>For N5230A and E836x/A/B models</em></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. <strong>Navigate using MENU/DIALOG</strong></td>
<td>1. Click <strong>Scale</strong></td>
</tr>
<tr>
<td></td>
<td>2. then <strong>Magnitude Offset</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>For PNA-X and 'C' models</th>
<th><strong>Programming Commands</strong></th>
</tr>
</thead>
</table>
1. Press **Scale**
2. then [**More**]
3. then [**Magnitude Offset**]

---

**Magnitude Offset** dialog box help

The Magnitude offset setting affects only the active trace.

**Offset** Offsets the entire data trace by the specified value.

**Slope** Offsets the data trace by a value that changes with frequency. The offset slope begins at 0 Hz.

For your convenience, the offset value at the start frequency is calculated and displayed.

See where this operation is performed in the data processing chain.

---

Last modified:

9/12/06  Added link to programming commands
9/27/06  MX Added UI
Pre-configured Measurement Setups

- Pre-configured setups for NEW measurements
- Pre-configured arrangements for EXISTING measurements

Before reading this topic, it is important to understand Traces, Channels, and Windows in the PNA.

See other 'Setup Measurements' topics

Pre-configured Setups for NEW Measurements

Each of the following setups creates new traces. Existing traces and their settings will be lost, unless you first save them.

<table>
<thead>
<tr>
<th>How to select a pre-configured measurement setup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using front-panel HARDKEY [softkey] buttons</td>
</tr>
<tr>
<td>For N5230A and E836xA/B models</td>
</tr>
<tr>
<td>1. Press MEASURE SETUPS</td>
</tr>
<tr>
<td>2. then Active Entry keys</td>
</tr>
<tr>
<td>For PNA-X and 'C' models</td>
</tr>
<tr>
<td>1. Press RESPONSE</td>
</tr>
<tr>
<td>2. then [Display]</td>
</tr>
<tr>
<td>3. then [Meas Setups]</td>
</tr>
</tbody>
</table>

The following are the four pre-configured measurement setups:
**Arranging Existing Measurements**

The following arrangements place EXISTING measurements into pre-configured Window arrangements using a sort algorithm.

<table>
<thead>
<tr>
<th>How to select an Existing measurement arrangement</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Using front-panel HARDKEY [softkey] buttons</strong></td>
</tr>
</tbody>
</table>

For **N5230A** and **E836xA/B** models

1. Press [ ]

2. then **Active Entry** keys

For **PNA-X** and **'C' models**

1. Press **Display**

| **PNA Menu using a mouse** |

1. Click **Window**

2. then **Arrange**

| **Programming Commands** |

**Overlay Arrangement**

This configuration places all existing traces in a single window, all overlaid on each other.
Stack 2 Arrangement
This configuration places all existing traces in two "stacked" windows.

Split 3 Arrangement
This configuration places all existing traces in three windows, two on top and one below.

Quad 4 Arrangement
This configuration places all existing traces in four windows, one window in each screen quadrant.
Sort Algorithm

The sort algorithm for the Arrange Windows feature is designed to:

- Divide traces among windows based on their properties
- Group traces with common properties

The algorithm sorting is based on the following trace properties, in order of priority:

1. Format: circular (polar or Smith) versus rectilinear (log mag, lin mag, group delay, etc.)
2. Channel number
3. Transmission versus reflection

Note: The PNA traces per window limitation overrides this algorithm. An error occurs if the arrange selection cannot be completed with the current number of traces on the screen.

Last modified:

9/27/06   MX Added UI
9/12/06   Added link to programming commands
Path Configurator

Allows you to configure hardware components that are available with selected PNA-X options.

### How to access Path Configurator

<table>
<thead>
<tr>
<th>Using HARDKEY [softkey] buttons:</th>
<th>PNA Menu using a mouse:</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</td>
<td><strong>PNA-X</strong> and <strong>’C’</strong> models</td>
</tr>
<tr>
<td>1. Not Available</td>
<td>1. Not Available</td>
</tr>
<tr>
<td><strong>For PNA-X</strong> and <strong>’C’</strong> <strong>models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>TRACE/CHAN</strong></td>
<td>1. Click <strong>Trace/Chan</strong></td>
</tr>
<tr>
<td>2. then <strong>[Channel]</strong></td>
<td>2. then <strong>Channel</strong></td>
</tr>
<tr>
<td>3. then <strong>[More]</strong></td>
<td>3. then <strong>Hardware Setup</strong></td>
</tr>
<tr>
<td>4. then <strong>[Hardware Setup]</strong></td>
<td>4. then <strong>Path Config...</strong></td>
</tr>
<tr>
<td>5. then <strong>[Path Config...]</strong></td>
<td></td>
</tr>
</tbody>
</table>

The following image shows configuration with PNA-X Opt 423 (4-port, internal 2nd source, combiner, and mechanical switches). Your PNA-X may not include these options.

[See other RF path configuration diagrams.](#)
Path Configuration dialog box help

Different paths can be configured for each channel.
See Noise Figure tab help of the Path Configuration.

Configuration
Select, store, and delete factory configurations or user-defined configurations. Configurations are stored on the PNA hard drive.

Any configuration can be saved, and later recalled, from this dialog. Click Store, type a configuration name, then click OK.

Text area Displays text describing the physical connections required to complete the configuration. The text for factory configurations can NOT be edited. Text is saved as part of the configuration.

Cancel Closes the dialog and returns the configuration settings to the state they were in when the dialog was opened. Cancel does NOT undo Store and Delete actions that were performed while the dialog was open.

Notes
- Click or touch anywhere within a box to actually cycle through the available settings.
- Some switch settings alter graphics in areas other that where the switch is thrown.
- If you don't hear switches clicking, this could be why:
• Electronic switches are orange on the path configuration dialog. These switches do not make noise when being thrown. Mechanical switches are blue.

• The channel is in hold and not sweeping.

• PNA switch wear prevention logic does NOT allow mechanical switching with continuous triggering. To override the logic use group or single triggering. Learn more.

• Orange lines are jumpers on the front or rear panel.

• Notice on the block diagrams:

  • Extra filtering is available to optimize harmonics below 3.2 GHz on OUT1 of both sources. These filters are not used in the Hi Pwr setting. See specifications.

  • Each source optionally has pulse modulation capability.

• Copy channel feature copies path configuration settings.

• Saved and recalled as part of an instrument state.
Customize the PNA Screen

You can customize your PNA screen by showing or hiding the following display elements. All of these selections are made from the PNA View menu.

- **Status Bar**
- **Toolbars**
- **Tables**
- **Measurement Display**
- **Data and Memory Trace**
- **Title Bars**
- **Minimize Application**

Learn about using pre-configured measurements and windows arrangements
Learn about Traces, Channels, and Windows on the PNA

See other 'Setup Measurements' topics

**Status Bar**

| Status | CH 1: S11 | No Cor |

When enabled, the status bar is displayed along the bottom of the PNA screen. The primary status bar shows the following:

- **Channel Trigger State** (Cont, Groups, Single, Hold)
- Active channel
- Measurement parameter for the active trace
- **Trace Math**
- **Error correction** for the active trace
- **Averaging Factor** for the active channel
- **Smoothing** Percentage
- **Transform** (On)
- **Gating** (On)
- IF Gating Enabled for Pulsed App: (G)
- Manual IF Filtering for Pulsed App: (F)
- Delay if invoked using **Phase Offset**, **Electrical Delay**, or **Port Extensions**.
- Loss if invoked using **Magnitude Offset** or **Port Extensions**.
- GPIB status: Local (LCL), Remote Talker Listener (RMT), or System Controller (CTL).
- Error Status: (LVL, LCK, etc)

**Note:** A second level status bar appears when using **External Test Set Control** or **Interface control**.

The status bar state (ON or OFF) will not change when the PNA is Preset.

### How to display the Status Bar

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using <strong>MENU/DIALOG</strong></td>
<td>1. Click <strong>View</strong></td>
</tr>
<tr>
<td></td>
<td>2. then <strong>Status Bar</strong></td>
</tr>
<tr>
<td>For <strong>PNA-X</strong> and <strong>'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>DISPLAY</strong></td>
<td>1. Click <strong>Response</strong></td>
</tr>
<tr>
<td>2. then <strong>[More]</strong></td>
<td>2. then <strong>Display</strong></td>
</tr>
<tr>
<td>3. then <strong>[Status Bar]</strong></td>
<td>3. then <strong>Status Bar</strong></td>
</tr>
</tbody>
</table>

### Toolbars

You can display up to five different toolbars to allow you to easily set up and modify measurements.
### How to display toolbars

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click <strong>View</strong></td>
</tr>
<tr>
<td>2. then <strong>Toolbar</strong></td>
<td>2. then <strong>Toolbar</strong></td>
</tr>
<tr>
<td>3. then the toolbar to turn ON/OFF</td>
<td>3. then the toolbar to turn ON/OFF</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>DISPLAY</strong></td>
<td>1. Click <strong>Response</strong></td>
</tr>
<tr>
<td>2. then <strong>[More]</strong></td>
<td>2. then <strong>Display</strong></td>
</tr>
<tr>
<td>3. then <strong>[Toolbars]</strong></td>
<td>3. then <strong>More</strong></td>
</tr>
<tr>
<td></td>
<td>4. then <strong>Toolbars</strong></td>
</tr>
</tbody>
</table>

**List of toolbars:**

- **Active Entry**
- **Markers**
- **Measurement**
- **Sweep Control**
- **Stimulus**
- **Time Domain**
- **Port Extension**
- **All Off**

**Note:** There is also a Cal Set toolbar available for Monitoring Error Terms.

**Active Entry Toolbar (For N5230A and E836xA/B models)**

The active entry toolbar is displayed at the top of the screen, below the menu bar. It allows you to make selections from the active function using the mouse or by pressing the front panel key with the corresponding color.
Learn more about using the front panel interface

Entry Toolbar (For PNA-X and 'C' models)

When used with softkeys, this area allows numeric values to be entered for PNA-X settings. From the keyboard, enter G for Giga, M for Mega or milli, K for kilo, and so forth.

Markers Toolbar

The markers toolbar allows you to set up and modify markers. It shows:

- Marker number
- Stimulation value
- Marker functions:
  - Delta
  - Start/Stop
  - Center/Span

Tip: To use the Front Panel Knob to change marker position, first click the Stimulus field of the marker toolbar. Then turn the knob.

Learn more about Markers

Measurement Toolbar

The measurement toolbar allows you to create a new trace for a desired S-parameter measurement in a current window or new window.

Sweep Control Toolbar

In left to right order, the buttons on this toolbar set the active channel to:

- Hold mode
- Single sweep, then Hold mode
- Continuous sweep

Learn more about Channel Trigger State.
**Stimulus Toolbar**

The stimulus toolbar allows you to view, set up, and modify the sweep stimulus. It shows the:

- **Start** value
- **Stop** value
- **Number of points**

**Time Domain**

The Time Domain toolbar allows you to do the following:

- Turn Transform and Gating ON / OFF
- Change the Start / Stop times for both Transform and Gating
- **More**... launches the Time Domain Transform dialog box
- **X** Closes the toolbar

The front panel **Tab** key steps through all of the settings on all of the toolbars on the display. If Tab does not work, press one of the Active Toolbar (color) keys.

**Port Extension**

The Port Extension toolbar allows you to set Port Extensions while viewing the measurement trace. Learn more about [Port Extensions](#).

**All Off**

This allows you to **hide all toolbars** with a single selection.

**Tables**

Tables are displayed at the bottom of the selected window. Only one table may be displayed at a time for a window.
### How to display tables

Using front-panel HARDKEY [softkey] buttons

<table>
<thead>
<tr>
<th>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click <strong>View</strong></td>
</tr>
<tr>
<td>2. then <strong>Tables</strong></td>
<td>2. then <strong>Tables</strong></td>
</tr>
<tr>
<td>3. then the table to turn ON/OFF</td>
<td>3. then the table to turn ON/OFF</td>
</tr>
</tbody>
</table>

For **PNA-X** and **'C' models**

| 1. Press **DISPLAY**                    | 1. Click **Response** |
| 2. then [**More**]                      | 2. then **Display**   |
| 3. then [**Tables**]                    | 3. then **More**      |
|                                        | 4. then **Tables**    |

### List of tables:

- **Marker Table**
- **Limit Line Table**
- **Segment Table**

### Marker Table

You can display a table of marker settings. These settings include the:

- Marker number
- Marker reference (for delta measurements)
- Frequency
- Time and Distance (for Time Domain measurements)
- Response

Learn more about **Markers**

### Limit Line Table
You can display, set up, and modify a table of limit test settings. These include:

- Type (MIN, MAX, or OFF)
- Beginning and ending stimulus values
- Beginning and ending response values

Learn more about Limit Lines

---

**Segment Sweep Table**

You can display, set up, and modify a table of segment sweep settings. These include:

- State (On/Off)
- Start and Stop frequencies
- Number of Points
- IF Bandwidth (if independent levels)
- Power Level (if independent levels)
- Sweep Time (if independent levels)

Learn more about Segment sweep

---

**Measurement Display Items**

**How to show and hide Measurement Display items**

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>N5230A</strong> and <strong>E836x A/B</strong> models</td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click View</td>
</tr>
<tr>
<td></td>
<td>2. then Meas Display</td>
</tr>
<tr>
<td></td>
<td>3. then the display item to show/hide</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>For <strong>PNA-X</strong> and <strong>'C'</strong> models</th>
<th><strong>Programming Commands</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>DISPLAY</strong></td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then <strong>[Display Items]</strong></td>
<td>2. then Display</td>
</tr>
<tr>
<td></td>
<td>3. then <strong>Display Items</strong></td>
</tr>
</tbody>
</table>
Measurement Display items:

- **Trace Status**
- **Frequency Stimulus**
- **Marker Readout**
- **Limit Test Results**
- **Limit Lines**
- **Title**

---

**Trace Status**

For **N5230A** and **E836xA/B models**

Trace status buttons are displayed to the left of each window. 
The depressed button indicates the Active Trace. 
Click to select a trace.

For **PNA-X** and **'C' models**

Trace status is annotated at the top of each window. 
The highlighted trace number indicates the Active Trace. 
Click to select a trace.

Trace Status shows the following:

- Trace number (Tr x). This is the trace number of the channel; NOT the window trace number which is used in many programming commands.
- Measurement parameter. This can be replaced with a custom Trace Title.
- Format
- Scaling factor
- Reference level

---

**How to show/hide Trace Status**

---

**Frequency/Stimulus**

| Ch1: Start 300.000 kHz | Stop 3.00000 GHz |

Frequency/stimulus information is displayed at the bottom of each window on the screen. It shows:

- Channel number
How to show/hide Frequency/Stimulus information

**Marker / Bandwidth / Trace Statistics Readout**

How to show/hide Readout settings

The Marker / Bandwidth / Trace Statistics Readout area, in the upper-right corner of each window, can contain up to **20** total readout lines. However, all readout lines may not be visible depending on the window size and whether Large Marker Readout is enabled.

![Readout](image)

The image shows 3 readout lines.

- Markers use one readout line per marker.
- Marker Bandwidth and Trace Statistics use three readout lines per trace.

**Marker Readout**

- Checked - Shows readout information.
- Cleared - Shows no readout information.

**One Readout Per Trace**

- Checked - Shows the readout of only the active marker for each trace.
- Cleared - Shows up to 20 total readouts lines.

**Large Marker Readout**

This setting also controls Trace Statistics readout.

- Checked - Shows the marker readout in large font size for easy reading.
- Cleared - Shows the marker readout in normal font size.

Learn more about Markers

---

**Limit Line Test Results**
Limit line test results, **Pass** or **Fail**, are displayed on the right side of the designated window.

**Limit Lines**

Limit lines are displayed for the active trace in the designated window. Their position depends on:

- Limit levels
- Format
- Scaling
- Reference level

Learn more about [Limit Lines](#)

[How to show/hide Limit Lines and Results](#)

---

**Title**

You can create and display a title for each **window** using the keyboard. You can also use the following Title Entry dialog box.

![Title Entry Dialog Box](image)

The title is displayed in the upper-left corner of the selected window.

To clear a title, delete the title from the dialog box entry area and click OK.

See also [Trace Titles](#)

[How to show/hide a Title](#)

---

**Data Trace and Memory Trace**

You can view or hide the active data or memory trace.

- Make a trace active by clicking the trace status button

To view a memory trace you must first store a trace in memory. Click **Trace**, then **Math / Memory**, then **Data => Memory**.

Learn more about [Math operations](#)

---

**Title Bars**

The Title bar shows the window number and Minimize / Maximize icons.
• Checked - Title bars for all PNA windows are shown.
• Cleared - Title bars for all PNA windows are hidden. This allows more room to display measurement results.

How to show/hide the Title Bars

Using front-panel HARDKEY [softkey] buttons

For N5230A and E836xA/B models

1. Navigate using MENU/DIALOG

For PNA-X and 'C' models

1. Press DISPLAY
2. then [More]
3. then [Title Bars]

PNA Menu using a mouse

1. Click View
2. then Title Bars

Minimize Application

The Network Analyzer application can be minimized to show the desktop and Windows taskbar.

How to minimize the Network Analyzer Application

Using front-panel HARDKEY [softkey] buttons

For N5230A and E836xA/B models

1. Navigate using MENU/DIALOG

For PNA-X and 'C' models

1. Press DISPLAY

PNA Menu using a mouse

1. Click View
2. then Title Bars
3. then More
4. then Title Bars

1. Click File
2. then [Windows]
3. then [More]
4. then [Minimize]

To restore the PNA application, click the PNA application on the Windows taskbar.

Last modified:

27-Aug-2007   Edited readout section
9/12/06        Added link to programming commands
9/27/06        MX Added UI
Copy Channels

You can copy the channel settings from an existing channel to a new or another existing channel.

- Why Copy Channels
- How to Copy Channels
- List of Channel Settings

Other Setup Measurements Topics

Why Copy Channels

Copy channel settings if you need to create several channels that have slightly different settings.

For example, if you have an amplifier that you want to characterize over a frequency span with several different input power levels.

Follow these steps:

1. Create one measurement with your optimized channel settings.
2. Copy that channel to new channels.
3. Change the power level on the new channels.

The alternative to using Copy Channels is to create new default measurements on new channels. Then change every channel setting to your new requirement. This is very time consuming and thus shows the benefit of the Copy Channels feature.

Note: Copy Channels does NOT work with any of the PNA Applications, such as FCA, Gain Compression, or Noise Figure.
# How to Copy Channels

## Using front-panel HARDKEY [softkey] buttons

<table>
<thead>
<tr>
<th>PNA Menu using a mouse</th>
<th>Programming Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>For PNA-X and 'C' models</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Press TRACE/CHAN</td>
</tr>
<tr>
<td>2. then <strong>Channel</strong></td>
<td>2. then <strong>Channel</strong></td>
</tr>
<tr>
<td>3. then <strong>More</strong></td>
<td>3. then <strong>Copy Channel</strong></td>
</tr>
<tr>
<td>4. then <strong>Copy Channel</strong></td>
<td></td>
</tr>
</tbody>
</table>

## Using PNA Menu using a mouse

<table>
<thead>
<tr>
<th>Programming Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. Click Channel</strong></td>
</tr>
<tr>
<td><strong>2. then Copy Channel</strong></td>
</tr>
</tbody>
</table>

![Copy Channel Dialogue Box](image)
Copy Channel dialog box help

Copies an existing channel's settings to another channel.

**Copy channel:** Select a channel to copy.

**to:** Scroll to select a channel to copy settings to. Channel numbers that are currently being used are highlighted. They can be selected and overwritten.

**Notes:**

- You can copy channel settings to ONLY one new or existing channel. Repeat this operation to copy to more than one channel.

- The new channel is ALWAYS copied to the Active window. If you want the new channel in its own window, first create a new measurement in a new window. Then make sure it is the Active window before you copy the channel into it.

- The measurement in the new channel becomes the active measurement.

- Only the channel settings are copied. The measurement trace is NOT copied to the new channel.

  - If measurements already exist on a channel being copied to, the measurements on that channel will not change, but they will assume the new channel settings.

  - If a NEW channel is copied TO, an S11 measurement is created in order to view the channel settings.

For example:

1. **Existing** channel 1: S21 measurement
2. **Copy** channel 1 to NEW channel 2
3. **Result:** channel 2: S11 measurement with channel 1 copied settings. Both measurements are in the same window. The S11 measurement is the active measurement.

For more information see [Traces, Channels, and Windows on the PNA](#)

---

**List of Channel Settings**

- **Frequency Span**
- **Power**
- **Cal Set usage**
- **Source Power Cal data**
- **IF Bandwidth**
- **Number of Points**
- **Sweep Settings**
- **Average**
- **Trigger (some settings)**

Last modified:

13-Feb-2008   Added note about Apps

9/12/06       Added link to programming commands
ADC Measurements

The PNA is equipped with one or more ADC (Analog to Digital Converter) inputs. These ADC inputs can be used as measurement receivers and display measurements on the PNA screen.

- Analog Inputs can be used for measuring from -10V to +10V. These inputs can be considered auxiliary receivers and used in a similar way as S-Parameter receivers.
- Analog Output Sense inputs (AOS1 and AOS2) can be used to measure the corresponding DAC outputs.
- Analog Ground input (AG1) can be used to measure the instruments analog ground (PNA-X only).

Supported Hardware
PNA-X:  Power I/O connector
Other models: Aux I/O connector

How to create ADC receiver measurements

For PNA-X and 'C' models

1. Press TRACES
2. then [New Trace]

Programming Commands

1. Click Trace/Chan
2. then New Trace
New Trace (ADC) dialog box help

**Note:** Sweep speed slows dramatically when measuring more than two ADC receivers.

On the [New Trace dialog](#), click the **Receivers** tab.

**Activate** - check any empty line to create a trace.

**Numerator** - select from the following:

- **AI1** - Input 1 (PNA-X only)
- **AI2** - Input 2
- **AOS1, AOS2** - Output sense 1 or 2
- **AIG** - Analog ground (PNA-X only)

**Denominator** - NOT available (ONLY unratioed measurements)

**Source Port** - The ADC receiver is measured when the specified source port is sweeping.

---

ADC receiver traces are labeled as shown in the following images:

- The ADC1 input is being measured, with 2 as the source port.
- The Y axis is U (unitless).
- The default trace **format** is Real (linear).

---

**ADC Traces and other useful PNA functions**

Although most PNA functions work with ADC traces, the following may be especially useful.

- **Equation Editor** can be used with the trace data. Although the PNA-X ADC is measuring voltage (-10V to +10V range in 14 bits), by using a trace formula, this voltage can represent other types of measurement parameters (such as current, temperature, or a scaled voltage). See PAE example.

- **Trace averaging** and **Trace Smoothing** can be used to remove trace noise.

- **Dwell time** can be used to allow for settling.

---

**PNA Functions Not Supported**

- Calibration for ADC receivers is NOT supported.
- Use with FCA is NOT supported.
While the PNA is sweeping an ADC measurement, do NOT use the rear-panel Analog I/O SCPI commands.
Setting System Impedance

The system impedance can be changed for measuring devices with an impedance other than 50 ohms, such as waveguide devices. The PNA mathematically transforms and displays the measurement data as though the PNA ports were the specified impedance value. Physically, the test ports are always about 50 ohms.

How to change the System Impedance

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click System</td>
</tr>
<tr>
<td></td>
<td>2. then Configure</td>
</tr>
<tr>
<td></td>
<td>3. then System Z0</td>
</tr>
<tr>
<td><strong>For PNA-X and ‘C’ models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press SYSTEM</td>
<td>1. Click Utility</td>
</tr>
<tr>
<td>2. then [Configure]</td>
<td>2. then System</td>
</tr>
<tr>
<td>3. then [System Z0]</td>
<td>3. then Configure</td>
</tr>
<tr>
<td></td>
<td>4. then System Z0</td>
</tr>
</tbody>
</table>

![System Z0](image)
### System Z0 dialog box help

Allows you to change the system impedance (default setting is 50 ohms).

**Z0** Displays the current system impedance.

**For 75 ohm devices:**

1. Change the system Z0 to 75 ohms.
2. Connect minimum loss pads (75 ohm impedance) between the analyzer and the DUT to minimize the physical mismatch.
3. Perform a calibration with 75 ohm calibration standards.

**For waveguide devices:**

1. Change the system Z0 to 1 ohm.
2. Perform a calibration with the appropriate waveguide standards.

---

Last modified:

<table>
<thead>
<tr>
<th>Date</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>9/27/06</td>
<td>MX Added UI</td>
</tr>
<tr>
<td>9/12/06</td>
<td>Added link to programming commands</td>
</tr>
</tbody>
</table>
**Dynamic Range**

Dynamic range is the difference between the analyzer receiver's maximum input power and the minimum measurable power (noise floor). For a measurement to be valid, input signals must be within these boundaries.

Increasing dynamic range is important if you need to measure very large variations in signal amplitude, such as filter bandpass and rejection. The dynamic range is shown below for an example measurement.

![Dynamic Range Diagram]

To help reduce measurement uncertainty, the analyzer dynamic range should be greater than the response that the DUT exhibits. For example, measurement accuracy is increased when the DUT response is at least 10 dB above the noise floor. The following methods can help you increase the dynamic range.

- **Increase the Device Input Power**
- **Reduce the Receiver Noise Floor**
- **Use the Front-Panel Jumpers (if your PNA has a configurable test set)**

### Other topics about Optimizing Measurements

#### Increase Device Input Power

Increase the DUT input power so that the analyzer can more accurately detect and measure the DUT output power. However, use caution - too much power can damage the analyzer receiver or cause compression distortion.

**Caution! Receiver input damage level: +15 dBm.**

[See how to increase input power to the device](#).

**Tip:** You can further increase dynamic range by using an external booster amplifier to increase the input power to the DUT. See [High Power Amplifier Measurements](#).

#### Reduce the Receiver Noise Floor

You can use the following techniques to lower the noise floor and increase the analyzer's dynamic range.

- Reduce crosstalk between the PNA receivers when measuring signals close to the noise floor. See [Receiver Crosstalk](#).
- Use **Sweep Averaging** - learn more about [Sweep Average](#).
Reduce the **IF Bandwidth** - learn more about [IF Bandwidth](#).

- In [Segment sweep](#) mode each segment can have its own IF bandwidth. For example, when measuring a filter:
  - In the passband, the IF bandwidth can be set wider for a fast sweep rate, as long as high-level trace noise is kept sufficiently small.
  - In the reject band, where noise floor contributes significantly to measurement error, the IF bandwidth can be set low enough to achieve the desired reduction in average noise level.

**Use the Front-Panel Jumpers (if your PNA has the configurable test set)**

If your PNA has FOUR front-panel jumpers, you can bypass the test-port couplers and apply signals directly into the receivers. See [Dynamic Range - 4 Jumpers](#). Using this configuration, you can achieve up to 143 dB dynamic range with **Response Calibration** using segment sweep mode.

If your PNA has MORE THAN FOUR front-panel jumpers ([Configurable Test Set](#)), you can use the front-panel jumpers to reverse a test-port coupler. See [Dynamic Range - Configurable Test Set Option](#). Using this configuration, you can achieve up to 143 dB dynamic range with **Full 2-port Calibration** using segment sweep mode.

**Note:** Bypassing a port's directional coupler increases the port mismatch by approximately 15 dB (the coupling factor of the directional coupler).

- For information about upgrading your PNA to include front-panel jumpers, see [PNA Options](#).
- Discover the measurement possibilities using [front-panel jumpers](#).
Improving Dynamic Range with FOUR front-panel jumpers

To improve dynamic range you can bypass the test-port coupler and apply the signal directly into the receiver. As shown in the following graphic, the signal is applied to the front-panel connector for the B In or Rcvr B In front-panel jumper rather than Port 2. Using this configuration, you can achieve up to 143 dB dynamic range with response calibration using segment sweep mode.

Note: Your PNA may not be equipped with front-panel jumpers or all of the components shown in this block diagram.
To improve dynamic range you can reverse the signal path in the test-port coupler and bypass the loss typically associated with the coupled arm. As shown in the following graphic, the signal is applied to Port 2. The signal bypasses the coupled arm via the jumper cable connected to the Coupler Thru (or Coupler In) and the Receiver B In (or B In) ports. Using this configuration, you can increase the forward measurement dynamic range up to 143 dB with full 2-port calibration using segment sweep mode. When making full 2-port error corrected measurements, the reverse measurement is degraded by 15 dB, with up to 113 dB of dynamic range available.

Explore the graphic with your mouse.

**Note:** Your analyzer’s block diagram may contain different components than shown below.
**Number of Points**

A data point is a sample of data representing a measurement at a single stimulus value. You can specify the number of data points that the PNA measures across a sweep. (A "sweep" is a series of consecutive data point measurements, taken over a sequence of stimulus values.)

The PNA sweep time changes proportionally with the number of points. However, the overall measurement cycle time does not. See Technical Specifications for more information on how the number of points, and other settings, affect the sweep time.

**Note:** You may experience a significant decrease in computer processing speed with increased number of points, number of traces, and calibration error terms (full 2-port or 3-port). If this becomes a problem, you can increase the amount of RAM with PNA Option 022.

### How to change the number of data points

Select a number or click Custom to invoke a dialog box.

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For PNA-L and E836x models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press <strong>Sweep Setup</strong></td>
<td>1. Click <strong>Sweep</strong></td>
</tr>
<tr>
<td>2. then [F1]</td>
<td>2. then <strong>Number of Points</strong></td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press <strong>Sweep</strong></td>
<td>1. Click <strong>Stimulus</strong></td>
</tr>
<tr>
<td>2. then [<strong>Number of Points</strong>]</td>
<td>2. then <strong>Sweep</strong></td>
</tr>
<tr>
<td></td>
<td>3. then <strong>Number of Points</strong></td>
</tr>
</tbody>
</table>

![Number of Points dialog box](image)
**Number of Points** dialog box help

Specifies the number of data points that the analyzer gathers during a measurement sweep. You can specify any number from 1 to 20,001. The default value is 201.

Two data points are required for Time Domain.

**Tips:**

- To achieve the greatest trace resolution, use the maximum number of data points.
- For faster throughput use the smallest number of data points that will give you acceptable resolution.
- To find an optimized number of points, look for a value where there is not a significant difference in the measurement when you increase the number of points.
- To ensure an accurate measurement calibration, perform the calibration with the same number of points that will be used for the measurement.

---

Last modified:

- 14-Dec-2007  Decreased min to 1
- 21-Jun-2007  MX Increased maximum
- 9/12/06      Added link to programming commands
Phase Measurement Accuracy

You can increase the accuracy of phase measurements by using the following PNA features.

- **Electrical Delay**
- **Phase Offset**
- **Spacing Between Frequency Points (Aliasing)**

See Also

Port Extensions
Comparing the PNA Delay Functions.

Learn more about Phase measurements

---

**Electrical Delay**

Electrical delay is a mathematical function that simulates a variable length of lossless transmission line. Use the electrical delay feature to compensate for the linear phase shift through a device. This feature allows you to look at only the deviation from linear phase of the device.

You can set the electrical delay independently for each measurement trace.

### How to set Electrical Delay

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY (softkey) buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xB models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Scale</td>
</tr>
<tr>
<td></td>
<td>2. then Electrical Delay</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press SCALE</td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then [Electrical Delay]</td>
<td>2. then Scale</td>
</tr>
<tr>
<td></td>
<td>3. then Electrical Delay</td>
</tr>
</tbody>
</table>
**Electrical Delay** dialog box help

**Electrical Delay**  Specifies the value of delay added or removed, in units of time. This compensates for the linear phase shift through a device. You can set the electrical delay independently for each measurement trace.

**Velocity Factor**  Specifies the velocity factor that applies to the medium of the device that was inserted after the measurement calibration. The value for a polyethylene dielectric cable is 0.66 and 0.7 for Teflon dielectric. 1.0 corresponds to the speed of light in a vacuum.

Velocity factor can also be set from the Port Extensions toolbar / More settings and Time Domain Distance Marker Settings.

**Media**

- **Coax**  select if the added length is coax. Also specify the velocity factor of the coax.
- **Waveguide**  Select if the added length is waveguide. Also specify the low frequency cutoff of the waveguide.
- **Cutoff Freq**  Low frequency cutoff of the waveguide.

Learn about Electrical Delay (scroll up)

**Phase Offset**

**Phase** offset mathematically adjusts the phase measurement by a specified amount, up to 360°. Use this feature in the following ways:

- **Improve the display of a phase measurement.** This is similar to the way you would change the reference level in an amplitude measurement. Change the phase response to center or align the response on the screen.

- **Emulate a projected phase shift in your measurement.** For example, if you know that you need to add a cable and that the length of that cable will add a certain phase shift to your measurement, you can use phase offset to add that amount and simulate the complete device measurement.
How to set Phase Offset

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Scale</td>
</tr>
<tr>
<td></td>
<td>2. then Phase Offset</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press SCALE</td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then [Phase Offset]</td>
<td>2. then Scale</td>
</tr>
<tr>
<td></td>
<td>3. then Phase Offset</td>
</tr>
</tbody>
</table>

**Phase Offset**

Phase Offset dialog box help

**Phase Offset** Type a value or use the up and down arrows to select any value up to 360 degrees.
Learn about Phase Offset (scroll up)

**Spacing Between Frequency Points (Aliasing)**

The analyzer samples data at discrete frequency points, then connects the points, creating a trace on the screen.

If the phase shift through a device is >180° between adjacent frequency points, the display can look like the phase slope is reversed. This is because the data is undersampled and aliasing is occurring.

If you are measuring group delay and the slope of the phase is reversed, then the group delay will change sign. For example, the following graphic shows a measurement of a SAW bandpass filter.

- The left measurement has 51 points and indicates the group delay is negative, which is a physical impossibility. That is, the response is below 0 seconds reference line.
- The right measurement shows an increase to 201 points which indicates the group delay is positive. That is, the response is above the 0 seconds reference line.
**Tip:** To check if aliasing might be occurring in a measurement, either increase the number of points or reduce the frequency span.

Last modified:

- March 10, 2008   MX Added UI
- Sept. 12, 2006   Added link to programming commands
Electrically-Long Device Measurements

A signal coming out of a device under test may not be exactly the same frequency as the signal going in to a device at a given instant in time. This can sometimes lead to inaccurate measurement results. You can choose between two techniques to eliminate this situation and increase measurement accuracy.

- **Why Device Delay May Create Inaccurate Results**
- **Solutions to Increase Measurement Accuracy**
  - **Slow the Sweep Speed**
  - **Add Electrical Length to the R Channel**

### Other topics about Optimizing Measurements

**Why Device Delay May Create Inaccurate Results**

The following graphic shows an example of this situation:

- In the network analyzer, the source and receiver are phase locked together and sweep simultaneously through a span of frequencies.

- The signal flow through the Device Under Test (DUT) is shown as different colors for different frequencies.

- You can see as a stimulus frequency travels through the DUT, the analyzer tunes to a new frequency just before the signal arrives at the receiver. This causes inaccurate measurement results.

If the analyzer is measuring a long cable, the signal frequency at the end of the cable will lag behind the network analyzer source frequency. If the frequency shift is appreciable compared to the network analyzer's IF detection bandwidth (typically a few kHz), then the measured result will be in error by the rolloff of the IF filter.

**Note:** There is no fixed electrical length of a device where this becomes an issue. This is because there are many variables that lead to measurement speed. When high measurement accuracy is critical, lower the sweep speed until measurement results no longer change.
Solutions to Increase Measurement Accuracy

Choose from the following methods to compensate for the time delay of an electrically long device.

- **Slow the Sweep Speed**
- **Add Electrical Length to the R Channel**

**Slow the Sweep Speed**

The following methods will slow the sweep speed.

- **Increase the Sweep Time**
- **Increase the Number of Points**
- **Use Stepped Sweep**
- **Set Dwell Time**

**Add Electrical Length to the R Channel**

**Note:** This method applies to PNA models with front panel loops.

Instead of slowing the sweep, you can compensate for the electrical length of a cable or fixture.

a. Remove the R-channel jumper on the front panel of the analyzer.

b. Replace the jumper with a cable of about the same length as the device under test.

   1. Add the cable on the R1 channel for S11 and S21 measurements.
   2. Add the cable on the R2 channels for S22 and S12 measurements.

c. Set the analyzer for a fast sweep.

**Configuration for S22 and S12 Measurements**

![Configuration Diagram]

This method balances the delays in the reference and test paths, so that the network analyzer's ratioed transmission measurement does not have a frequency-shift error.

**Note:** This method works well if the delay is in a cable or fixture. For devices with long delays, this method is only suitable for uncalibrated measurements.
Reflection Accuracy on Low-Loss 2-Port Devices

To make accurate reflection measurements that have a 1-port calibration, you should terminate the unmeasured port.

- **Why Terminate the Unmeasured Port**
- **How to Terminate the Unmeasured Port**
- **Resulting Measurement Uncertainty**

### Other topics about Optimizing Measurements

### Why Terminate the Unmeasured Port

A 2-port calibration corrects for all 12 twelve error terms. A 1-port calibration corrects for directivity, source match and frequency response, but not load match. Therefore, for highest accuracy, you must make the load match error as small as possible. This especially applies for low-loss, bi-directional devices such as filter passbands and cables. You do not need to be concerned with load match when you are measuring a device with high reverse isolation, such as an amplifier.

### How to Terminate the Unmeasured Port

Use one of the following methods:

- Connect a high-quality termination load (from a calibration kit, for example) to the unmeasured port of your device. This technique yields measurement accuracy close to that of a Full SOLT 2-port calibration.

- Connect the unmeasured port of your device directly to the analyzer, inserting a 10 dB precision attenuator between the device output and the analyzer. This improves the effective load match of the analyzer by approximately twice the value of the attenuator, or 20 dB.

### Resulting Measurement Uncertainty

The following graph illustrates the measurement uncertainty that results from terminating **with** and **without** a
precision 10 dB attenuator on the output of the test device.

Legend

- Filter Reflection
- Uncertainty with attenuator
- Uncertainty without attenuator

The calculations below show how adding a high-quality 10 dB attenuator improves the load match of the analyzer.

**Note:** The corresponding linear value is shown in parentheses.

**Network Analyzer:**

\[
\begin{align*}
\text{Load match (NALM)} & = 18 \text{ dB (.126)} \\
\text{Directivity (NAD)} & = 40 \text{ db (.010)}
\end{align*}
\]

**Filter:**

\[
\begin{align*}
\text{Insertion loss (FIL)} & = 1\text{ dB (.891)} \\
\text{Return loss (FRL)} & = 16 \text{ dB (.158)}
\end{align*}
\]

**Attenuator:**

\[
\begin{align*}
\text{Insertion loss (AIL)} & = 10 \text{ dB (.316)} \\
\text{SWR (ASWR)} & = 1.05 \text{ (.024)}
\end{align*}
\]

\[
32.26 \text{ dB Return Loss}
\]

**Calculations:**

```plaintext
394
```
<table>
<thead>
<tr>
<th></th>
<th>Without Attenuator</th>
<th>With Attenuator</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Attenuator</strong></td>
<td>NA</td>
<td>NA + Atttn.</td>
</tr>
<tr>
<td><strong>Worst Case Error</strong></td>
<td>= NA</td>
<td>= NA + Atttn.</td>
</tr>
<tr>
<td></td>
<td>= .1</td>
<td>= .1 + .019</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= .029</td>
</tr>
<tr>
<td><strong>Uncertainty Adds</strong></td>
<td>= -20log(FRL)+(EWC)+(NAD)</td>
<td>= -20log(FRL)+(EWC)+(NAD)</td>
</tr>
<tr>
<td></td>
<td>= -20log(.158)+(100)+(0.010)</td>
<td>= -20log(.158)+(0.029)+(0.010)</td>
</tr>
<tr>
<td></td>
<td>= 11.4 dB</td>
<td>= 14.1 dB</td>
</tr>
<tr>
<td><strong>Uncertainty Subtr</strong></td>
<td>= -20log(FRL)-(EWC)-(NAD)</td>
<td>= -20log(FRL)-(EWC)-(NAD)</td>
</tr>
<tr>
<td></td>
<td>= -20log(.158)-(100)-(0.010)</td>
<td>= -20log(.158)-(0.029)-(0.010)</td>
</tr>
<tr>
<td></td>
<td>= 26.4 dB</td>
<td>= 18.5 dB</td>
</tr>
</tbody>
</table>
Measurement Stability

There are several situations that can cause unstable measurements. To ensure that you are making repeatable measurements, you can use various methods to create a stable measurement environment.

- **Frequency Drift**
- **Temperature Drift**
- **Inaccurate Measurement Calibrations**
- **Device Connections**

### Other topics about Optimizing Measurements

**Frequency Drift**

The analyzer frequency accuracy is based on an internal 10 MHz frequency oscillator. See Technical Specifications for stability and aging specifications.

If your measurement application requires better frequency accuracy and stability, you can override the internal frequency standard and provide your own high-stability external frequency source through the 10 MHz Reference Input connector on the rear panel.

**Temperature Drift**

Thermal expansion and contraction changes the electrical characteristics of the following components:

- Devices within the analyzer
- Calibration kit standards
- Test devices
- Cables
- Adapters

To reduce the effects of temperature drift on your measurements, do the following.

- Switch on the analyzer 1/2 hour before performing a measurement calibration or making a device measurement.
- One hour before you perform a measurement calibration, open the case of the calibration kit and take the standards out of the protective foam.
- Use a temperature-controlled environment. All specifications and characteristics apply over a 25 °C ±5 °C range (unless otherwise stated).
- Ensure the temperature stability of the calibration kit devices.
Avoid handling the calibration kit devices unnecessarily during the calibration procedure.

Ensure the ambient temperature is ±1°C of the measurement calibration temperature.

**Inaccurate Measurement Calibrations**

If a measurement calibration is inaccurate, you will not measure the true response of a device under test. To ensure that your calibration is accurate, you should consider the following practices:

- Perform a measurement calibration at the points where you connect the device under test, that is, the reference plane.

- If you insert any additional accessory (cable, adapter, attenuator) to the test setup after you have performed a measurement calibration, use the port extensions function to compensate for the added electrical length and delay.

- Use calibration standards that match the definitions used in the calibration process.

- Inspect, clean, and gage connectors. See Connector Care.

See Accurate Measurement Calibrations for more detailed information.

**Device Connections**

Good connections are necessary for repeatable measurements. To help make good connections, do the following:

- Inspect and clean the connectors for all of the components in the measurement setup.
- Use proper connection techniques.
- Avoid moving the cables during a measurement.
Noise Reduction Techniques

Random electrical noise which shows up in the analyzer receiver chain can reduce measurement accuracy. The following PNA functions help reduce trace noise and the noise floor which can lead to better dynamic range and more accurate measurements.

**Note:** The trace noise in microwave PNAs becomes worse below 748 MHz and is especially obvious between 10 MHz and 45 MHz. See Reduce IFBW.

- **Sweep Average**
- **IF Bandwidth**
- **Trace Smoothing**

**See Also**
- Increase Dynamic Range
- PNA data processing map.

**Other topics about Optimizing Measurements**

**Sweep Average**

Sweep average is a feature that reduces the effects of random noise on a measurement. The PNA computes each data point based on the average of the same data point over several consecutive sweeps. You determine the number of consecutive sweeps by setting the Average factor. The higher the average factor, the greater the amount of noise reduction.

- **Average Counter** appears on the screen when Averaging is ON, displaying the number of sweeps that has been averaged. The effect on the signal trace can be viewed as the Average Factor increases. This can assist in the selection of the optimum number of sweep averages.

- **Channel wide** - Averaging is applied to all measurements in a channel. The Average counter is displayed for each channel.

- **Unratioed** measurements - Although you can average unratioed (single receiver) measurements, you may get unexpected results:
  - Phase results may tend toward 0. This is because phase measurements are relative by nature. Measuring absolute phase with a single receiver appears random. Averaging random positive and negative numbers will tend toward 0.
  - The noise floor does not drop when averaging unratioed measurements as on ratioed measurements.

- **Average vs IF Bandwidth** - Both can be used for the same benefit of general noise reduction. For minimizing very low noise, using Average is more effective than reducing system bandwidth. Generally, Averaging takes slightly longer than IF bandwidth reduction to lower noise, especially if many averages are required. Also, changing the IF bandwidth after calibration results in uncertain accuracy.
- **Calibration** - Because averaging is a mathematical process that occurs after the raw measurement is made, averaging can be turned ON before, or after, calibration without invalidating the error correction terms. If averaging is ON before calibration, the measurement of calibration standards are averaged measurements. More sweeps are needed to perform the calibration, but there will be less noise in the resulting error correction terms. Subsequent corrected measurements will also have less noise error. In addition, noise is further reduced by turning Averaging ON after calibration. See the PNA data processing map.

- **Point-averaging** - The PNA does NOT have a “point-averaging” feature like the Agilent 8510 network analyzer. That feature measures and averages each data point BEFORE moving to the next data point. Therefore, all data points are averaged in a single, slower sweep. To accomplish similar results with the PNA, try lowering the IFBW.

### Effects of Sweep Average

![Without Average](Without_Average.png) ![With Average](With_Average.png)

### How to Set Averaging

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Channel</td>
</tr>
<tr>
<td>2. then <strong>Average</strong></td>
<td>2. then <strong>Avg</strong></td>
</tr>
</tbody>
</table>

| **For PNA-X and 'C' models**               | **Programming Commands**    |
| 1. Press **Avg**                           | 1. Click **Response**       |
| 2. then [Averaging]                        | 2. then **Avg**             |
|                                            | 3. then **Average**         |
Average dialog box help

Average ON
Checked - Averaging is applied
Cleared - Averaging is NOT applied

Average Factor  Specifies the number of sweeps that is averaged. Range of 1 to 65536 (2^16).

Restart  Begins a new set of measurements that are used for the average. This set of measurements is equal to the average factor.

Learn about Averaging (scroll up)

IF Bandwidth

The PNA converts the received signal from its source to a lower intermediate frequency (IF). The bandwidth of the IF bandpass filter is adjustable from 40 kHz (for most PNA models) down to a minimum of 1 Hz.

Reducing the IF receiver bandwidth reduces the effect of random noise on a measurement. Each tenfold reduction in IF bandwidth lowers the noise floor by 10 dB. However, narrower IF bandwidths cause longer sweep times.

- **Channel wide** - IF bandwidth can be set independently for each channel
- **Segment sweep** - IF bandwidth can be set independently for each segment of segment sweep.
- **Calibration** - Changing the IF bandwidth after calibration will cause a 'C-delta' correction level, which means that calibration accuracy is uncertain.

Effect of Reducing IF Bandwidth

10 Hz IF BW

3000 Hz IF BW
How to set IF Bandwidth

Using front-panel HARDKEY [softkey] buttons

<table>
<thead>
<tr>
<th>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Navigate using <strong>MENU/DIALOG</strong></td>
<td>1. Click <strong>Sweep</strong></td>
</tr>
<tr>
<td>2. then <strong>IF Bandwidth</strong></td>
<td>2. then <strong>IF Bandwidth</strong></td>
</tr>
</tbody>
</table>

For **PNA-X** and **'C' models**

| 1. Press **Avg**                      | 1. Click **Response**        |
| 2. then **[IF Bandwidth]**            | 2. then **Avg**              |
|                                        | 3. then **IF Bandwidth**     |

**IF Bandwidth**

**IF Bandwidth** dialog box help

**IF Bandwidth**  Specifies the IF (receiver) bandwidth. The value of IF bandwidth is selected by scrolling through the values available in the IF bandwidth text box. The IF BW is set independently for each channel.

The list of selectable IF Bandwidths is different depending on PNA model.

The following values are common to all models:

- 1 | 2 | 3 | 5 | 7 | 10 | 15 | 20 | 30 | 50 | 70 | 100 | 150 | 200 | 300 | 500 | 700 | 1k | 1.5k | 2k | 3k | 5k | 7k | 10k | 15k | 20k | 30k

In addition, the following values are PNA Model specific:

5230A Opt 020, 025, 120, 125, 140, 145, 146, 240, 245, 146

- 50k | 70k | 100k | 150k | 200k | 280k | 360k | 600k

N5230A Opt 220, 225, 420, 425, 520, 525:
The following limitations apply for the highlighted IFBW settings (1 MHz and above).

Note: These wider IFBWs do NOT provide faster sweep speeds. They are used to make wideband pulsed measurements.

- Dwell time is not allowed.
- Sweep times that are slower than the default value are not allowed.
- Step sweep mode only - NOT available in Analog sweep.
- External Trigger Delay is not allowed.
- Number of points for CW sweep is limited to 1001.
- A slight shift (.1dB) in Log Mag traces may be seen when switching in and out of these bandwidths.

**Reduce IF BW at Low Frequencies**

On PNA models with a maximum frequency of 20 GHz and higher, the trace noise becomes worse below 748 MHz. This is especially obvious between 10 MHz and 45 MHz and also when Time Domain is ON. See PNA models / maximum frequencies.

When this box is checked, the PNA uses a smaller IF Bandwidth than the selected value at frequencies below 748 MHz.

This setting:

- can be made for each channel.
- is ON (checked) by default.
- also applies to segment sweep.
- is NOT available on 4-port PNA-L (model N5230A Opt 240 and 245).

Use the following calculations to determine the actual IF Bandwidth value that is used below 748 MHz.

If the result is NOT a selectable IF BW value, the next higher selectable value is used.
<table>
<thead>
<tr>
<th>Less than 20 GHz models:</th>
<th>10 MHz to 44.999999 MHz</th>
<th>45 MHz to 748 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

**ALL 2-port 20 GHz PNA models:**

Actual IF BW = (selected IF BW) x (.05)  
Actual IF BW = selected IF BW (No reduction)

**ALL 40 GHz and higher models:**

Actual IF BW = (selected IF BW) x (.025)  
Actual IF BW = (selected IF BW) x (.5)

**PNA-X Models**

<table>
<thead>
<tr>
<th>Start Freq</th>
<th>Stop Freq</th>
<th>Actual IF BW = (selected IF BW) x n</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz</td>
<td>19 MHz</td>
<td>n = .05</td>
</tr>
<tr>
<td>19 MHz+</td>
<td>53 MHz</td>
<td>n = .1</td>
</tr>
<tr>
<td>53 MHz+</td>
<td>75 MHz</td>
<td>n = .5</td>
</tr>
<tr>
<td>75 MHz+</td>
<td>26.5 GHz</td>
<td>n = 1</td>
</tr>
</tbody>
</table>

+ indicates plus 1 Hz

**Example:**

On a 67 GHz PNA, the selected IF BW is 30 KHz.

With **Reduce IF BW at Low Frequencies** checked, the actual IF Bandwidths used are:

- From **10 MHz to 44.999999 MHz**: 30,000Hz * .025 = 750 Hz (PNA uses next higher selectable value: 1000 Hz.)
- From **45 MHz to 748 MHz**: 30,000Hz * .5 = 15 KHz
- From **748 MHz** to stop sweep: 30 KHz

**OK** Selects the value of IF bandwidth shown in the text box.

Learn about IF Bandwidth (scroll up)

---

**Trace Smoothing**

Trace smoothing averages a number of adjacent data points to smooth the displayed trace. The number of adjacent data points that get averaged together is also known as the smoothing aperture. You can specify aperture as either the number of data points or the percentage of the x-axis span.

Trace Smoothing reduces the peak-to-peak noise values on broadband measured data. It smooths trace noise and does not increase measurement time significantly.

Because Trace Smoothing follows Format in the PNA data processing map, the formatted data is smoothed.
Smoothing is automatically turned off if the format is Polar or Smith Chart.

Learn more about Data Format Types.
See the PNA data processing map.

Tips:

- Start with a high number of display points and reduce until you are confident that the trace is not giving misleading results.
- Do not use smoothing for high-resonance devices, or devices with wide trace variations. It may introduce misleading information.
- Smoothing is set independently for each trace.

**Effects of Smoothing on a Trace**

![Without Smoothing](image1.png) ![With Smoothing](image2.png)

**How to set Trace Smoothing**

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Trace</td>
</tr>
<tr>
<td></td>
<td>2. then Smoothing</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press Avg</td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then [Smoothing]</td>
<td>2. then Avg</td>
</tr>
<tr>
<td></td>
<td>3. then Smoothing</td>
</tr>
</tbody>
</table>
Smoothing dialog box help

**Smoothing ON**  When checked, applies smoothing to the displayed trace.

**Percent of Span**  Specify percent of the swept stimulus span to average. For example, for a trace that contains 100 data points, and specify a percent of span = 11%, then the number of data points that are averaged is 11.

**Points**  Specify the number of adjacent data points to average.

[Learn about Trace Smoothing](#) (scroll up)

Last modified:

Sept.12, 2006  Added link to programming commands
Crosstalk

Crosstalk is energy leakage between analyzer signal paths. This can be a problem with high-loss transmission measurements. Although the crosstalk specification of the PNA is exceptional, you can reduce the effects of crosstalk by doing the following:

- **Set the Sweep to Alternate**
- **Perform an Isolation Calibration**

### Other topics about Optimizing Measurements

## Set the Sweep to Alternate

Alternate sweep measures only one receiver per sweep. When one receiver is measured, the analyzer switches off the other receiver. This helps reduce receiver crosstalk.

Learn how to set Alternate Sweep.

### Notes

- Alternate sweep mode is set independently for each measurement channel. Therefore, if multiple measurement channels are in use, you may want to set Alternate sweep for each channel.
- When more than one receiver is being used to make measurements, the Alternate Sweep setting doubles the sweep cycle time.
- The PNA noise floor has to be lowered substantially before crosstalk is visible. You may need to use averaging or narrow the IF bandwidth.

## Perform an Isolation Calibration

For transmission measurements, a response and isolation measurement calibration helps reduce crosstalk because the analyzer measures and then subtracts the leakage signal during the measurement calibration. The calibration improves isolation so that it is limited only by the noise floor.

**Note:** Isolation is never performed on a Smart (Guided) Calibration. Learn more.

Generally, the isolation error falls below the noise floor. So when you are performing an isolation calibration you should use a noise reduction technique such as sweep averages or reducing the IF bandwidth.
Effects of Accessories

Accessories in a configuration may affect the results of a device measurement. You can choose between two analyzer features that reduce the effects of accessories.

- **Power Slope to Compensate for Cable Loss**
- **Gating to Selectively Remove Responses**

Other topics about Optimizing Measurements

Power Slope to Compensate for Cable Loss

If you have a long cable or other accessory in a measurement configuration where a power loss occurs over frequency, apply the power slope function. This function increases the analyzer source power by a rate that you define (dB/GHz).

1. In the **Channel** menu, click **Power**.
2. If the slope function is not already switched on, click the **Slope** check box.
3. In the **dB/GHz** box, enter the rate that you want the source power to increase over the frequency sweep. Click **OK**.

Gating to Selectively Remove Responses

Gating is a feature in the time domain (option 010) that allows the analyzer to mathematically remove responses. You can set the gate for either a reflection or transmission response, but you will see different results.

- **Gating a reflection response** isolates a desired response (such as a filter's return loss), from unwanted responses (such as adapter reflections or connector mismatches).
- **Gating a transmission response** isolates a specific path in a multipath device that has long electrical lengths.

See **Time Domain Gating** for more information.
Achieve Fastest Sweep

You can achieve the fastest measurement sweep by adjusting the following:

- **Sweep Settings**
- **Noise Reduction Settings**
- **Measurement Calibration Choice**
- **Unnecessary Functions**

### Other topics about Optimizing Measurements

#### Sweep Settings

Consider changing each of the following settings as suggested.

- **Frequency Span** - Measure only the frequencies that are necessary for your device.
- **Segment Sweep** - Use segments to focus test data only where you need it.
- **Switch Off Stepped Sweep** - Use linear swept mode to minimize sweep time when possible.
- **Auto Sweep Time** - Use this default to sweep as quickly as possible for the current settings.
- **Number of Points** - Use the minimum number of points required for the measurement.

For more information on how number of points and other settings affect sweep cycle time, see [Technical Specifications](#).

#### Noise Reduction Settings

Using a combination of these settings, you can decrease the sweep time while still achieving an acceptable measurement.

- **IF Bandwidth**. Use the widest IF bandwidth that will produce acceptable trace noise and [dynamic range](#).
- **Average**. Reduce the average factor, or switch Average off.

#### Measurement Calibration Choice

Choose the appropriate type of calibration for the required level of accuracy.

When full 2-port error correction is applied, the PNA takes both forward and reverse sweeps to gather all 12 error correction terms. This occurs even with a single S11 measurement displayed. All displayed measurements are updated as the second sweep is performed. Both sweeps are performed using the specified sweep time.
When calibrating greater than 2 ports, the following formula is used to determine the number of sweeps required:

- \( N \times (N-1) \) where \( N \) = the number of ports.

When full 3-port calibration is applied, 6 sweeps are required; forward and reverse for each port pair. With full 4-port correction, 12 sweeps are required, and so forth.

To limit the measurement time, perform ONLY the level of calibration that your measurements require. For example, if making only an S11 measurement, perform a 1-port calibration on that port.

Sweep speed is about the same for uncorrected measurements and measurements done using a response calibration, or one-port calibration. For more information see Select a Calibration.

**Unnecessary Functions**

The analyzer must update information for all active functions. To achieve an additional increase in sweep speed, switch off all of the analyzer functions that are not necessary for your measurement application.

- **Delete Unwanted Traces**
- **Switch Off Unwanted Markers**
- **Switch Off Smoothing**
- **Switch Off Limit Testing**
- **Switch Off Math Functions**

Analyzer sweep speed is dependent on various measurement settings. Experiment with the settings to get the fastest sweep and the measurement results that you need.
Switch Between Multiple Measurements

If you need to make multiple measurements to characterize a device, you can use various methods to increase throughput. Experiment with these methods to find what is best for your measurement application needs.

- **Set Up Measurements for Increased Throughput**
  - Arrange Measurements in Sets
  - Use Segment Sweep
  - Trigger Measurements Selectively
- **Automate Changes Between Measurements**
- **Recall Measurements Quickly**

Other topics about Optimizing Measurements

Set Up Measurements for Increased Throughput

To achieve optimum throughput of devices that require multiple measurements, it is helpful to know the operation of the PNA. This knowledge allows you to set up the measurement scenarios that are best for your applications.

Learn more about Traces, Channels, and Windows on the PNA

Arrange Measurements in Sets

If you arrange measurements to keep the complete set of device measurements in one instrument state, you can save them so that you can later recall a number of measurements with one recall function.

See Pre-configured Measurement Setups for more information.

Use Segment Sweep

Segment sweep is helpful if you need to change the following settings to characterize a device under test.

- Frequency Range
- Power Level
- IF Bandwidth
- Number of Points

The segment sweep allows you to define a set of frequency ranges that have independent attributes. This allows you to use one measurement sweep to measure a device that has varying characteristics.

See Segment Sweep for more information.
**Trigger Measurements Selectively**

You can use the measurement trigger to make measurements as follows:

- Continuously update only the measurements that have rapidly changing data.
- Occasionally update measurements that have infrequently changing data.

For example, if you had four channels set up as follows:

- Two channels measuring the data that is used to tune a filter
- Two channels measuring the data for the out-of-band responses of the filter

You would want to constantly monitor only the measurement data that you use for tuning the filter. If you continuously update all of the channels, this could slow the response of the analyzer so that you would not be able to tune the filter as effectively.

**Note:** You must either trigger the infrequent measurement manually or with remote interface commands.

**To trigger measurements selectively:**

This procedure shows you how to set up two different measurements with the following behavior:

- Channel 1 measurement will continuously update the data.
- Channel 2 measurement will occasionally update the data.

1. In the *Windows* menu, click *Meas Setups, Setup D*.

**Set Up a Measurement Trigger for Continuous Updates**

2. In the *Sweep* menu, click *Trigger, Trigger*.


4. Under *Channel Trigger State*, select *Channel 1*, and click *Continuous*.

**Set Up a Measurement Trigger for Occasional Updates**

5. Under *Channel Trigger State*, select *Channel 2*, and click *Single, OK*.
   - If you want the analyzer to trigger more than a single sweep, click the *Enable Groups* check box and enter the number of sweeps.

6. In the *System* menu, click *Keys, Trigger*.

**Update the Measurement**

7. Click on the lower window to make Channel 2 the *active channel*. 
8. On the active entry toolbar, click the type of trigger you set up.
   - Click **Single** if you set up the analyzer for a single sweep per trigger.
   - Click **Groups** if you set up the multiple sweeps per trigger.

*Note:* A trace must be active for you to initiate a trigger for that measurement.

**Automate Changes Between Measurements**

If there are slight differences between the various measurements that you need to characterize a device, you may find that it is faster to change the measurement settings using programming.

**Recall Measurements Quickly**

The most efficient way to recall measurements is to recall them as a set of measurements (instrument state).

- It only takes a short time longer to recall an instrument state that includes multiple measurements, than it does to recall an instrument state with only one measurement.
- Each recall function has time associated with it. You can eliminate that time by setting up the measurements as a set so you can recall them as a set.

See [Save and Recall Files](#) for more information.
Data Transfer Speed

When testing devices remotely using COM or SCPI, the following techniques can be used to transfer data quickly between the PNA and remote computer, helping you achieve the best measurement throughput.

- **Use single sweep (trigger) mode** to ensure that a measurement is complete before starting a data transfer.

- **Transfer the minimum amount of data** needed. For example, a trace with a few points, using segment sweep rather than a full trace with many linearly spaced points. Also, use markers instead of trace transfers.

- **Choose the REAL data format** to provide the fastest transfer speed when using SCPI programs for automated applications.

- **Use SCPI over LAN** for applications that are automated with SCPI programs.

- **Use COM programs** to provide the fastest transfer speed when using an automated application. See [Data Transfer Time](#) specifications.

![Data Transfer Speed Comparison](chart.png)
Using Macros

Macros are executable programs that you write, load into the analyzer, and then run from the analyzer. You can have up to 12 macros set up to run on the analyzer.

- **How to Setup Macros**
- **How to Run Macros**
- **Macro Example**

### How to Setup Macros

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836x A/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press Menu + Macro</td>
<td>1. Click System</td>
</tr>
<tr>
<td>2. Press</td>
<td>2. then Macro</td>
</tr>
<tr>
<td></td>
<td>3. then Macro Setup</td>
</tr>
<tr>
<td><strong>For PNA-X and ‘C’ models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press MACRO</td>
<td>1. Click Utility</td>
</tr>
<tr>
<td>2. then [Macro Setup]</td>
<td>2. then Macro</td>
</tr>
<tr>
<td></td>
<td>3. then Macro Setup</td>
</tr>
</tbody>
</table>

**In the Macro Setup dialog box:**

1. Click on a blank line below the last entry. (There may be NO entry.)
2. Click **Edit**
3. In the **Macro Title** box, type a descriptive title for your macro.
4. Click **Browse**.
5. Change **Files of Type**
6. Find and click your macro file
5. Click OK
6. Click OK
7. Click OK on the Macro Setup dialog box.

Macro Setup dialog box help

Allows you to create a set of 12 macros so that you can launch other programs from within the PNA application.

**Note:** To add a Macro, select a blank line then click **Edit**

**Macro Title** Shows the titles that appear in the active entry toolbar when you press the Macro key. These titles are associated with the executable files and should be descriptive so you can easily identify them. For example, if you wanted to launch the Agilent Home Page, you could title the executable "Agilent Home."

**Macro Executable** Lists the complete path to the executable file. To follow the example of launching the Agilent PNA Series Home Page, the path to the executable could be "C:\Program Files\Internet Explorer\iexplore.exe."

**Macro Runstring Parameters** Lists the parameters that get passed to the program that is referenced in the executable file. Again following the example of launching the PNA Series Home Page, you could assign the runstring parameters "http://www.agilent.com/find/pna."

**Edit** Invokes the **Macro Edit dialog box**.

**Delete** Deletes the selected macro.

**Up** Allows you to reorder the macros, moving the selected macro up one line. For the 12 possible macros there are 12 lines, indicating the order that they appear in the active entry toolbar when you press the Macro key. Since there are four titles that can be shown at one time in the toolbar, when you repeatedly press the Macro key, the toolbar changes the macro titles to the next set of four macro titles.

**Down** Moves the selection down one line in the list of macros.
Macro Edit dialog box help

**Macro Title**  Allows you to modify the title that appears in the active entry toolbar.

**Macro Executable**  Allows you to modify the complete path to the macro executable file.

**Browse**  Allows you to look through drives and directories, to locate the macro executable file and establish the complete path to the file.

**Macro run string parameters**  Allows you to modify the parameters that are passed to the program referenced in the executable file.

See Macro Setup dialog box

---

How to Run Macros

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press until your macro is visible</td>
<td>1. Click System</td>
</tr>
<tr>
<td>2. then</td>
<td>2. then Macro</td>
</tr>
<tr>
<td></td>
<td>3. then select the macro to run</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press MACRO</td>
<td>1. Click Utility</td>
</tr>
<tr>
<td>2. then select the macro to run</td>
<td>2. then Macro</td>
</tr>
<tr>
<td></td>
<td>3. then select the macro to run</td>
</tr>
</tbody>
</table>

Macro Example
The following is an example Visual Basic Scripting (vbs) program that you can copy, install, and run on your PNA

**Note:** Print these instructions if viewing in the analyzer. This topic will be covered by the Macro Setup dialog box.

1. Copy the following code into a Notepad file.
2. Save the file on the analyzer hard drive in the **C:\Documents** folder. Name the file **FilterTest.vbs**
3. Close Notepad
4. **Setup the macro in the PNA**
5. **Run the macro**

```
'Start copying here
'This program creates a S21 measurement, with Bandwidth markers for testing a 175MHz Bandpass filter
'It is written in VBscript using COM commands

Set PNA = CreateObject("AgilentPNA835x.Application")
PNA.Preset
Set chan=PNA.activechannel
Set meas=PNA.activemeasurement
Set limits = meas.LimitTest
Set trce = PNA.ActiveNAWindow.ActiveTrace

meas.ChangeParameter "S21",1
chan.StartFrequency = 45e6
chan.StopFrequency = 500e6
trce.ReferencePosition = 8
PNA.TriggerSignal = 3

'Do Test
for t=1 to 5
    call measure
    call compare
next
msgbox("Done Testing")

sub measure
    msgbox("Connect Device " & t & " and press OK")
PNA.ManualTrigger True
meas.SearchFilterBandwidth
end sub

sub compare
    BW = meas.FilterBW
    if bw>6.5e7 then msgbox("Failed BW: " & BW)
    Loss = meas.FilterLoss
    if loss>5 then msgbox("Failed Loss: " & Loss)
end sub

'End copying here
```
Calibration Overview

The following is discussed in this topic:

- **What Is Measurement Calibration?**
- **Why Is Calibration Necessary?**
- **Conditions Where Calibration Is Suggested**
- **What Is ECal?**

### What Is Measurement Calibration?

Calibration removes one or more of the systematic errors using an equation called an error model. Measurement of high quality standards (for example, a short, open, load, and thru) allows the analyzer to solve for the error terms in the error model. See [Measurement Errors](#).

You can choose from different calibration types, depending on the measurement you are making and the level of accuracy you need for the measurement. See [Select a Calibration Type](#).

The accuracy of the calibrated measurements is dependent on the quality of the standards in the calibration kit and how accurately the standards are modeled (defined) in the calibration kit definition file. The calibration-kit definition file is stored in the analyzer. In order to make accurate measurements, the calibration-kit definition must match the actual calibration kit used. To learn more, see [Accurate Calibrations](#).

Calibration Wizard provides the different calibration methods used in the PNA. See [Calibration Wizard](#).

There are quick checks you can do to ensure your measurement calibration is accurate. To learn more see [Validity of a Measurement Calibration](#).

If you make your own custom-built calibration standards (for example, during in-fixture measurements), then you must characterize the calibration standards and enter the definitions into a user modified calibration-kit file. For more information on modifying calibration kit files, see [Calibration Standards](#).

**Note:** Instrument Calibration is ensuring the analyzer hardware is performing as specified. This is not the same as measurement calibration.

### Why Is Calibration Necessary?

It is impossible to make perfect hardware that would not need any form of error correction. Even making the hardware good enough to eliminate the need for error correction for most devices would be extremely expensive.

The accuracy of network analysis is greatly influenced by factors external to the network analyzer. Components of the measurement setup, such as interconnecting cables and adapters, introduce variations in magnitude and phase that can mask the actual response of the device under test.

The best balance is to make the hardware as good as practically possible, balancing performance and cost. Calibration is then a very useful tool to improve measurement accuracy.
Conditions Where Calibration Is Suggested

Generally, you should calibrate for making a measurement under the following circumstances:

- You want the best accuracy possible.
- You are adapting to a different connector type or impedance.
- You are connecting a cable between the test device and an analyzer test port.
- You are measuring across a wide frequency span or an electrically long device.
- You are connecting an attenuator or other such device on the input or output of the test device.

If your test setup meets any of the conditions above, the following system characteristics may be affected:

- Amplitude at device input
- Frequency response accuracy
- Directivity
- Crosstalk (isolation)
- Source match
- Load match

What Is ECAL

ECal is a complete solid-state calibration solution. It makes one port (Reflection), full two and three-port calibrations fast and easy. See Using ECal.

- It is less prone to operator error.
- The various standards (located inside the calibration module) never wear out because they are switched with PIN-diode or FET switches.
- The calibration modules are characterized using a TRL-calibrated network analyzer.
- ECal is not as accurate as a good TRL calibration.

For information about ordering ECal modules, see Analyzer Accessories or contact your Agilent Support Representative.
Calibration Standards

This following section explains the general principles and terms regarding calibration kit files. To learn how to modify calibration kit files, See Modify Calibration Kits.

- About Calibration Kits
- Calibration Standards
- Standard Type
- Standard Definitions
- Class Assignments

See other Calibration Topics

About Calibration Kits

A calibration kit is a set of physical devices called standards. Each standard has a precisely known or predictable magnitude and phase response as a function of frequency.

In order to calibrate the analyzer using the standards in a calibration kit, the response of each standard must be mathematically defined and then organized into standard classes that correspond to the error models used by the analyzer.

To be able to use a particular calibration kit, the known characteristics from each standard in the kit must be stored into analyzer memory. This is done for you with the PNA. All Agilent Cal Kits containing standard definitions are stored in the PNA. For a list of Agilent calibration kits, see Analyzer Accessories.

Calibration Standards

Calibration standards provide the reference for error-corrected measurements in the network analyzer. Each standard has a precisely known definition that includes electrical delay, impedance, and loss. The analyzer stores these definitions and uses them to calculate error correction terms.

During measurement calibration, the analyzer measures standards and mathematically compares the results with "ideal models" of those standards. The differences are separated into error terms that are later removed from device measurements during error correction. See Systematic Errors.

Standard Type

A standard type is one of five basic types that define the form or structure of the model to be used with that standard. The standard types are shown below:
### Standard Definitions

Standard definitions describe the electrical characteristics of the standards and the frequencies they will be used. Standard definitions can be viewed from the Advanced Modify Cal Kit menu selection. Standard definitions include:

- **Minimum Frequency** Specifies the minimum frequency the standard is used for calibration.
- **Maximum Frequency** Specifies the maximum frequency the standard is used for calibration.
- **Z0** Specifies the characteristic impedance of the standard (not the system characteristic impedance or the terminal impedance of the standard).
- **Delay** Specifies a uniform length of transmission line between the standard being defined and the actual calibration plane.
- **Type** Specifies type of standard (SHORT, OPEN, THRU/LINE, LOAD, ARBITRARY).
- **Loss** Specifies energy loss, due to skin effect, along a one-way length of coaxial cable.

**Loss model equation:**

- The value of loss is entered as ohms/second at 1 GHz.
- To compute the loss of the standard, measure the delay in seconds and the loss in dB at 1 GHz. Then use the following formula:

\[
\text{Loss (ohms/sec)} = \frac{\text{Loss (dB)} \times Z_0(f)}{4.3429 \times \text{delay (s)}}
\]

**Capacitance model equation:**

- C0, C1, C2, C3. Specifies the fringing capacitance for the open standard.

\[
C = C0 + (C1 \times F) + (C2 \times F^2) + (C3 \times F^3)
\]

(F is the measurement frequency).
- The terms in the equation are defined when specifying the open as follows:

<table>
<thead>
<tr>
<th>Standard</th>
<th>Terminal Impedance</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHORT</td>
<td>zero ohms</td>
</tr>
<tr>
<td>OPEN</td>
<td>infinite ohms</td>
</tr>
<tr>
<td>LOAD</td>
<td>system impedance, Z0</td>
</tr>
<tr>
<td>THRU/LINE</td>
<td>no terminal impedance</td>
</tr>
<tr>
<td>ARBITRARY</td>
<td>user-defined</td>
</tr>
</tbody>
</table>
- C0 term is the constant term of the third-order polynomial and is expressed in Farads.
- C1 term is expressed in F/Hz (Farads/Hz).
- C2 term is expressed in F/Hz².
- C3 term is expressed in F/Hz³.

**Inductance model equation:**

L₀, L₁, L₂, L₃. Specifies the residual inductance for the short standard.

- \( L = (L₀) + (L₁ \times F) + (L₂ \times F²) + (L₃ \times F³) \)
- (F is the measurement frequency).

The terms in the equation are defined when specifying the short as follows:

- L₀ term is the constant term of the third-order polynomial and is expressed in Henries.
- L₁ term is expressed in H/Hz (Henries/Hz)
- L₂ term is expressed in H/Hz².
- L₃ term is expressed in H/Hz³.

**Class Assignments**

Once a standard is characterized, it must be assigned to a standard "class". A standard class is a group of standards that are organized according to the calibration of the PNA error model.

The number of classes needed for a particular calibration type is equal to the number of error terms being corrected.

A class often consists of a single standard, but may be composed of multiple standards. These may be required for accuracy or to cover a wide frequency range.

**Example:** A response calibration requires only one class, and the standards for that class may include an OPEN, or SHORT, or THRU. A 1-port calibration requires three classes. A 2-port calibration requires 10 classes, not including two for isolation.

The number of standards assigned to a given class may vary from one to seven for unguided calibrations. Guided calibrations allow as many standards as needed.

Calibration Classes are assigned in the Advanced Modify Cal Kit menu selection.

**The different classes used in the PNA:**

**S11A, S11B, S11C (S22A, S22B, S22C and so forth)**

These are the three classes for port 1-reflection calibrations (three classes also for S22 and S33). They are used in the one-port calibrations and the full two-port calibration. They are required in removing the directivity, source match, and reflection tracking errors. Typically, these classes might consist of an open, a short and a load standard for each port.

**Transmission and Match (forward and reverse)**

These classes are used to perform a full two-port calibration. The transmission class relates primarily to the transmission tracking, while the match class refers to load match. For both of these classes, the typical standard is a thru or delay.
Isolation

The isolation classes are used to perform a full two-port and the TRL two-port calibrations. The isolation classes apply to the forward and reverse crosstalk terms in the PNA error model.

TRL THRU

These are used to perform a TRL two-port calibration. The TRL thru class should contain a thru standard or a short line. If it contains a non-zero length thru standard, then the calibration type is called LRL or LRM.

TRL REFLECT

This class is used to perform a TRL two-port calibration. The TRL reflect class should contain a standard with a high reflection coefficient, typically an open or short. The actual reflection coefficient need not be known, but its phase angle should be specified approximately correctly (± 90 deg). The exact same reflection standard must be used on both ports in the TRL calibration process.

TRL LINE or MATCH

These are used to perform a TRL two-port calibration. The TRL line or match class should contain line standards, load standards, or both. If a line standard is used, its phase shift must differ from that of the TRL THRU standard by 20° to 160°. This limits the useable frequency range to about 8 to 1. Two or more line standards of different lengths may be specified to get broader frequency coverage. It is also common to include a load standard for covering low frequencies, where the line's length would be impractically long. When a load is used, the calibration type is called TRM or LRM.

Note: For more information, read application note 8510-5A, "Specifying Calibration Standards for the Agilent 8510 Network Analyzer". Although the application note is written for the Agilent 8510 series of network analyzers, it applies to the PNA as well. The part number for the publication is 5956-4352.
## Calibration Wizard

The Calibration Wizard allows you to choose a Calibration method and then perform the calibration.

- **How to Start Calibration Wizard**
- **Guided Calibration: Mechanical Standards**
- **Unguided Calibration**
- **Saving a Calibration**

### Other Cal Topics

### How to start Calibration Wizard

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press <strong>CAL</strong></td>
<td>1. Click <strong>Calibration</strong></td>
</tr>
<tr>
<td>2. then <strong>Active Entry</strong> keys</td>
<td>2. then <strong>Cal Wizard</strong></td>
</tr>
<tr>
<td><strong>For PNA-X and ’C’ models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press <strong>CAL</strong></td>
<td>1. Click <strong>Response</strong></td>
</tr>
<tr>
<td>2. then <strong>[Start Cal]</strong></td>
<td>2. then <strong>Cal Wizard</strong></td>
</tr>
<tr>
<td>3. then <strong>[Cal Wizard]</strong></td>
<td></td>
</tr>
</tbody>
</table>

![Calibration Wizard: Begin Calibration](image)

- **Programming Commands**

424
Calibration Wizard Begin dialog box help

Select the calibration method:

**SmartCal (Guided Calibration)**

This method provides a step-by-step "wizard" interface. You describe the connectors on your DUT and the cal kits you will use; it walks you through the most accurate calibration possible.

**Note:** SmartCal DOES allow you to measure calibration standards in any order. However, you must click **Next** and **Back** without measuring standards until you get to the standard you want to measure.

- Supports ALL Cals **EXCEPT** simple open, short, and thru response Cals. See Also [TRL Calibration](#).
- Use a different Cal Kit (including ECal) for each port.

**Unguided Calibration**

This method provides a familiar calibration interface, but with limited capability. You choose the type of cal to perform; it allows you the flexibility to measure the standards in any order.

- Supports all Cals **EXCEPT** full 3-port, full 4-port.
- TRL is NOT supported on multiport PNAs.
- Only one Cal Kit can be used.

**Use Electronic Calibration**

- This method provides fast, software-controlled calibrations.
- Only one ECal module can be used. Use SmartCal when more than one ECal module is needed.

**Save Preferences**

- When cleared, you will continue to see this page on subsequent calibrations.
- When checked, saves your calibration method choice and the dialog no longer appears.
- To make this dialog re-appear, click **Calibration**, then **Preferences**.
- Learn more about [Calibration Preferences](#).
The Calibration Window / Channel

During a Guided Calibration, a 'Cal Window' is created for you to view the connection of calibration standards before standards are measured. This Cal Window uses a new Cal channel that is created and duplicates the settings in the channel being calibrated. Correction is ALWAYS OFF for the displayed calibration channel. At the completion of the calibration, the calibration channel and window are deleted.

With PNA Rev. 8.0, the measurement of calibration standards can be performed while viewing any PNA window configuration you choose. The Cal Window is appended to your Custom Cal Window setting, and all windows are visible and sweeping below the Cal Wizard before the Measure (cal standard) button is pressed. The windows to be viewed and channels to be swept during the cal process are specified using SCPI commands. See an example.

The new Cal Window settings do not work in a FCA channel.

SmartCal (Guided Calibration)

Guided Calibration automatically determines the calibration type and suggests a calibration kit that matches your DUT connectors.

Guided Calibration can perform the following Cal Types:

- ALL Cals EXCEPT Open, Short, and Thru Response Cals.
- ECal on one or more ports, beginning with PNA firmware revision 5.24.
- TRL - Learn how to do TRL cals

Programming Commands

Note: SmartCal DOES allow you to measure calibration standards in any order. However, you must click Next and Back without measuring standards until you get to the standard you want to measure.

The PNA displays the following dialog boxes when performing a Guided calibration.
Select Ports for Guided Calibration dialog box help

Allows you to select ports to calibrate.

Cal Type Selection  Select the number of ports to calibrate.

N Port Cal Configuration  If not calibrating all PNA ports, specify which ports to calibrate.

Show Advanced Settings (Orientation & Thru Cal Section)  Available only for ECal.

Back  Return to Cal Wizard Begin dialog. If you did not see the 'Cal Wizard Begin' dialog but want to, click Back, then clear the Save Preferences checkbox.

For greater than 4-port cals, see External Test Set calibration - Select Cal Type.

---

Select DUT Connectors and Cal Kits dialog box help

Allows you to select the connector type and Cal Kit for each DUT port to be calibrated.

Connectors  To change selection, click the connector field for each DUT port.

If your DUT connectors are:

- **Waveguide**  Change the system impedance to 1 ohm before performing a calibration. See Setting System Impedance.

- **Not listed**  You can create your own connector type and calibration kit file. The PNA includes the following example cal kits that can be used as a template. See Calibration kits for more information.
  - If using a gendered (male and female) connector type, select Type A as the connector type.
  - If using a connectorless device such as on-wafer probes., select Type B as the connector type.

Cal Kits  Select the Cal Kit to be used to calibrate each test port. The list for each DUT Port displays kits having the same connector type as the DUT.

Identical ECal models connected?  ECal modules can be distinguished by serial number. This can have implications on your remote SCPI programs.
**85056K**

- To calibrate 2.4 mm connectors using the 85056K cal kit, select 85056D as the cal kit. The 85056K definitions in the PNA are for 2.92mm standards (2.4mm plus 2.92 adapters). The 85056D kit contains exactly the same standards WITHOUT the adapters.

**TRL**

- To perform a TRL Cal, assign a TRL Cal Kit to the lowest port number of each port pair.

- When selecting a TRL Cal Kit on a 4-port PNA, and a Global Delta Match Cal is not available, the Cal type will be set to SOLT and a "Could not find a Global Delta Match Cal." message is displayed on the dialog box. If the selected Cal Kit will not support SOLT, the Next button will not be available. Then you must select a different Cal Kit to proceed or Cancel and perform a Global Delta Match Cal.

**Modify Cal** Check, then click Next, to Modify Cal (Standards AND Thru Method).

For greater than 4-port cals, see External Test Set calibration - Select DUT Connectors.

---

**Error: Frequency Range** dialog box help

The current cal kit does not cover the current frequency range of the measurement. Do one of the following to correct the problem:

- **Frequency** Change the frequency range of the active channel.
- **Edit** Modify the class assignments so that a different standard is selected.
- **Back** Select a different Cal Kit that covers the required frequency range.
- **Cancel** Exit the Cal Wizard
Modify Cal dialog box help

**Thru #n**
Lists the proposed Thru connections to be made during the calibration process. You can change these Thru connections to better suit your test setup.

- The proposed Thru connections are listed automatically.
- Additional Thru connections can be selected for higher accuracy. Learn more.

**Add Thru**
Click to add a Thru connection. Learn more

**Remove Thru**
Select a Thru by clicking the "Thru #N" field or the "1st Port / 2nd Port" field. Then click "Remove Thru". This selection is NOT available if the selected Thru is required for the calibration.

**1st Port / 2nd Port**
Click to select the two ports to be included in the Thru connection. The order of the port numbers is not critical.

**Thru Cal Method**
Lists the available Thru Cal methods for the specified port pairs. Learn about the Thru Cal Method choices.

**Cal Type/ Stds**
Click to invoke the View / Modify Properties of Cal dialog box

**Do orientation** - Appears ONLY if an ECal module is selected for use.

When this box is checked (default) the PNA automatically senses the model and direction in which an ECal module port is connected to the PNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually.

Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range. If you have an E8361A or E836xB PNA and do an ECal completely within 10 - 20 MHz OR 60 - 67 GHz, you may need to do orientation manually. There may not be sufficient power to orient the ECal module at those frequencies.
Choose delta match - Available when a Delta Match Cal is required.

- Check, then click Next to invoke the Select Cal Set for Delta Match dialog box.
- Clear - The Cal Wizard uses the Global Delta Match Cal if available.

View/Detect ECal Characterizations - Appears ONLY if an ECal module is selected for use.
Click to invoke the View ECal Modules and Characterizations dialog box. Displays a list of ECal modules that are connected to the PNA.

View/Modify Properties of Cal for Ports... dialog box help

Select calibration type

Another chance to change the Thru method.

Learn about the Thru Cal Method choices.

Advanced

Select the cal method for each connector of the Thru pair.

- **TRL** is only available if a TRL cal kit was selected for the lowest port number of the port pair.

- **QSOLT** Only available when "Defined Thru" or "Flush Thru" is selected. "QSOLT 2 <= 1" refers to the receive port 2 and source port 1(where reflection standards are connected).

- **Enhanced Response** Only available when "Defined Thru" or "Flush Thru" is selected. "EnhResp 2 <= 1" refers to the receive port 2 and source port 1.

View Modify  Click to invoke the Preview and Modify Calibration Selections dialog box.

Note: Changes made to the Cal Kit through this dialog are temporary that last only for this calibration. To make permanent changes to the Cal Kit, perform Advanced Modify Cal Kits.
Select Cal Set for Delta Match dialog box help

This dialog box appears when a Delta Match Cal is required and Choose delta match was selected. Learn more.
Displays the Cal Sets that meet the requirements of the Delta Match Cal.
Select either a User Cal Set or Global Delta Match Cal.
If there is no suitable choice for a Delta Match Cal:

1. Click Cancel, then Cancel again to quit the Cal Wizard.
2. Perform either a Global Delta Match Cal or a SOLT cal and save the result in a User Cal Set.
3. Start the Cal Wizard to re-initiate this calibration.
4. Select the Global Delta Match Cal or User Cal Set.
Calibration Steps dialog box help

**Note:** Beginning in PNA Rev. 6.0, calibration can be performed with External triggers. [Learn more.]

As each new cal step prompt appears, the traces are setup for the next standard measurement. Also, sweeps are triggered continuously until the Measure button is pressed. This way you can view the integrity of the standard connection.

Prompts for standards to be measured.

- **Measure** Click to measure the standard.
- **Done** Click after a standard is re-measured and all measurements for the calibration are complete.
- **Next** Click to continue to the next calibration step. Does NOT measure the standard.

If a standard is NOT measured, a warning appears and Done will not be available after the last Cal step.

**Note:** SmartCal DOES allow you to measure calibration standards in any order. However, you must click Next and Back without measuring standards until you get to the standard you want to measure.

Sliding Load Measurement dialog box help

Allows you to measure the sliding load standard. [Learn more about the Sliding Load standard.]

To ensure an accurate calibration, carefully follow the instructions that were provided with your siding load.

**To Measure a Sliding Load:**

1. Connect the sliding load to the measurement port.
2. Position the sliding element, then click Measure. Do not move the sliding element until measurement is complete.
3. Measure the sliding load for at least five and up to seven positions for best accuracy.

   **Note:** The positions of the sliding element should cover the full length of the slide, but be unequally spaced to reduce the possibility of overlapping data points. Most sliding loads have marks for each slide position.
4. Click Done after the final measurement.
5. Remove sliding load from the measurement port.
6. Measure the remaining standards.

**Specify nominal delay** dialog box help

This dialog appears ONLY when Adapter Removal or Unknown Thru calibrations are performed.

The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

**Nominal adapter delay** To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here.

**Nominal phase offset** (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

For FCA calibrations, this dialog box appears twice: once for the input frequencies and once for the output frequencies. The values can be slightly different.

The Calibration Complete dialog box appears after all standards are measured.

**Unguided Calibration**

The PNA displays the following dialog boxes when performing an Unguided calibration:
Select Calibration Type for Mechanical Standards dialog box help

Unguided calibration does **NOT** support cals greater than 2 ports or **ECal** calibrations.

TRL Cal should be performed using **Guided Calibration**.

**Calibration Type Selection**

- 2-Port SOLT
- 1-Port SOL
- **TRL** - NOT available on PNA models with more than 2 ports.
- **Response** - Reflection and Thru (if the active measurement is transmission)

**Cal Configuration** If not calibrating all PNA ports, specify which ports to calibrate.

**Back** Return to **Cal Wizard Begin** dialog. If checked, you can clear the **Save Preferences** checkbox to see the Begin page when the Cal Wizard begins.

**View/Select Cal Kit** Click to invoke the **Select Cal Kit dialog box**.

**Next** Click to continue to **Measure Mechanical Standards** dialog box.

**Note:** If the DUT connector type has an impedance other than 50 ohms (waveguide = 1 ohm), change the system impedance before performing a calibration. See **Setting System Impedance**.
Select Cal Type dialog box help

This dialog box only appears if the selected Cal Type is TRL in the previous dialog box.

TRL Reference Plane Select which standard to use to establish the position of the measurement reference plane.

THRU Standard Select if the THRU standard is zero-length or very short.

REFLECT Standard Select if the THRU standard is not appropriate AND the delay of the REFLECT standard is well defined.

TRL Impedance

LINE Standard Specifies that the characteristic impedance of the LINE standard should be used as the system impedance. This ignores any difference between Offset Z0, Offset Loss, and System Z0.

SYSTEM Impedance Transforms the LINE standard impedance and loss to that of the system impedance for use with the calibration error terms. The TRL calibration will first compute the error terms assuming the LINE standard impedance is the system's characteristic impedance (same as previous LINE selection), then modify the error terms to include the impedance transformation. This should only be used with coax since the skin effect model used is a coaxial model.

Learn how to change System Z0.

To learn to substitute other calibration kits, see Advanced Modify Cal Kits

Select Cal Kit dialog box help

Displays the calibration kit files available for Unguided calibration. Select the desired calibration kit file and click OK.

Choose class type Unguided TRL calibration is NOT available on the 4-port PNA.

Edit Class Assignments Allows modification of the selected Cal Kit class assignments.

To learn to substitute other calibration kits, see Advanced Modify Cal Kits
Measure Mechanical Standards dialog box help

**Note:** Beginning in PNA Rev. 6.0, calibration can be performed with External triggers. [Learn more.](#)

Displays the calibration kit file and standards required for the calibration.

- Standards may be connected and measured in any order.
- Connect the standard to the measurement port and click its associated green button. A check mark indicates the standard has been measured.
- If a standard type contains multiple standards, the [Multiple Standards dialog box](#) opens to display the multiple standards included in the calibration kit file.
- If a sliding load is included in the calibration kit file, the [Sliding Load dialog box](#) opens to perform the measurement with the standard.
- **Reflection Response** Select EITHER Open or Short standard, then click **Next**.
- **Isolation** Requires one load for each test port of the PNA. [Learn more about Isolation.](#) Use when your measurement requires maximum dynamic range (> 90 dB). See also Isolation Portion of 2-Port Calibration.
- **Normalize** Available when performing a response cal for any measurement. After Normalize is pressed and the Cal is complete, the data trace is flat when the same physical connections are present on the port. This is similar to [Data/Memory](#), except that the response cal is saved with Cal data and can be applied to other like measurements. Data/Memory is still available after using Normalize. You would usually connect a THRU standard when calibrating a transmission measurement, and a SHORT standard when calibrating a reflection measurement.

**Show Prompts** Check to provide a reminder for the required connection when you click on the standard.
Multiple Standards dialog box help

Select the standards to be measured.

Note: You may see both male and female standards. The Unguided cal has no knowledge of the gender of your connector types. Choose the gender of your DUT connector, NOT the test port. Then click OK.

To modify this calibration class to show only one standard, on the Calibration menu, click Advanced Modify Cal Kits. Select the Cal kit and click Edit Kit. In Class Assignment, click Edit. Learn more about Modify Calibration Class Assignments.

- Connect the standard to the measurement port and click its associated button. A check mark in the Acquired box indicates the standard has been measured.

- To cover the entire frequency range, you may need to measure more than one standard. The order in which the standards are measured is important. The last standard that is measured will override the others in respect to the frequency range of the standard definition. Example: In the case of measuring both a broadband load and a sliding load, you would measure the sliding load last. This is because the frequency range of the sliding load is a subset of the broadband load.

Learn more about Modify Calibration Class Assignments

Saving a Calibration

SmartCal, ECal, and Unguided Calibrations end with the following dialog box:

Calibration Completed dialog box help

Finish  Save to the channel's calibration register.

Save As User Cal Set  Invokes the Save as User Cal Set dialog box AND save to the channel's calibration register.

Cancel  Calibration is NOT applied or saved.

Learn about Calibration Registers.

Learn about User Cal Sets
Save as User Cal Set dialog box help

**Existing Cal Sets**  - Lists the Cal Set names saved on the PNA.

**Select Cal Set from list or type new name below**  Specify a name for the new Cal Set. Either accept the suggested new name, type a new name, or select a name from the list to overwrite an existing name.

**Edit Name**  If there is no keyboard, click to start the PNA typing tool that can be used from the PNA front panel.

**Save**  Saves the Cal Set to the new Cal Set name.

Learn about [User Cal Sets](#)

---

Last modified:

- **21-Sep-2007**  Added note about no TRL on 4-port PNAs
- **January 20, 2007**  Added note about any order for SmartCal.
- **January 20, 2007**  MX Added UI
- **Sept 18, 2006**  MQ Major modifications for multiport
Select a Calibration Type

The following calibration types are available in the PNA.

<table>
<thead>
<tr>
<th>Cal Type</th>
<th>Interface</th>
<th>Accuracy</th>
<th>Thru Methods allowed</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRL Family</td>
<td>SmartCal</td>
<td>Very High</td>
<td>NOT unknown or adapter removal</td>
</tr>
<tr>
<td>SOLT</td>
<td>Both</td>
<td>High</td>
<td>All</td>
</tr>
<tr>
<td>Enhanced Response</td>
<td>SmartCal</td>
<td>High</td>
<td>Defined Thru or Flush Thru</td>
</tr>
<tr>
<td>QSOLT (Quick SOLT)</td>
<td>SmartCal</td>
<td>Medium</td>
<td>Defined Thru or Flush Thru</td>
</tr>
<tr>
<td>1-Port Reflection</td>
<td>Both</td>
<td>High</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Open/Short Response</td>
<td>Unguided</td>
<td>Low</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Thru Response</td>
<td>Unguided</td>
<td>Low</td>
<td>Known Thru or Flush Thru</td>
</tr>
</tbody>
</table>

Learn how to select a default Cal Type.

Other Cal Types (Separate Topic)

- Source and Receiver Power Cals
- FCA Scalar and Vector Mixer Cals

See other Calibration Topics

TRL Family

Application: Used to accurately calibrate any pair of ports when calibration standards are not readily available.

Note: A Delta Match Cal is required to cal test ports that do not have a dedicated reference receiver.

Learn more about TRL family cal
For more information on modifying standards, see Calibration Standards.

Calibration Method: SmartCal

General Accuracy: Very High

Standards Required: THRU, REFLECT, LINE or similar combination

Systematic Errors Corrected:
- Directivity
- Source match
- Isolation
<table>
<thead>
<tr>
<th>Load match</th>
<th>Frequency response transmission tracking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency response reflection tracking</td>
<td></td>
</tr>
</tbody>
</table>

**SOLT**

Application: Used to accurately calibrate any number of ports.

General Accuracy: High

Calibration Method: SmartCal, Unguided Calibration, ECal

Standards Required: (SHORT, OPEN, LOAD, THRU) or ECal module

Systematic Errors Corrected (on all ports):
- Directivity
- Source match
- Isolation
- Load match
- Frequency response transmission tracking
- Frequency response reflection tracking

**Enhanced Response**

Application: Used to calibrate two ports when only measurements in one direction (forward OR reverse) are required. Measurements are faster because a second sweep is NOT required.

- Reflection Standards (OPEN, SHORT, LOAD) are connected to the source port to be calibrated.
- Defined THRU or Flush THRU standard is connected between port pairs.

Much quicker than SOLT when using a mechanical cal kit. ECal can also be used.

**To select Enhanced Response:**

For a standard S-parameter Cal, select **SmartCal** in the Cal Wizard.

Then, for all cals:

1. At the 'Select DUT Connectors page', check **Modify Cal**, then click **Next**.
2. Under ‘Cal Type’, select **Enhanced Response**.

Enhanced Response cal also be selected as the default Cal Type using **Cal Preferences**.

General Accuracy: High

Calibration Method: SmartCal, ECal

Standards Required: (SHORT, OPEN, LOAD, Defined THRU or Flush THRU)

Systematic Errors Corrected:
- Directivity (source port)
- Source match (source port)
- Isolation
Load match (receiver port)
Frequency response transmission tracking (receiver port)
Frequency response reflection tracking (source port)

**QSOLT (Quick SOLT)**

Application: Used to quickly calibrate any number of ports. Developed specifically for use with [external multiport test sets](#).

**Note:** A [Delta Match Cal](#) is required to cal test ports that do not have a dedicated reference receiver.

- Reflection Standards (OPEN, SHORT, LOAD) are connected to only ONE of the ports to be calibrated. The lower port number of the ports to be calibrated is selected by default. This can be changed through the [Modify Cal / Cal Type](#) setting.
- [Defined THRU](#) or [Flush THRU](#) standards are connected from the reflection standard port to the remaining ports to be calibrated.
- Much quicker than SOLT when using a mechanical cal kit.
- Based on TRL math.

General Accuracy: Not as high as SOLT

Calibration Method: [SmartCal](#), [ECal](#)

Standards Required: (SHORT, OPEN, LOAD, [Defined THRU](#) or [Flush THRU](#))

Systematic Errors Corrected:
- Directivity
- Source match
- Isolation
- Load match
- Frequency response transmission tracking
- Frequency response reflection tracking

**1-Port (Reflection)**

Application: Used to accurately calibrate any single test port for reflection measurements only.

Calibration Method: [SmartCal](#), [Unguided Calibration](#), [ECal](#)

General Accuracy: High

Standards Required: (SHORT, OPEN, LOAD) or ECal module

Systematic Errors Corrected:
- Directivity
- Source match
- Frequency response reflection tracking
**Open / Short Response**

Application: Used to quickly calibrate any single test port for reflection measurements only.

Calibration Method: **Unguided Calibration**

General Accuracy: Low

Standards Required: OPEN or SHORT

Systematic Errors Corrected:
Frequency response reflection tracking

**Thru Response (Isolation Optional)**

Application: Used to quickly calibrate any pair of test ports for transmission measurements only. Isolation is not usually recommended. Learn more about Isolation

Calibration Method: **Unguided Calibration**

General Accuracy: Low

Standards Required: THRU

Isolation: One LOAD for each PNA test port.

Systematic Errors Corrected:
- Frequency response reflection tracking
- Isolation

Last modified:

- February 23, 2007  Added Enhanced Response
- 9/12/06  Added QSOLT
Using Calibration Sets

- What are PNA Cal Sets
- Cal Registers and User Cal Sets
- How to Manage and Apply Cal Sets
- Examples of Cal Set Usage
- Archiving Cal Sets using .cal files

See other Calibration Topics

What are PNA Cal Sets

At the completion of a calibration, all calibration data is stored to a Cal Set. The Cal Set can be applied later to any channel that has the same stimulus settings as the Cal Set, thereby saving the time it takes to perform another calibration. The following data is saved to a Cal Set:

- Name
- Cal Set Description
- Cal Set Attributes - stimulus settings, cal type, port association
- Standards data
- Error term data
- GUID (Globally Unique Identifier)

Cal Registers and User Cal Sets

There are two types of Cal Sets:

- **Cal Registers** (channel specific)
- **User Cal Sets**

Calibration data is automatically saved to a Cal Register at the end of every calibration. You can also choose to save the cal data to a User Cal Set.

Calibration Registers

New with PNA Release 5.0, Calibration Registers are designed to simplify calibrations for most users. When a calibration is complete, the data is automatically saved to the channel's Cal Register, overwriting (or appended to) the previous cal data stored in that register. This concept is similar to previous Agilent Vector Network Analyzers.
• Every channel has ONE dedicated Cal Register. They are named CHn_CALREG, where n is the channel number. The name cannot be changed.

• Cal Registers are more volatile because they are overwritten (or appended) each time a calibration is performed on that channel. The Cal data is always saved, but only temporarily.

• Cal Registers can be applied to other measurements, but ONLY on the same channel as the Cal Register.

User Cal Sets
At the end of a calibration, you can choose to also save cal data to an existing or new User Cal Set.

• User Cal Sets can be applied to any number of channels simultaneously.

• User Cal Sets are named by you for easy identification.

• You can have an unlimited number of User Cal Sets.

• At any time, you can copy Cal Register data to create a User Cal Set. See User Cal Sets Properties.

Appending Data in a Cal Set
At the end of a calibration, data is saved to the channel's Cal Register and, if you choose, to an existing User Cal Set. The PNA attempts to append new error terms to a non-empty Cal Set. The existing Cal Set data is completely overwritten UNLESS the new data can coexist with the existing data according to the following two rules:

• The stimulus settings of the new data must exactly match the existing data.

• The new cal must involve different ports from the existing cal.

For example:
Case 1 - An existing Cal Set contains a full 2-port cal between ports 1 and 2. Using the same stimulus settings, you perform a 1-port cal on port 3. At the end of the cal, you click Save As User Cal Set and select the existing full 2-port User Cal Set.

Result: The 1-port cal is appended to the 2-port User Cal Set. There is NO overlap between them.

Case 2 - Same situation as Case 1, except the 1-port cal is performed on port 1.

Result: The Cal Set will contain a 1 port cal on port1 and a 1 port cal on port 2. The overlapping tracking terms are removed rendering the original full 2 port cal invalid.

How to Manage and Apply Cal Sets and Cal Types
The PNA attempts to apply a Cal Set, and turn error correction ON, for ALL of the measurements on the active channel. This may not always be possible. For example, suppose a channel contains both S11 (reflection) and S21 (transmission) measurements. If a Cal Set that contains only an S11 Cal Type is applied to that channel, the Cal Set does not contain the error terms to correct the S21 measurement. Error correction is turned ON for the S11 measurement and NOT turned on for the S21 measurement.

There are two ways to apply an existing Cal Set (Cal Register or User Cal Set) to a measurement:
1. Recalling an Instrument State with Cal data (.cst file) - A .cst file contains an Instrument State with all measurement attributes AND a ‘pointer’ to the Cal Set that was used to calibrate the measurement. Before saving a .cst file, be sure that a User Cal Set (NOT a Cal Register) is being used for the measurement. Because Cal Registers are automatically overwritten when a new calibration is performed, it is likely that the Cal Register data will change before the .cst file is recalled.

2. Create a new measurement and select a Cal Set to apply to the active channel.

**Note:** NEVER copy or modify Cal Sets from Windows Explorer or other applications. Cal Sets should only be accessed through the PNA Application.

### How to select and apply a Cal Set to the active channel

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
</table>

#### For N5230A and E836xA/B models

1. **Navigate using** MENU/DIALOG

2. **then** Cal Set

#### For PNA-X and ’C’ models

1. **Press** CAL

2. **then** Manage Cals

3. **then** Cal Set

### Calibration Selection dialog box help

This dialog box allows you to manage and apply Cal Sets.

Although the number of Cal Sets you can have is limited only by the amount of PNA memory, it is considered unusual to have more than about 10 existing Cal Sets, or one current Cal Set for every unique channel setup.
Old Cal Sets (with 'stale' data) should be deleted or overwritten. The active channel's Cal Register always appears, even if empty. Cal Registers that belong to other channels appear in the list of Cal Sets only if the channel exists, and only if they contain data.

- Learn about **Cal Registers**.
- Learn how to **View the Error Terms of a Cal Set**.

**To apply a Cal Set** to the active channel, click a row to select that Cal Set, then click Apply Cal.

**Note:** A Cal Set must have been generated from the same **measurement class** as the active channel in order for it to be Applied.

<table>
<thead>
<tr>
<th><strong>Columns</strong></th>
<th>click a heading to sort by that column</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cal Set Name</strong></td>
<td>Name to identify the Cal Set.</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>User-settable text to further identify the Cal Set.</td>
</tr>
<tr>
<td><strong>Channels</strong></td>
<td>Channel numbers that are currently using this Cal Set. A blank entry means it is not currently in use.</td>
</tr>
<tr>
<td><strong>CalType / Ports</strong></td>
<td>Type of Cal contained in the Cal Set. <strong>Learn about applying appropriate Cal Types.</strong></td>
</tr>
<tr>
<td><strong>Cal Type Abbreviations:</strong></td>
<td></td>
</tr>
<tr>
<td>1P, 2P, 3P, 4P...</td>
<td>Full n-Port calibrations</td>
</tr>
<tr>
<td>R</td>
<td>Response (instead of ports, shows the measurement type that it corrects.)</td>
</tr>
<tr>
<td>ER/x-y</td>
<td>Enhanced Response, where x is the receive port; y is the source port.</td>
</tr>
<tr>
<td>VMC</td>
<td>Vector Mixer Cal</td>
</tr>
<tr>
<td>SMC</td>
<td>Scalar Mixer Cal</td>
</tr>
<tr>
<td><strong>Modified</strong></td>
<td>Date and time the Cal Set was last modified.</td>
</tr>
</tbody>
</table>

**Buttons**

- **Copy** Invokes the **Save as User Cal Set** dialog box. Type a name for the copy of the selected Cal Set data.
- **Show / Edit Properties** Invokes the Cal Set Properties dialog box. This allows you to view all of the Cal Set properties and create a **duplicate** User Cal Set from an existing User Cal Set or Cal Register.
- **Delete** Permanently deletes the Cal Set after you choose OK to a warning prompt.
- **Delete All** Permanently deletes ALL listed Cal Sets and Cal Registers after you choose OK to a warning prompt.
- **Apply Cal** Applies the selected Cal Set to the active channel. If the stimulus settings of the Cal Set and channel are different, **a choice must be made.**
- **Unselect** Available ONLY if the selected Cal Set is being used by the active channel. Click to 'Un-apply' the Cal Set, then click **Close** to exit with the Cal Set un-applied.
- **OK** Always APPLIES THE SELECTED CAL SET to the active channel, then closes the dialog box.
- **Close** Exit the dialog box. Performs no further action.
Cal Set Properties dialog box help

Allows you to view all of the Cal Set properties and create a **duplicate** User Cal Set from an existing User Cal Set or Cal Register.

- **Name**  Edit name of the User Cal Set. You can NOT change the name of a Cal Register.
- **Description**  Descriptive text to further identify the Cal Set.
- **Cal Set Properties**  Lists descriptive information and stimulus conditions of the Cal Set.

Learn how to [View the Error Terms of a Cal Set](#).

---

**Stimulus Setting Different between Cal Set and Measurement**

The Cal Set that you have selected has different stimulus settings than the active channel.

Please select one of the following actions:

- **A**  Do not change the active channel's stimulus settings. If necessary, interpolation will be turned on automatically.
- **B**  Change the active channel's stimulus settings to match those of the selected Cal Set.

[Select Cal Set -- Choose Stimulus Settings](#)
The Cal Set contains the channel stimulus settings that were in place when the Cal Set was saved. This dialog appears when the Cal Set channel settings are different than those of the channel to which the Cal Set is being applied. Choose between the following options. (See above image).

- A. Keep the Active Channel Stimulus settings. Interpolate if possible.
  - If the Cal Set frequency range is greater the active channel, then Interpolation will be turned ON. Learn more about Interpolation Accuracy.
  - If the Cal Set frequency range is less than the active channel, then this option is not available.
- B. Keep the Cal Set Stimulus settings. The Active Channel stimulus setting are changed.

OK  Make the change.
Cancel  Cal Set will NOT be applied.

**Examples of Cal Set Usage**

The following examples show how Cal Sets increase flexibility and speed in making analyzer measurements.

- **Using one User Cal Set with many Channels**
- **Using one Measurement with many Cal Sets**

**Using one User Cal Set with many Channels**

It is possible to do one calibration, then apply it to several channels.

**An example:**

During a manufacturing process, you may have many calibrated channels. You may wish to continuously cycle through the measurements and examine them individually. Occasionally, you may wish to refresh the calibration without having to recreate all the measurement state files.

**Here is how:** Examine the stimulus settings for each channel. Then make the User Cal Set stimulus range a super-set of the whole group. Each channel can then use the same User Cal Set. Some calibrations will be interpolated. **Note:** Make sure that interpolation is turned on.

Notice in the following image, Cal Set 78 is used on more than one channel, in this case Channel 5 and 16.

<table>
<thead>
<tr>
<th>Channel</th>
<th>Cal Set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Using one Measurement with many Cal Sets

The drawback with having one very large User Cal Set associated with many instrument states could be a loss of accuracy due to interpolation. In such cases, consider using one User Cal Set for each stimulus setting. The stimulus conditions can then be changed for a channel by applying different User Cal Sets. Other settings (window setups, measurement definitions, scaling, limits, markers) will not change. This may result in faster state changes than if you saved and recalled *.cst files for each set of stimulus conditions.

Example #1: An amplifier needs to be measured at several input power levels. Calibrate at several power levels and save each calibration in a separate User Cal Set. Then, apply the User Cal Sets to the single measurement consecutively.

Example #2: Making an S21 Measurement, you need to measure both wide span and narrow span characteristics of the device. One Cal Set covers the wide span setup; another the narrow span setup.

Archiving Cal Sets using .cal or .csa files

Because User Cal Sets can easily be deleted, provide extra backup by also saving your calibration as a .cal or .csa file (see saving a .cal file).

Example:

One person performs a calibration, names and saves it as a User Cal Set. This Cal Set is available for any other person to use. A second user could accidentally delete or modify the User Cal Set requiring the originator to repeat the calibration.

Security can be provided for calibration data by saving the Cal Set to a .cal file or .csa file. At a later time, the file could be recalled and the original calibration restored.

Last modified:

9/12/06    Added link to programming commands
Error Correction and Interpolation

Error Correction and Interpolation settings work together to provide you with the highest level of calibration accuracy possible.

- **How to set Error Correction**
- **Error Correction**
- **Viewing Correction Levels**
- **How to set Interpolation**
- **Interpolation and Accuracy**

See other Calibration Topics

### How to set Error Correction

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click <strong>Calibration</strong></td>
</tr>
<tr>
<td>2. then <strong>Correction ON/off</strong></td>
<td>2. then <strong>Correction ON/off</strong></td>
</tr>
<tr>
<td>For <strong>PNA-X</strong> and <strong>’C’</strong> models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press <strong>CAL</strong></td>
<td>1. Click <strong>Cal</strong></td>
</tr>
<tr>
<td>2. then <strong>Correction ON/off</strong></td>
<td>2. then <strong>Correction ON/off</strong></td>
</tr>
</tbody>
</table>

**Error Correction**

The Error Correction ON setting means that the calibration error terms are applied to the measurement. Error Correction is automatically turned ON when a calibration is performed or if a Cal Set is applied to a measurement. The PNA attempts to turn error correction ON for ALL of the measurements on the active channel. This may not always be possible when applying Cal Sets. For more information, see [Applying Cal Sets](#).

When full 2-port error correction is ON, both forward and reverse sweeps are required to gather all 12 error terms, even if only one reflection measurement is displayed. This may result in a higher measurement speed than expected. [Learn more](#).

You can always turn Error Correction OFF for the active measurement by clicking Correction OFF. The PNA will
turn Error Correction OFF automatically when making stimulus changes under some conditions. To turn correction back ON, click Correction ON. Then:

- If Interpolation can NOT be performed, a dialog box will ask if you would like to change the stimulus settings to those of the applied calibration. Click OK or Cancel.
- If Interpolation can be performed, the stimulus setting will change and correction turned ON.

Viewing Correction Level

The correction level provides information about the accuracy of the active measurement. Correction level notation is displayed on the status bar for different calibration types like response, full 2-port, TRL, or power calibration.

To View Correction Levels:

In the View menu, click Status Bar. The status bar appears and displays the following items:

- Active Channel
- Measurement parameter
- Correction Level (see description below)
- Calibration type

<table>
<thead>
<tr>
<th>Correction Level</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>C Full</td>
<td>Highest</td>
</tr>
<tr>
<td>C* Interpolated</td>
<td>Uncertain</td>
</tr>
<tr>
<td>C Changed</td>
<td>Uncertain</td>
</tr>
<tr>
<td>No Cor</td>
<td>Lowest</td>
</tr>
</tbody>
</table>

C Full Correction

Full Correction level is displayed immediately after a calibration is performed or when a valid Cal Set is applied. If you require optimum accuracy, avoid adjusting analyzer settings after calibration so your measurement remains at this level.

C* Interpolated Correction

"C star" appears in the status bar when a measurement is being interpolated. See Interpolation (above) and Interpolation Accuracy.

C Changed Settings
“C-delta” appears in the status bar when one or more of the following stimulus settings change. The resulting measurement accuracy depends on which parameter has changed and how much it has changed. For optimum accuracy, recalibrate using the new settings.

- **Sweep time**
- **IF Bandwidth**
- **Port power**
- **Stepped sweep enabled/disabled**

**No Corr  No Correction**

The following will cause the PNA to turn Error Correction OFF for the channel:

- Decrease the start frequency
- Increase the stop frequency
- Change start frequency, stop frequency, or number of points with Interpolation OFF.
- **Change sweep type**

### How to set Interpolation

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Calibration</td>
</tr>
<tr>
<td></td>
<td>2. then Interpolation ON/off</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press CAL</td>
<td>1. Click Cal</td>
</tr>
<tr>
<td>2. then [More]</td>
<td>2. then More</td>
</tr>
<tr>
<td>3. then [Interpolation ON/off]</td>
<td>3. then Interpolation ON/off</td>
</tr>
</tbody>
</table>

**Interpolation**

Calibration interpolation adjusts calibration error terms to match changes to the following settings that you make AFTER a calibration is performed or a **Cal Set applied**.
The Interpolation **ON** setting means that interpolation is **enabled** for the active measurement. This does not necessarily mean that the measurement is interpolated. When enabled (ON), if interpolation becomes necessary because you change any of the following stimulus settings, **then** interpolation will be applied. When stimulus settings change while interpolation is OFF, interpolation is NOT applied but instead, error correction is turned OFF.

Interpolation occurs (if enabled) when you change any of the following settings:

- Start frequency increased
- Stop frequency decreased
- Number of points

**Note:** Decreasing the start frequency, or increasing the stop frequency will always turn correction OFF. (Exception: **Power Calibration** DOES extrapolate to the start and stop frequencies.)

### Interpolation Accuracy

When a measurement is interpolated, the accuracy of the measurements cannot be predicted. It may be affected significantly or not at all. Identifying measurement errors in these cases must be determined on a case-by-case basis.

Significant measurement inaccuracy WILL occur when the phase shift between measurement points increases more than 180 degrees. The PNA will incorrectly interpolate the new phase data. For more information, see **phase accuracy**.

In general, the chances of significant inaccuracy increases when interpolating measurements under the following conditions:

- when increasing, rather than decreasing, the frequency span between measurement points.
- when frequency span between measurement points becomes much greater.
- when measurement frequencies are very high, especially above 10 GHz.

---

**Last modified:**

- March 10, 2008  MX Added UI
- Sept 12, 20/06  Added link to programming commands
Calibration Thru Methods

What is a Non-Insertable Device

Choosing a Thru Method

Flush Thru

Adapter Removal

Defined Thru

Unknown Thru

ECal Thru Method Choices

Other Cal Topics

What is a Non-Insertable Device

To understand the Thru method choices, you must first understand what is meant by "Non-Insertable device". These definitions also apply to ECal modules. Substitute "ECal module" for "device". Then see ECal Thru Method Choices.

A non-insertable device is one whose connectors could NOT mate together. They either do not have the same type of connector or they have the same gender. This also means that the test port cables would not mate together, as in the following diagram.

An insertable device is one whose connectors could mate together. They have the same type of connector and opposite, or no, gender. This also means that the test port cables would mate together, as in the following diagram.
Choosing a Thru Method of Calibration

The Thru method is selected from the Cal Wizard. Select the **Modify** checkbox in the **Select DUT Connectors and Cal Kits** dialog box.

**Notes:**
For ECal, the following choices have different meanings. See [THRU methods for ECal](#).
For 4-port calibration, see [How can we measure only 3 THRU connections?](#)

**Choice for Insertable Devices: FLUSH Thru** (also known as **Zero-length Thru**)

When calibrating for an insertable device, the test ports at your measurement reference plane connect directly together. This is called a zero-length THRU, or Flush THRU meaning that the THRU standard has zero-length: no delay, no loss, no capacitance, and no inductance. Your calibration kit may not have a physical THRU standard because it is assumed you have an insertable device and will be using a zero-length THRU.

**Choices for Non-Insertable Devices**

The following methods calibrate for a non-insertable device:

- Adapter Removal  
  Accurate, but least convenient.

- Defined Thru

- **Unknown Thru Cal**  
  Preferred method.

- Swap-Equal-Adapters Method is a valid choice, but NOT included in the PNA firmware.

**Adapter Removal Calibration**

This method is potentially very accurate. However, it requires many connections which increases the chances of inaccurate data.

Two full 2-port calibrations are performed: one with the adapter connected at port 1, and the other with the adapter connected to port 2. The result of the two calibrations is a single full 2-port calibration that includes accurate characterization and removal of the mismatch caused by the adapter.

Performing an Adapter Removal Cal requires:

- a THRU adapter with connectors that match those on the DUT.
• calibration standards for both DUT connectors.

To select Adapter Removal during a SmartCal, select the Modify checkbox in the Select DUT Connectors and Cal Kits dialog box. The Cal Wizard will guide you through the steps.

Learn how to perform an Adapter Removal Cal using ECal.

**Defined Thru** (also known as Known Thru, Cal Kit Thru, ECal Thru, Characterized Thru)

Defined Thru uses the THRU definition that is stored in the Cal Kit file or ECal module. The THRU standard may have worn over time, making it not as accurate as when it was new. Defined Thru is usually more accurate than Adapter Removal, but not as accurate as **Unknown Thru** method.

**Notes**

• If performing an ECal, this is the THRU standard in the ECal Module.

• If Defined Thru appears as a potential THRU method in the SmartCal Wizard, this means that there is a defined THRU standard in the selected Cal Kit. This could be a Zero-length Thru. The SmartCal Wizard will prompt you to connect the required standard when appropriate.

To define a THRU standard in a Cal Kit (not ECal module):

1. From the PNA Menu, click Calibration, Advanced Modify Cal Kits.
2. Select the Cal Kit
3. Click Edit Kit
4. Click Add
5. Select THRU
6. Complete the dialog box.

The next time you perform a Guided Cal, this Defined THRU standard will be available if the DUT connector types match the THRU standard.

**Unknown Thru Cal**

Unknown Thru Cal is the preferred THRU method of calibrating the PNA to measure a non-insertable device.

The Unknown Thru calibration is also known as Short-Open-Load-Reciprocal Thru (SOLR) calibration.

• Very easy to perform.

• Better accuracy than Defined Thru and usually better than Adapter Removal.

• Does not rely on existing standard definitions that may no longer be accurate.

• Causes minimal cable movement if the THRU standard has the same footprint as the DUT. In fact, the DUT can often BE the THRU standard.
About the Unknown Thru Process

SmartCal guides you through the process. Although the following process describes ports 1 and 2, Unknown Thru can be performed on any two ports when using a multiport PNA.

1. Perform 1-port cal on port 1.
2. Perform 1-port cal on port 2.
3. Connect Unknown Thru between ports 1 and 2.
4. Measure Unknown Thru.
5. **Confirm Estimated Delay.** This estimate may be wrong if there are too few frequency points over the given frequency span. You can measure the delay value independently and enter that value in the dialog box.

The Unknown Thru Standard

- Can have up to about 40 dB of loss and long electrical length.
- Must be reciprocal: S21=S12.
- Must know the phase response to within 1/4 wavelength (see step 5 above).
- Can be the DUT if it meets these conditions.

Unknown Thru Limitations

- Unknown Thru is NOT supported during a TRL calibration.
- Beginning with PNA code release 5.25, Unknown Thru CAN be performed using a 4-port PNA-L that does NOT have a reference receiver for each test port. However, a Delta Match Calibration is usually required before the Unknown Thru is measured.
- Unknown Thru is NOT supported on E8801A, E8802A, and E8803A.

ECal Thru Method Choices

When the ECal module connectors exactly match the DUT connectors, choose from the following THRU methods:

**ECal Thru as Unknown Thru**  Learn more about Unknown Thru.

- Measures the THRU state of the ECal module as an Unknown Thru.
- The default method when the ECal module connectors match the DUT.
- Very accurate and easy.
- May require a Delta Match Cal.
Flush Thru (zero-length Thru)  Learn more about Flush Thru

- Requires an insertable ECal module / DUT.
- Remove the ECal module and connect the two reference planes directly together for a zero-length thru.
- Accurate, but not as easy as 'ECal Thru as Unknown Thru'.

ECal (Defined Thru)

- Measures the THRU state of the ECal module.
- Very easy, but not very accurate.

Unknown Thru

- Remove the ECal module.
- Then connect a Thru adapter to be measured as Unknown Thru.
- May require a Delta Match Cal.

When the ECal module connectors do NOT exactly match the DUT connectors, choose from the following two methods:

Adapter Removal

- Can be used with ECal when your DUT is NON-insertable. However, the ECal module MUST be insertable, and the adapter connectors must exactly match the connectors of the DUT as in the following diagram.

**Note:** With PNA release 4.8, adapter removal now performs 2-port measurements on both sides of the adapter. It previously performed 2-port measurements on one side and 1-port measurements on the other. This improves the accuracy of the adapter removal calibration.
ECal User Characterization

In cases when adapter removal cannot be performed, ECal User Characterization is ALWAYS possible if you have the right adapters. A User Characterization is performed once and stored in the ECal module. However, accuracy is compromised every time you remove, then reconnect, the adapter with the ECal module.

Last Modified:

20-Feb-2008    Added bullet for default ECal method
Accurate Measurement Calibrations

Calibration accuracy is affected by the type of calibration, quality of the calibration standards, and the care with which the calibration is performed. This section provides additional information about how to make accurate calibrations.

- **Measurement Reference Plane**
- **Effects of Using Wrong Calibration Standards**
- **Data-based versus Polynomial Calibration Kits**
- **Accuracy Level of Interpolated Measurement**
- **Effects of Power Level**
- **Using Port Extensions**
- **Isolation Portion of 2-Port Calibration**
- **Choosing a Thru Method**

Learn how to [determine the validity of your calibration](#).

### See other Calibration Topics

**Measurement Reference Plane**

Most measurement setups will NOT allow you to connect a device under test (DUT) directly to the PNA front panel test ports. More likely, you would connect your device to test fixtures, adapters, or cables that are connected to the PNA.

A calibration takes place at the points where calibration standards are connected during the calibration process. This is called the measurement reference plane (see graphic). For the highest measurement accuracy, make the calibration reference plane the place where your DUT is connected. When this occurs, the errors associated with the test setup (cables, test fixtures, and adapters used between the analyzer ports and the reference plane) are measured and removed in the calibration process.
Effects of Using Wrong Calibration Standards

Normally, a calibration is performed using a calibration kit that contains standards with connectors of the same type and sex as your device under test.

However, your calibration kit may not always have the same connector type and gender as your device. For example, suppose your device has 3.5mm connectors, but you have a Type-N calibration kit. If you use an adapter to connect the Type-N standards to the 3.5mm test port, then the adapter becomes part of the calibration and NOT part of the test setup. This will result in significant errors in your reflection measurements.

Data-based versus Polynomial Calibration Kits

The Select DUT Connectors and Cal Kits dialog box offers a data-based model and a polynomial model for the newest high-frequency cal kits. See PNA Accessories. The data-based models provide higher accuracy for describing calibration standards than the polynomial models. It is RECOMMENDED that the data-based model be used if the most accurate results are desired.

<table>
<thead>
<tr>
<th>How accurate is the model?</th>
<th>Data-Based Model</th>
<th>Polynomial Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provides highest calibration accuracy. Eliminates the errors that can be the result of polynomial model approximations.</td>
<td>Provides high calibration accuracy.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How does the model define calibration standards?</th>
<th>Data-Based Model</th>
<th>Polynomial Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uses S-Parameter measurements.</td>
<td>Uses traditional four-term polynomial calibration standard modeling parameters.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How do I manually edit the definitions of the calibration standards when using the model?</th>
<th>Data-Based Model</th>
<th>Polynomial Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the Advanced Modify Cal Kit function.</td>
<td>Use the Advanced Modify Cal Kit function.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>How do I use the Calibration Wizard with the model?</th>
<th>Data-Based Model</th>
<th>Polynomial Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use only the SmartCal (Guided) Calibration method.</td>
<td>Use the SmartCal (Guided) or the Unguided Mechanical Calibration methods.</td>
<td></td>
</tr>
</tbody>
</table>

Learn about the “Expanded Math” feature.

Effects of Power Level

To attain the most accurate error correction, do NOT change the power level after a calibration is performed. However, when changing power within the same attenuator range at which the measurement calibration was performed, S-parameter measurements can be made with only a small degradation of accuracy. If a different attenuator range is selected, the accuracy of error correction is further degraded.

To check the accuracy of a calibration, see Validity of a Calibration.

Using Port Extensions

Use the port extensions feature after calibration to compensate for phase shift of an extended measurement.
reference plane due to additions such as cables, adapters, or fixtures.

Port extensions is the simplest method to compensate for phase shift, mismatch, and loss of the path between the calibration reference plane and the DUT.

Learn how to apply port extensions.

Learn about characterizing a test fixture.

Isolation Portion of 2-Port Calibration

The isolation portion of a calibration corrects for crosstalk, the signal leakage between test ports when no device is present. When performing an UNGUIDED 2-port calibration, you have the option of omitting the isolation portion of the calibration.

**Note:** Isolation is never performed on a Smart (Guided) Calibration.

The uncorrected isolation between the test ports of the PNA is exceptional (typically >100dB). Therefore, you should only perform the Isolation portion of a 2-port calibration when you require isolation that is better than 100dB. Perform an isolation calibration when you are testing a device with high insertion loss, such as some filter stopbands or a switch in the open position.

The isolation calibration can add noise to the error model when the measurement is very close to the noise floor of the analyzer. To improve measurement accuracy, set a narrow IF Bandwidth.

How to perform an isolation calibration

Isolation is measured when the Load standards are connected to the PNA test ports. For best accuracy, connect Load standards to BOTH test ports each time you are prompted to connect a load standard. If two Loads are not available, connect the untested PNA port to any device that will present a good match.

Choosing a Thru Method

When calibrating for a non-insertable device, you must choose a method to calibrate for the THRU error terms. This can have a significant effect on measurement accuracy. Learn more about choosing a thru method.
Validity of a Calibration

This section helps you determine if your calibration is valid and how the analyzer displays correction level information for your measurement.

- **Frequency Response of Calibration Standards**
- **Validating a Calibration**
- **Quick Check**
- **ECal Confidence Check**
- **Verification Kit**

### See other Calibration Topics

---

**Frequency Response of Calibration Standards**

In order for the response of a calibration standard to show as a dot on the smith chart display format, it must have no phase delay with respect to frequency. The only standards that exhibit such "perfect" response are the following:

- 7-mm short (with no offset)
- Type-N male short (with no offset)

There are two reasons why other types of calibration standards show phase delay after calibration:

1. The reference plane of the standard is electrically offset from the mating plane of the test port. Such devices exhibit the properties of a small length of transmission line, including a certain amount of phase shift.

2. The standard is an open termination, which by definition exhibits a certain amount of fringe capacitance and therefore phase shift. Open terminations which are offset from the mating plane will exhibit a phase shift due to the offset in addition to the phase shift caused by the fringe capacitance.

The most important point to remember is that all standards are measured in order to remove systematic errors from subsequent device measurements. As a result, if calibration standards with delay and fringe capacitance are measured as a device after a calibration, they will NOT appear to be "perfect“. This is an indication that your analyzer is calibrated accurately and working properly.

---

**Validating a Calibration**

At the completion of a calibration or selection of a stored Cal Set, validation can accomplish the following:
**Improve Measurement Accuracy** – Once a measurement calibration has been performed, its performance should be checked before making device measurements. There are several sources of error that can invalidate a calibration: bad cables, dirty or worn calibration standards that no longer behave like the modeled standards, and operator error.

**Verify Accuracy of Interpolation** – You should validate the calibration if you are testing a device and the measurements are uncertain because of interpolation. For more information see Interpolation Accuracy.

**Verify Accuracy of Cal Standards** – To check accuracy, a device with a known magnitude and phase response should be measured.

**Quick Check**

For this test, all you need are a few calibration standards. The device used should not be one of the calibration standards; a measurement of one of these standards is merely a measure of repeatability.

The following reflection and transmission Quick Check tests can be applied to all test ports.

**To verify reflection measurements, perform the following steps:**

1. Connect either an OPEN or SHORT standard to port 1. The magnitude of S11 should be close to 0 dB (within a few tenths of a dB).

2. Connect a load calibration standard to port 1. The magnitude of S11 should be less than the specified calibrated directivity of the analyzer (typically less than -30 dB).

**To verify transmission measurements:**

1. Connect a THRU cable (or known device representative of your measurement) from port 1 to port 2. Verify the loss characteristics are equivalent to the known performance of the cable or device.

2. To verify S21 isolation, connect two loads: one on port 1 and one on port 2. Measure the magnitude of S21 and verify that it is less than the specified isolation (typically less than -80 dB).

**Note:** To get a more accurate range of expected values for these measurements, consult the analyzer's specifications.

**ECal Confidence Check**

ECal Confidence Check is a method to check the accuracy of a calibration performed with mechanical standards or an ECal module. The confidence check allows you to measure an impedance state in the ECal module (called the confidence state), and compare it with factory measured data stored in the module.

In order for this test to be valid, the test ports of the ECal module must connect directly to the calibration reference plane (without adapters).

**How to Perform ECal Confidence Check:**

1. Connect ECal module to the analyzer with the USB cable. See Connect ECal Module to the PNA. **Note:** Terminate any unused ECAL ports with a 50 ohm load.
2. Allow the module to warm up for 15 minutes or until the module indicates READY.

3. Do one of the following to start ECal Confidence Check

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Calibration</td>
</tr>
<tr>
<td></td>
<td>2. then ECal Confidence Check</td>
</tr>
</tbody>
</table>

For **PNA-X** and **'C'** models

<table>
<thead>
<tr>
<th><strong>Programming Commands</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Click Calibration</td>
</tr>
<tr>
<td>2. then ECal Confidence Check</td>
</tr>
<tr>
<td>3. then More</td>
</tr>
<tr>
<td>4. then ECal</td>
</tr>
<tr>
<td>5. then Confidence Check</td>
</tr>
</tbody>
</table>

On the following **ECal Confidence Check dialog box**:

4. Click **Read Module Data**. The following occurs:

- ECal module is set to "confidence state".
- PNA reads and displays stored data.
- PNA measures and displays "confidence state".

5. If you want to view a different parameter, select **Change Measurement** and select the check box for the desired parameter. (The default is the active channel parameter).

6. Select the viewing option in the Trace View Options block.

7. Compare the stored and measured data for each measurement parameter.

**Notes:**

- If the two traces show excessive difference, there may be a loose or dirty connection at the test ports or damage to the test cables. Carefully inspect the cables and connections. Then clean and gage each connector, and re-calibrate if needed.
- The User Characterization setting selects the user-characterization data instead of the factory data.
characterization data (available when a User-Characterization is stored in the ECal module).

**ECal Confidence Check**

Begin by Selecting Measurement. Then Use ECal Module.

**Measurement**

- **Change Measurement**  Opens the Measure dialog box.

**Use ECal Module**

- **Read Module Data**
- **User Characterization**

**Trace View Options**

- **Data and Memory**
- **Data / Memory**
- **Data - Memory**

**Scale**

- **Show Prompts**  Check to show a reminder for the connection (default).

**Verification Kit**

- **Data / Memory Trace**  Displays current measurement data and Memory trace.
- **Data + Memory**  Performs an operation where the current measurement data is added to the data in memory.
- **Data / Memory**  Performs an operation where the current measurement data is divided by the data in memory.

**Learn more about ECal Confidence Check.**
Measuring known devices, other than calibration standards, is a straightforward way of verifying that the network analyzer system is operating properly. Verification kits use accurately known verification standards with well-defined magnitude and phase response. These kits include precision airlines, mismatch airlines, and precision fixed attenuators. Traceable measurement data is shipped with each kit on disk and verification kits may be re-certified by Agilent.

See Analyzer Accessories for a list of Agilent verification kits.

Last modified:

March 10, 2008  MX Added UI

9/12/06  Added link to programming commands
Using ECal

This topic discusses all aspects of performing an ECal:

- **ECal Overview**
- **Connect ECal Module to the PNA**
- **How to Perform a Calibration Using ECal**

See Also:
- ECal User-Characterization
- Restore ECal Module Memory

---

**See other Calibration Topics**

---

**ECal Overview**

ECal is a complete solid-state calibration solution. Every ECal module contains electronic standards that are automatically switched into position during a PNA measurement calibration. These electronic standards have been measured at the factory and the data stored within the memory of the ECal module. The PNA uses this stored data, along with the PNA-measured data, to calculate the error terms for a measurement calibration.

ECal modules are available in 2-port and 4-port models and a variety of connector types, covering many frequency ranges. See Analyzer Accessories for more about available ECal modules and ordering information.

You can perform the following calibrations with ECal:

- 1-Port Reflection calibration
- Full 2-Port calibration
- Full 3-Port calibration
- Full 4-Port calibration

Verify the validity of a mechanical or ECal calibration with ECal confidence check.

**Care and Handling of ECal Modules**

You can improve accuracy, repeatability, and avoid costly repair of equipment in the following ways.

- Practice proper connector care. See Connector Care.
- Protect equipment against ESD damage. Read Electrostatic Discharge Protection.
- Do not apply excess power to ports. Refer to specifications provided with your ECal module.
Connect ECal Module to the PNA

ECal modules are controlled and powered through a USB connection to the PNA. When you connect the module, the PNA automatically recognizes the type of module, frequency range, and connector type.

"First Time" Note

When you connect an ECal module that has a serial number never before seen by that PNA, the **Welcome to the Found New Hardware Wizard** will appear.

Click "**No, not this time**", then click **Next** or **Finish** at the remaining dialog prompts.

You must be logged on to the PNA with **Administrator privileges** to complete the first time registration process. This occurs automatically unless you change your default User Account.

ECal modules connect to the USB port on the front or rear panel of the PNA.

1. Wear a grounded wrist strap when making connections.

2. Connect the USB cable **Type B** connector to the ECal module and the USB cable **Type A** connector to the front or rear panel USB connector of the analyzer, as shown in the following graphics.

![USB Type B and Type A connectors](image)

Notes:

- Unused ECal modules that have completed a calibration may remain connected to the USB port.

- You can connect and disconnect the ECal module while the analyzer is operating. However, **DO NOT** connect or disconnect the module while data transfer is in progress. This can result in damage or at least corrupted data.

- A USB hub may be used to connect more than one USB device to the analyzer. See **Analyzer Accessories** for more information about USB peripheral equipment.
# How to Perform a Calibration Using ECal

Select an ECal module that has connectors of the same type and gender as the DUT. If such an ECal module is not available, a module with connectors different from the DUT can be used by using Advanced Settings or User Characterization.

Connect the ECal module ports to the PNA ports. During the calibration process the PNA can either automatically detect how the ECal module is connected, or the orientation can be performed manually.

**Note:** Terminate any unused ECal ports with a 50 ohm load.

1. Connect the ECal module USB cable to the analyzer USB. See Connect ECal Module USB to PNA USB.
2. Allow the module to warm up until it indicates READY.
3. Enter the analyzer settings. See Set Up Measurements.
4. Do one of the following to start the Calibration Wizard:
   - **Using front-panel HARDKEY [softkey] buttons**
     - **For N5230A and E836xA/B models**
       1. Navigate using MENU/DIALOG
          2. Click Calibration
          3. then Cal Wizard
     - **For PNA-X and 'C' models**
       1. Press CAL
          2. then [Start Cal]
          3. then [Cal Wizard]

5. In the Calibration Wizard Begin dialog box, click Use Electronic Cal (ECal).

**Note:** To calibrate with more than one ECal module, select SmartCal, then choose the ECal modules as your Cal Kits.
Select Calibration Ports and ECal Module dialog box help

Allows you to select calibration type and settings.

**Cal Type Selection / Configuration** Select the number of ports to calibrate. Then select the port number configuration.

- **4 Port ECal** Available only if using a 4-port PNA. No additional configuration necessary.
- **3 Port ECal** Available only if using a 4-port or 3-port PNA.
- **2 Port ECal**
- **1 Port ECal** (Reflection) Advanced Settings are not available.

**View/Select ECal Module** Click to Select the ECal module if more than one ECal module is connected to the PNA. Also, Select the User Characterization within the module. Learn more about User Characterization.

**Show Advanced Settings** Check to display the Advanced Settings when Next is clicked.

**Back** Return to Cal Wizard Begin dialog. If checked, you can clear the Save Preferences checkbox to see the Begin page when the Cal Wizard begins.

**Note:** The PNA no longer allows ECal isolation to be performed. The inherent isolation of the PNA is better than that attained with correction using an ECal module.
**ECal module not found** dialog box help

Displays an error message indicating the ECal module is not connected or has not been recognized by the network analyzer.

**Retry**  Check the USB connections and click to continue.

**Notes:**

- If your ECal module is not detected, try to unplug, then reconnect the USB connector to the PNA.
- When the ECal module is connected to the network analyzer for the first time, it may take approximately 30 seconds for the analyzer to recognize the module and make it available for calibration.
- For best accuracy, allow the ECal module to warm-up until it indicates READY.
- Agilent 8509x and N4431 ECal modules, when first connected, draw significantly more current than other modules. This could cause the USB to stop working in certain situations. See [USB limitations](#).
- See [Connect ECal Module USB to PNA USB](#).

---

**Select Module and Characterization** dialog box help

**ECal Module**  Select one of the ECal modules that are connected to the PNA.

**Detect Connected ECals**  Click to rescan the USB for ECal modules.

**Available Characterizations**  Select either the Factory Characterization of your ECal module or a [User Characterization](#). Once selected, that characterization becomes the default selection until the PNA is turned OFF and restarted. When restarted, Factory again becomes the default selection.
**Error: Frequency Range** dialog box help

The current cal standards (or ECAL module) does not cover the current frequency range of the measurement. Do one of the following to correct the problem:

- **Frequency**  Change the frequency range of the active channel.
- **Edit**  This selection is not relevant to ECAL modules.
- **Back**  Select a different characterization that covers the required frequency range.
- **Cancel**  Re-characterize the module with an increased frequency range.

**Select DUT Connectors and Cal Kits** dialog box help

If the ECAL module or selected User Characterization has more than one connector type, then the following dialog box is presented which allows you to describe the DUT connector type. Otherwise, click next to proceed to Advanced Settings (if checked) or ECAL Steps.

- **Connectors**
  
  The available connectors are listed for each DUT port.
Advanced Settings dialog box help

Thru #n
Lists the proposed Thru connections to be made during the calibration process. You can change these Thru connections to better suit your test setup.

- The proposed Thru connections are listed automatically.
- Additional Thru connections can be selected for higher accuracy. Learn more.

Add Thru
Click to add a Thru connection. Learn more

Remove Thru
Select a Thru by clicking the "Thru #N" field or the "1st Port / 2nd Port" field. Then click "Remove Thru". This selection is NOT available if the selected Thru is required for the calibration.

1st Port / 2nd Port
Click to change the two ports to be included in the Thru connection. The order of the port numbers (1st or 2nd) is not critical.

Thru Cal Method
Lists the available Thru Cal methods for the specified port pairs. Learn about ECal Thru Methods

Cal Type/ Stds
Click to invoke the View / Modify Properties of Cal dialog box

Do orientation
When this box is checked (default) the PNA automatically senses the model and direction in which an ECal module port is connected to the PNA ports. If power to the ECal module is too low, it will appear as if there is no ECal module connected. If you use low power and are having this problem, clear this check box to provide the orientation manually.

Orientation occurs first at the middle of the frequency range that you are calibrating. If a signal is not detected, it tries again at the lowest frequency in the range. If you have an E8361A or E836xB PNA and do an ECal completely within 10 - 20 MHz OR 60 - 67 GHz, you may need to do orientation manually. There may not be sufficient power to orient the ECal module at those frequencies.

Choose delta match
Available only when a Delta Match Cal is required.

- Check, then click Next to invoke the Select Cal Set for Delta Match dialog box.
- Clear - The Cal Wizard uses the Global Delta Match Cal if available.
## Specify how the ECal module is connected dialog box help

This dialog box appears when the **Do orientation** checkbox in the previous dialog box is cleared.

Click the ECal Port that is connected to each PNA port.

---

## Electronic Calibration Steps dialog box help

**Note:** Beginning in PNA Rev. 6.0, ECal can be performed with External triggers. Learn more. Displays the instructions for each measurement required for calibration.

- **Measure** Measures the ECal standards.
- **Done** Click when last standard has been measured.

## Saving an ECal Calibration

When complete, you can save the new calibration. Learn how.

---

Last modified:

- 4-Sep-2007 Added First time note
- 14-Sep-2007 MX Added UI
- Sept. 12, 2006 MQ Modified images for multiport
ECal User Characterization

- **Overview**
- **How to Perform a User Characterization**
- **Restore ECal Module Memory**

See also [Using ECal](#)

### Overview

A user-characterized ECal module allows you to add adapters to the ECal module, re-measure the standards in the ECal module, INCLUDING the adapters, then add that data to ECal memory. This extends the reference plane from the module test ports to the adapters.

### Why perform a user characterization?

- If you need to use adapters with your ECal module, you could characterize your ECal module with the adapters attached and perform subsequent ECals in a single step.

- If you have a 4-port ECal module, you could configure the module with adapters of different connector types, then perform a user characterization of the module. When you need to test a DUT with a pair of the connector types on your module, calibrate the PNA with a 1-step ECal using the same two connectors on the User-characterized module.

- If you test devices in a fixture, you could embed the characterization of the fixture in the characterization of the module. To do this, during the mechanical calibration portion of the user characterization, calibrate at the reference plane of the device as you would normally calibrate. Then remove the fixturing to be embedded and insert the ECal module to be characterized. When measuring the ECal module, the PNA removes the effects of the fixturing and stores the measurement results in the user characterized ECal module. Subsequent calibrations with that user characterized module will also remove the fixture effects.

### Notes:

- User Characterization does not delete the factory characterization data. The factory data is saved in the ECal module in addition to the user characterization data.

- You can save up to five different user characterizations in a single ECal module. There are memory limitations; the PNA will determine if the contents of a user characterization will fit inside the module before it is performed. **Note:** This is a new feature with PNA Rev. 3.0. Previous versions of PNA will NOT recognize more than one user characterization.

- Both 2-port and 4-port ECal modules support user characterization.

- With PNA release 6.03, a user characterization can now be performed beyond the frequency range of the ECal module. Although this practice is allowed, calibration accuracy with the extended user characterization...
is likely to be degraded. To determine the level of degradation, compare measurements of a variety of devices using a PNA with a mechanical cal kit calibration versus an ECal extended user characterization calibration.

**How to Perform a User Characterization**

**SUMMARY** (A detailed procedure follows.)

1. Select adapters for the module to match the connector configuration of the DUT.

![Diagram](image1)

2. Either calibrate the PNA using mechanical standards or recall an existing Cal Set.

![Diagram](image2)

3. Measure the ECal module, including adapters, as though it were a DUT.

![Diagram](image3)

4. The measurement results are the characterization data that then gets stored inside the module.

![Diagram](image4)
A 2-port PNA can be used to perform a User Characterization on a 4-port ECal module. However, a 4-port ECal module has SIX different port pairs. The PNA must be recalibrated for each port pair that uses unique connector types or gender.

- If all 4 ECal module ports have the same connector type and gender, then only one PNA calibration is required to measure all six port pairs.
- If all 4 ECal module ports have different connector types or gender, then 6 calibrations are required.

When more than one PNA calibration is required during a User Characterization, then ALL calibrations must be performed using the standard Cal Wizard, saved to Cal Sets, and then recalled from Cal Sets DURING the User Characterization.

### Detailed steps to Perform a User Characterization

1. Connect the ECal module to the network analyzer with the USB cable. See [Connect ECal Module USB to PNA USB](#).
2. Allow the module to warm up until it indicates **READY**.
3. **Preset** the analyzer.
4. **Set up the measurement**. For best accuracy, the **IF bandwidth** should be set to **1 kHz** or less.
5. Start and complete the **Characterize ECal Module** Wizard:

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>No Programming commands</strong></td>
</tr>
<tr>
<td>1. <strong>Navigate using</strong> <strong>MENU/DIALOG</strong></td>
<td>1. Click <strong>Calibration</strong></td>
</tr>
<tr>
<td></td>
<td>2. then <strong>Characterize ECal module</strong></td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>No Programming commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>CAL</strong></td>
<td>1. Click <strong>Response</strong></td>
</tr>
<tr>
<td>2. then <strong>[More]</strong></td>
<td>2. the <strong>Cal</strong></td>
</tr>
<tr>
<td>3. then <strong>[ECal]</strong></td>
<td>3. then <strong>More</strong></td>
</tr>
<tr>
<td>4. then <strong>[Characterize ECal module]</strong></td>
<td>4. then <strong>ECal</strong></td>
</tr>
<tr>
<td></td>
<td>5. then <strong>Characterize ECal module</strong></td>
</tr>
</tbody>
</table>
Select Module and Characterization dialog box help

ECal Module  Select one of the ECal modules that are connected to the PNA.

Detect Connected ECals  Click to rescan the USB for ECal modules.

Available Characterizations  Scroll to view all of the parameters of the stored characterizations. Select an empty location or select to overwrite an existing characterization.

Data can NOT be deleted from the ECal module.

Next  Click to continue to the Select Connectors for the Characterization dialog box.

See note regarding extended frequency use.

Select Connectors for the Characterization dialog box help

Note:  When performing an ECal User Characterization, do NOT use a custom connector name that you added to this list. If you need to use a custom-defined connector type, select "Type B", or one of the "Type A" variations from the list of connectors for each port.

Allows you to select the adapters for the ECal module test ports. Select No adapter if no adapter is used on a port.

PORT A  Lists the connector types available for Port A.

PORT B  Lists the connector types available for Port B.

PORT C  Lists the connector types available for Port C (available with a 4-port ECal module).

PORT D  Lists the connector types available for Port D (available with a 4-port ECal module).

Next  Click to continue to the Calibrations to perform or recall dialog box.
Calibrations to perform or recall dialog box help

The PNA must be calibrated before measuring the ECal module and necessary adapters. This dialog box displays the number and types of mechanical calibrations required for the characterization.

**Guide me through this cal now**  Click to perform a Guided calibration. A calibration kit is required for each connector type.

**Note:** TRL calibrations can NOT be performed on a 4-port PNA during the calibration portion of a User Characterization. However, this type of Cal can be performed using the Cal Wizard, saved to a Cal Set, then recalled at this point in the User Characterization.

If more than one calibration is required, this selection is not available. See Note.

Let me recall this cal from a cal set  Click to select an existing Cal Set. You cannot select a Cal Set that is currently in use. Learn more about Using Cal Sets.

**Next**  Click to continue to either the Select Cal Kits or the Select Cal Set dialog box.

Select Cal Kits dialog box help

Provides a list of calibration kits to perform the calibration. Select the Cal Kit you will use for each port.

**Enable Unknown Thru for characterizing the module**  Check to enable. This reduces the number of steps required to characterize the THRU standard. This setting is available only on PNA models with one reference receiver per test port.

**Next**  Click to continue to the Select Cal Set dialog box.
Select Cal Set dialog box help

The calibration that you perform will be written to a Cal Set. This dialog box allows you to select a Cal Set to overwrite, or to write to a new Cal Set. The current choice is visible below the Select Cal Set button.

- **Select Cal Set** Click to open the Select A Cal Set dialog box.
- **Create new Cal Set** Check to create a new Cal Set to store the calibration. Clear to select and overwrite a stored Cal Set.
- **Next** Click to continue to the Guided Calibration Steps dialog box.

**Note:** Remember the Cal Set name for future reference.

Guided Calibration Steps dialog box help

Instructs you to connect each calibration standard to the measurement port.

- **Measure** Click to measure the standard.
- **Back** Click to repeat one or more calibration steps.
- **Done** Click after a standard is re-measured and all measurements for the calibration are complete.
- **Next** Click to continue to the next calibration step. (Does not measure the standard.)
- **Cancel** Exits Calibration Wizard.

The Specify nominal delay or Guided Calibration completed dialog box appears when the steps are completed.
Specify nominal delay dialog box help

This dialog ONLY appears when Adapter Removal or Unknown Thru calibrations are performed.

The following values were estimated from the measurement. Most of the time, they are adequate. However, for CW sweep or frequency sweep with large step sizes, the accuracy of the values may be improved.

**Nominal adapter delay**  To improve this value, measure and record the delay of the adapter with a dense step size. Enter that value here.

**Nominal phase offset**  (Waveguide ONLY). To improve this value, measure and record the phase offset of the Waveguide adapter with dense step size. Enter that value here.

When one connector is coax and the other connector is waveguide, the phase offset has an ambiguity of 180 degrees. For consistency, the estimate provided here is always between 0 and 180 degrees. You can change this estimate to any value between -180 degrees and +180 degrees.

Guided Calibration completed dialog box help

Allows you to finish the calibration and continue to the next characterization steps.

**No. Finish now**  Select to save Cal Set data.

**Yes**  Allows selection of Save options.

**Next**  Click to continue to the Exit to Inspect Quality of Calibration dialog box.
## Exit to Inspect Quality of Calibration dialog box help

Allows you to exit User Characterization to validate the calibration before proceeding with the characterization.

- **Back** Allows you to repeat calibration.
- **Next** Click to continue to the Characterization Steps dialog box.
- **Cancel** Exits the Calibration.

To return to the current step:

2. In the Select user number for new characterization dialog box, click Next.
3. In the Select Connectors for Characterization dialog box, click Next. (Previous entry is stored in memory.)
4. In the Calibrations to perform or recall dialog box, recall the Cal Set that you just performed.

## Characterization Steps dialog box help

Describes the instructions for each measurement required for characterization.

- **Measure** Measures the ECal module.
- **Next** Click to continue to the Information for the New Characterization dialog box when measurements are complete.
**Information for the New Characterization** dialog box help

Allows you to describe the properties of the User Characterization.

**Next** Click to continue to the [Write Characterized Data to the ECal module](#) dialog box.

To minimize the number of characters, use the following 3-character codes to describe the connectors listed.

<table>
<thead>
<tr>
<th>Connector Type</th>
<th>3-Character Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 mm female</td>
<td>10F</td>
</tr>
<tr>
<td>1.0 mm male</td>
<td>10M</td>
</tr>
<tr>
<td>1.85 mm female</td>
<td>18F</td>
</tr>
<tr>
<td>1.85 mm male</td>
<td>18M</td>
</tr>
<tr>
<td>2.4 mm female</td>
<td>24F</td>
</tr>
<tr>
<td>2.4 mm male</td>
<td>24M</td>
</tr>
<tr>
<td>2.92 mm female</td>
<td>29F</td>
</tr>
<tr>
<td>2.92 mm male</td>
<td>29M</td>
</tr>
<tr>
<td>3.5 mm female</td>
<td>35F</td>
</tr>
<tr>
<td>3.5 mm male</td>
<td>35M</td>
</tr>
<tr>
<td>7-16 female</td>
<td>16F</td>
</tr>
<tr>
<td>7-16 male</td>
<td>16M</td>
</tr>
<tr>
<td>Type F female</td>
<td>F7F</td>
</tr>
<tr>
<td>Type F male</td>
<td>F7M</td>
</tr>
<tr>
<td>N50 female</td>
<td>N5F</td>
</tr>
<tr>
<td>N50 male</td>
<td>N5M</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-----</td>
</tr>
<tr>
<td>N75 female</td>
<td>N7F</td>
</tr>
<tr>
<td>N75 male</td>
<td>N7M</td>
</tr>
<tr>
<td>APC 7</td>
<td>7MM</td>
</tr>
<tr>
<td>K-band waveguide</td>
<td>KBW</td>
</tr>
<tr>
<td>P-band waveguide</td>
<td>PBW</td>
</tr>
<tr>
<td>Q-band waveguide</td>
<td>QBW</td>
</tr>
<tr>
<td>R-band waveguide</td>
<td>RBW</td>
</tr>
<tr>
<td>U-band waveguide</td>
<td>UBW</td>
</tr>
<tr>
<td>V-band waveguide</td>
<td>VBW</td>
</tr>
<tr>
<td>W-band waveguide</td>
<td>WBW</td>
</tr>
<tr>
<td>X-band waveguide</td>
<td>XBW</td>
</tr>
</tbody>
</table>

**Write Characterized Data to the ECal module memory** dialog box help

The PNA writes User Characterization and factory characterization data to the ECal module memory. For more information, see [Restore ECal module memory](#).

**Write** Click to write data into the ECal module.

The **Summary of new user characterization** dialog box opens after data is saved to module.

Data can NOT be deleted from the ECal module.
### Summary of new user characterization dialog box help

Verify the status of the ECaI User Characterization.

- ECaI module model number
- summary from user characterization

**Cancel** Click to exit (characterization complete).

**Finish** Click to exit (characterization complete).

### Restore ECaI Module Memory

When user-characterized data is written to the ECaI module, the entire contents of ECaI memory is also written to the PNA hard drive. In the unlikely event that your ECaI module memory is lost, you can restore the user-characterized data to ECaI memory.

### How to Restore ECaI Module Memory

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td>No Programming commands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td></td>
</tr>
<tr>
<td>2. then Service</td>
<td></td>
</tr>
<tr>
<td>3. then Utilities</td>
<td></td>
</tr>
<tr>
<td>4. then Restore...</td>
<td></td>
</tr>
</tbody>
</table>

**For PNA-X and 'C' models**

<table>
<thead>
<tr>
<th></th>
<th>No Programming commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press SYSTEM</td>
<td></td>
</tr>
<tr>
<td>2. then [Service]</td>
<td></td>
</tr>
<tr>
<td>3. then [Utilities]</td>
<td></td>
</tr>
<tr>
<td>4. then [Restore...]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Click Utility</td>
<td></td>
</tr>
<tr>
<td>2. then System</td>
<td></td>
</tr>
<tr>
<td>3. then Service</td>
<td></td>
</tr>
<tr>
<td>4. then Utilities</td>
<td></td>
</tr>
</tbody>
</table>
5. then **Restore...**

### Module to be restored dialog box help

Verify the Serial number of the module to be restored. If two modules are connected to the PNA, choose the one to have data restored.

**Next**  Click to write data to the module.

---

**Last modified:**

- **April 25, 2007**  Added note about can NOT delete data.
- **Sept. 20, 2006**  MX Modified for cross-browser
TRL Calibration

TRL (Thru, Reflect, Line) represents a family of calibration techniques that measure two transmission standards and one reflection standard to determine the 2-port 12-term error coefficients. For example, TRM (Thru, Reflect, Match), LRL (Line, Reflect, Line), LRM (Line, Reflect, Match) are all included in this family.

The traditional SOLT calibration measures one transmission standard (T) and three reflection standards (SOL) to determine the same error coefficients.

- **Why Perform a TRL Cal?**
- **The TRL Calibration Process**
- **TRL Cal Kits**
- **Cal Standards Used in TRL**
- **TRL in 4-port PNA a**

### Why Perform a TRL Cal?

TRL calibration is extremely accurate, in most cases more accurate than an SOLT cal. However, very few calibration kits contain TRL standards. TRL Cal is most often performed when you require a high level of accuracy and do not have calibration standards in the same connector type as your DUT. This is usually the case when using test fixtures, or making on-wafer measurements with probes. Therefore, in some cases you must construct and characterize standards in the same media type as your DUT configuration. It is easier to manufacture and characterize three TRL standards than the four SOLT standards.

Another advantage of TRL calibration is that the TRL standards need not be defined as completely and accurately as the SOLT standards. While SOLT standards are completely characterized and stored as the standard definition, TRL standards are modeled, and not completely characterized. However, TRL cal accuracy is directly proportional to the quality and repeatability of the TRL standards. Physical discontinuities, such as bends in the transmission lines and beads in coaxial structures, will degrade the TRL calibration. The connectors must be clean and allow repeatable connections.

To learn more about Cal Standard requirements, see Cal Standards Used in TRL.

### The TRL Cal Process

Although TRL can be performed using the Cal Wizard Unguided Cal selection, the following process uses the easier SmartCal selection. Both selections require that you already have TRL calibration standards defined and included in a PNA cal kit.

1. Preset the PNA
2. Set up a S-parameter measurement and the desired stimulus settings.
3. Click Calibration / Calibration Wizard
4. Click **SmartCal (Guided Cal)**.

5. **Select the DUT connectors and Cal Kit** for each port. The LOWEST port number of each **port pair** MUST include TRL standards. TRL appears as the Cal Method.

6. Check **Modify Cal**, **Next**, then **View/Modify** to change **default TRL options** if necessary.

7. Follow the prompts to complete the calibration.

8. **Check the accuracy** of the calibration

---

**TRL Cal Kits**

Agilent Technologies offers two cal kits that include the required standards to perform a TRL calibration: 85050C (APC 7mm) and 85052C (3.5mm). Both kits include the traditional Short, Open, and Load standards. (The Thru standard, not actually supplied, assumes a **zero-length Thru**). In addition, the kits include an airline which is used as the LINE standard. To use the airline, the kits include an airline body, center conductor, and insertion / extraction tools. The APC 7 kit includes an adapter to connect the airline to the APC connector.

**Cal Standards Used in TRL**

These standards must be defined in your TRL cal kit:

**THRU**

**Note:** All **THRU calibration methods** are supported in a TRL Cal **EXCEPT** Unknown Thru.

- The THRU standard can be either a zero-length or non-zero length. However, a zero-length THRU is more accurate because it has zero loss and no reflections, by definition.

- The THRU standard cannot be the same electrical length as the LINE standard.

- If the insertion phase and electrical length are well-defined, the THRU standard may be used to **set the reference plane**.

- Characteristic impedance of the THRU and LINE standards defines the reference impedance of the calibration.

**REFLECT**

- The REFLECT standard can be anything with a high reflection, as long as it is the same when connected to both PNA ports.

- The actual magnitude of the reflection need not be known.

- The phase of the reflection standard must be known within 1/4 wavelength.

- If the magnitude and phase of the reflection standard are well-defined, the standard may be used to **set the reference plane**.

**LINE**
The LINE and THRU standards establish the reference impedance for the measurement after the calibration is completed. TRL calibration is limited by the following restrictions of the LINE standard:

- Must be of the same impedance and propagation constant as the THRU standard.
- The electrical length need only be specified within 1/4 wavelength.
- Cannot be the same length as the THRU standard.
- A TRL cal with broad frequency coverage requires multiple LINE standards. For example, a span from 2 GHz to 26 GHz requires two line standards.
- Must be an appropriate electrical length for the frequency range: at each frequency, the phase difference between the THRU and the LINE should be greater than 20 degrees and less than 160 degrees. This means in practice that a single LINE standard is only usable over an 8:1 frequency range (Frequency Span / Start Frequency). Therefore, for broad frequency coverage, multiple lines are required.
- At low frequencies, the LINE standard can become too long for practical use. The optimal length of the LINE standard is 1/4 wavelength at the geometric mean of the frequency span (square root of f1 x f2).

**MATCH**

If the LINE standard of appropriate length or loss cannot be fabricated, a MATCH standard may be used instead of the LINE.

- The MATCH standard is a low-reflection termination connected to both Port 1 and Port 2.
- The MATCH standard may be defined as an infinite length transmission line OR as a 1-port low reflect termination, such as a load.
- When defined as an infinite length transmission line, both test ports must be terminated by a MATCH standard at the same time. When defined as a 1-port load standard, the loads are measured separately. The loads are assumed to have the same characteristics.
- The impedance of the MATCH standard becomes the reference impedance for the measurement. For best results, use the same load on both ports. The load may be defined using the data-based definition, the arbitrary impedance definition, or the fixed load definition.

See Modify Calibration Kits for detailed information about creating and modifying Calibration kit definitions.

Find more information about TRL standards at [http://www.tm.agilent.com](http://www.tm.agilent.com). Click "Technical Support". Use "Application Notes" to search for App Note 8510-5A (part number 5956-4352). Although the application note is written for the Agilent 8510 Series Network Analyzers, it also applies to the PNA.

**TRL on a 4-port PNA and with an External Test Set**

Beginning with the PNA code revision 5.25, TRL CAN be performed on a 4-port PNA and with an External Test Set enabled. Previously, a TRL calibration required a PNA with a reference receiver for each test port. With the new TRL method, a Delta Match Calibration is first performed and applied.

The accuracy of this TRL cal greatly depends on the accuracy of the Delta Match Calibration. With an accurate Delta Match Calibration, the difference in accuracy between a traditional TRL cal and this TRL cal is negligible.
How to Perform a TRL Cal on a multiport PNA

1. Click Calibration, Cal Wizard.

2. Select a TRL cal kit for the ports to be calibrated.

3. During the calibration, the Cal Wizard prompts you for a valid Delta Match Cal.

---

Last modified:

9/12/06  with Ext Test Set
Calibration Preferences Wizard

The following Cal Preferences are set from this Wizard:

1. **Whether or not to show the first 'Method' Page of the Cal Wizard**
2. **Select and order the Cal Types that are available during a SmartCal with Mechanical Standards**
3. 

To change either of these choices, you must select *Yes, Enable the calibration preferences* at the first Wizard page.

### How to change Cal Preferences

Programming commands are NOT available for the preference settings discussed in this topic, although there are other Cal Preferences that can be set remotely.

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Calibration</td>
</tr>
<tr>
<td></td>
<td>2. then Cal Preferences</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press CAL</td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then [Start Cal]</td>
<td>2. then Cal</td>
</tr>
<tr>
<td>3. then [Preferences]</td>
<td>3. then Start Cal</td>
</tr>
<tr>
<td></td>
<td>4. then Preferences</td>
</tr>
</tbody>
</table>

![Calibration Preferences Wizard dialog](image)
**Cal Preferences Wizard** dialog box help

Use this dialog to change either of the following preferences:

- Show or Hide the first page of the Cal Wizard
- Select order of calibrations that are offered.

To change either of these choices, you must select **Yes, Enable the calibration preferences.**

---

**Cal Preferences of...** dialog box help

Use this dialog to change which Cal method to perform.

After making this selection, the first page of the Cal Wizard will not be shown on subsequent calibrations.

If you ONLY want to change the order of Cal Types that are offered (next page of the Cal Preferences), you must do the following:

1. Select one of these choices and click **Next**.
2. Select and order the Cal Types, then click **Next**
3. Click **Finish**
4. Click **Cal**, then **Cal Wizard**.
5. On the first Cal Wizard page that shows, click **Back**, then clear the **Preferences** checkbox.
This dialog is used to set which Cal Types are available, and the order in which they are selected as the default choice, during a SmartCal with Mechanical Standards. This setting is also used to set the default Cal Type for Guided calibrations using SCPI or COM.

The specified Cal Type order should allow you to make fewer changes to the Cal Type during a SmartCal with Mechanical Standards.

For example, in the above image, the first Cal Type on the list is TRL. When doing a SmartCal with Mechanical Standards:

- If a TRL Cal Kit is available for the specified DUT connectors, then TRL becomes the default Cal Type.
- If a TRL Cal Kit is NOT available, then the second Cal Type on the list (SOLT) is evaluated for compatibility with the available Cal Kits, and so forth with the Cal Types that remain on the list.
- If TRL is removed from the list, that Cal Type is NOT available for selection during a SmartCal with Mechanical Standards.

Learn more about Cal Types.

See where you choose Cal Type during a SmartCal

Prioritized list of choices for default Cal Type  Shows the current list of Cal Types and the order in which they will be selected for Mechanical calibrations.

Change  Click to invoke the Modify list of default Cal Types dialog.

Restore factory defaults  Returns the list to the original selections and order. The factory defaults are in order of accuracy from highest (TRL) to lowest (QSOLT).

Cancel  Closes the dialog without making changes.

Notes:

- Your Cal Type settings are saved only until the PNA application is closed. When re-opened, the factory default settings are restored.

Learn more about QSOLT Calibration
Modify list of default Cal Types dialog box help

Use this dialog to Add, Remove, and re-order the available Cal Types. There must be at least ONE selected Cal Type to perform a SmartCal with Mechanical Standards.

Unselected Cal Types  Cal Types in this list will not be presented as a choice during a Calibration.

Selected Cal Types  Cal Types in this list will be presented, in order, as the default choice during a Calibration. Click a Cal Type to select it, then click the following buttons to perform that operation.

Add / Remove buttons  Click to Add and Remove the selected Cal Types from the Selected Cal Types list.

Move Up / Down  Click to re-order the Selected Cal Types list.

Cal Preferences Complete dialog box help

Either Enable or Disable Cal Preferences.

Do you want to select ONLY Cal Type Preferences and continue to show the first Cal Wizard page? Learn how.

Last Modified:

1-Jan-2007    MX added UI
Measurement Errors

You can improve accuracy by knowing how errors occur and how to correct for them. This topic discusses the sources of measurement error and how to monitor error terms.

- **Drift Errors**
- **Random Errors**
- **Systematic Errors**
  - 3-Port Error Terms
  - 4-Port Error Terms
  - Monitoring Error Terms

---

**Drift Errors**

Drift errors are due to the instrument or test-system performance changing after a calibration has been done.

Drift errors are primarily caused by thermal expansion characteristics of interconnecting cables within the test set and conversion stability of the microwave frequency converter and can be removed by re-calibrating.

The time frame over which a calibration remains accurate is dependent on the rate of drift that the test system undergoes in your test environment.

Providing a stable ambient temperature usually minimizes drift. For more information, see Measurement Stability.

**Random Errors**

Random errors are not predictable and cannot be removed through error correction. However, there are things that can be done to minimize their impact on measurement accuracy. The following explains the three main sources of random errors.

**Instrument Noise Errors**

Noise is unwanted electrical disturbances generated in the components of the analyzer. These disturbances include:

- Low level noise due to the broadband noise floor of the receiver.
- High level noise or jitter of the trace data due to the noise floor and the phase noise of the LO source inside the test set.

You can reduce noise errors by doing one or more of the following:

- Increase the source power to the device being measured - ONLY reduces low-level noise.
- Narrow the IF bandwidth.
- Apply several measurement sweep averages.

**Switch Repeatability Errors**

Mechanical RF switches are used in the analyzer to switch the source attenuator settings. Sometimes when mechanical RF switches are activated, the contacts close differently from when they were previously activated. When this occurs, it can adversely affect the accuracy of a measurement. You can reduce the effects of switch repeatability errors by avoiding switching attenuator settings during a critical measurement.

**Connector Repeatability Errors**

Connector wear causes changes in electrical performance. You can reduce connector repeatability errors by practicing good connector care methods. See Connector Care.

**Systematic Errors**

Systematic errors are caused by imperfections in the analyzer and test setup.

- They are repeatable (and therefore predictable), and are assumed to be time invariant.
- They can be characterized during the calibration process and mathematically reduced during measurements.
- They are never completely removed. There are always some residual errors due to limitations in the calibration process. The residual (after measurement calibration) systematic errors result from:
  - imperfections in the calibration standards
  - connector interface
  - interconnecting cables
  - instrumentation

**Reflection** measurements generate the following three systematic errors:

- **Directivity**
- **Source Match**
- **Frequency Response Reflection Tracking**

**Transmission** measurements generate the following three systematic errors:

- **Isolation**
- **Load Match**
- **Frequency Response Transmission Tracking**
Notes about the following Systematic Error descriptions:

- The figures for the following six systematic errors show the relevant hardware configured for a forward measurement. For reverse measurements, internal switching in the analyzer makes Port 2 the source and Port 1 the receiver. 'A' becomes the transmitted receiver, 'B' becomes the reflected receiver, and 'R2' becomes the reference receiver. These six systematic errors, times two directions, results in 12 systematic errors for a two port device.

- For simplicity, it may be stated that ONE standard is used to determine each systematic error. In reality, ALL standards are used to determine ALL of the systematic errors.

- The following describes an SOLT calibration. This does not apply to TRL, or other types of calibration.

**Directivity Error**

All network analyzers make reflection measurements using directional couplers or bridges.

With an ideal coupler, only the reflected signal from the DUT appears at the 'A' receiver. In reality, a small amount of incident signal leaks through the forward path of the coupler and into the 'A' receiver. This leakage path, and any other path that allows energy to arrive at the 'A' receiver without reflecting off the DUT, contributes to directivity error.

![Diagram of Directivity Error](image)

**How the Analyzer Measures and Reduces Directivity Error.**

1. During calibration, a load standard is connected to Port 1. We assume no reflections from the load.

2. The signal measured at the 'A' receiver results from the incident signal leakage through the coupler and other paths.

3. Directivity error is mathematically removed from subsequent reflection measurements.

**Isolation Error**

Ideally, only signal transmitted through the DUT is measured at the 'B' receiver.

In reality, a small amount of signal leaks into the 'B' receiver through various paths in the analyzer.

The signal leakage, also known as crosstalk, is isolation error which can be characterized and reduced by the analyzer.
How the Analyzer Measures and Reduces Isolation Error

1. During calibration, load standards are connected to both Port 1 and Port 2.
2. The signal measured at the 'B' receiver is leakage through various paths in the analyzer.
3. This isolation error is mathematically removed from subsequent transmission measurements.

Source Match Error

Ideally in reflection measurements, all of the signal that is reflected off of the DUT is measured at the 'A' receiver. In reality, some of the signal reflects off the DUT, and multiple internal reflections occur between the analyzer and the DUT. These reflections combine with the incident signal and are measured at the 'A' receiver, but not at the 'R' receiver.

This measurement error is called source match error which can be characterized and reduced by the analyzer.

How the Analyzer Measures and Reduces Source Match Error

1. During calibration, all reflection standards are connected to Port 1. Known reflections from the standards are measured at the 'A' receiver.
2. Complex math is used to calculate source match error.
3. Source match error is mathematically removed from subsequent reflection and transmission measurements.

**Load Match Error**

Ideally in transmission measurements, an incident signal is transmitted through the DUT and is measured at the 'B' receiver. In reality, some of the signal is reflected off of Port 2 and other components and is not measured at the 'B' receiver. This measurement error is called load match error which can be characterized and reduced by the analyzer.

![Diagram showing load match error](image)

**How the Analyzer Measures and Reduces Load Match Error**

1. The Port 1 and Port 2 test connectors are mated together for a perfect zero-length thru connection. If this is not possible, a characterized thru adapter is inserted. This allows a known amount of incident signal at Port 2.

2. The signal measured at the 'A' receiver is reflection signal off of Port 2

3. The resulting load match error is mathematically removed from subsequent transmission and reflection measurements.

**Frequency Response Reflection Tracking Error**

Reflection measurements are made by comparing signal at the 'A' receiver to signal at the 'R1' receiver. This is called a ratio measurement or "A over R1" (A/R1).

For ideal reflection measurements, the frequency response of the 'A' and 'R1' receivers would be identical. In reality, they are not, causing a frequency response reflection tracking error. This is the vector sum of all test variations in which magnitude and phase change as a function of frequency. This includes variations contributed by:

- signal-separation devices
- test cables
- adapters
• variations between the reference and test signal paths

Frequency response reflection tracking error can be characterized and reduced by the analyzer.

---

**How the Analyzer Measures and Reduces Frequency Response Reflection Tracking Error.**

1. During calibration, all reflection standards are used to determine reflection tracking.

2. The average 'A' receiver response is compared with the 'R1' receiver response.

3. Complex math is used to calculate Frequency Response Reflection Tracking Error (see the following diagram). This frequency response reflection tracking error is mathematically removed from subsequent DUT measurements.

---

**Note:** In reflection response calibrations, only a single calibration standard is measured (open or short) and thus only its contribution to the error correction is used.

---

**Frequency Response Transmission Tracking Error**

Transmission measurements are made by comparing signal at the 'B' receiver to signal at the 'R1' receiver. This is called a ratio measurement or "B over R1" (B/R1).

For ideal transmission measurements, the frequency response of the 'B' and 'R1' receivers would be identical. In reality, they are not, causing a frequency response transmission tracking error. This is the vector sum of all test variations in which magnitude and phase change as a function of frequency. This includes variations contributed...
by:

- signal-separation devices
- test cables
- adapters
- variations between the reference and test signal paths

Frequency response transmission tracking error can be characterized and reduced by the analyzer.

How the Analyzer Measures and Reduces Frequency Response Transmission Tracking Error.

1. During calibration, the Port 1 and Port 2 test connectors are mated together for a perfect zero-length thru connection. If this is not possible, a characterized thru adapter is inserted. This allows a known amount of incident signal to reach Port 2.

2. Measurements are made at the 'B' and 'R1' receivers.

3. Complex math is used to calculate Frequency Response Transmission Tracking Error (see the following diagram). This frequency response transmission tracking error is mathematically removed from subsequent DUT measurements.
3-Port Error Terms

The following flow diagram displays the 3-port error term model:

where:

- \( E \) = error term
- DIR = Directivity
- MAT = Forward Source Match and Reverse Load Match
- TRK = Forward Reflection Tracking and Reverse Transmission Tracking

4-Port error terms

A full 4-port calibration requires the following terms:

Learn about the port numbering convention for error terms.
<table>
<thead>
<tr>
<th>Port</th>
<th>TTRK 3,1</th>
<th>TTRK 3,2</th>
<th>RTRK 3,3</th>
<th>TTRK 3,4</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>XTLK 3,1</td>
<td>XTLK 3,2</td>
<td>SRM 3,3</td>
<td>XTLK 3,4</td>
</tr>
<tr>
<td>4</td>
<td>LDM 4,1</td>
<td>LDM 4,2</td>
<td>LDM 4,3</td>
<td>DIR 4,4</td>
</tr>
<tr>
<td></td>
<td>TTRK 4,1</td>
<td>TTRK 4,2</td>
<td>TTRK 4,3</td>
<td>RTRK 4,4</td>
</tr>
<tr>
<td></td>
<td>XTLK 4,1</td>
<td>XTLK 4,2</td>
<td>XTLK 4,3</td>
<td>SRM 4,4</td>
</tr>
</tbody>
</table>

Reflection terms

- DIR: Directivity
- RTRK: Reflection Tracking
- SRM: Source Match

Transmission terms

- LDM: Load Match
- TTRK: Transmission Tracking
- XTLK: Cross Talk

**How can we measure only 3 THRU connections?**

On a 4-port PNA, a full 4-port cal can be performed while measuring only 3 THRU connections. Measuring more than 3 THRU connections will give higher accuracy.

By measuring all of the reflection terms, and 3 transmission THRU connections, there is adequate information available to calculate the remaining transmission terms. The following is a high level explanation of the concept. The actual calculations are much more complex.

To simplify, let's substitute letters (A,B,C,D) for port numbers from the diagram above so that they can be combined without confusion. Also for simplicity, let's assume that the source match and directivity errors are zero.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>AA</td>
<td>AB</td>
<td>AC</td>
<td>AD</td>
</tr>
<tr>
<td>B</td>
<td>BA</td>
<td>BB</td>
<td>BC</td>
<td>BD</td>
</tr>
<tr>
<td>C</td>
<td>CA</td>
<td>CB</td>
<td>CC</td>
<td>CD</td>
</tr>
<tr>
<td>D</td>
<td>DA</td>
<td>DB</td>
<td>DC</td>
<td>DD</td>
</tr>
</tbody>
</table>

- The reflection errors are all measured (AA, BB, CC, DD).
- Lets assume we measure a THRU between ports AB, AC, AD. The reverse direction for these THRU are
also measured at the same time (BA, CA, DA).

- The terms left to calculate are BC, CB, BD, DB, CD, DC.

The following shows how the BC term is calculated from BA and AC:

\[
\frac{BA \times AC}{AA} = \frac{B \times C}{A} = BC
\]

Similarly:

- CB is calculated from CA and AB
- BD is calculated from BA and AD
- DB is calculated from AB and DA
- CD is calculated from CA and AD
- DC is calculated from DA and AC

**Monitoring Error Terms using Cal Set Viewer**

You can use **Cal Set Viewer** to monitor the measured data and the calculated error term. This will help to determine the health of your PNA and the accuracy of your measurements.

By printing or saving the error terms, you can periodically compare current error terms with previously recorded error terms that have been generated by the same PNA, measurement setup, and calibration kit. If previously generated values are not available, refer to Typical Error Term Data in Appendix A, "Error Terms", of the Service Guide.

**Note**: The service guide for your PNA is available at [http://www.agilent.com/find/pna](http://www.agilent.com/find/pna). It is also on the CDROM that was shipped with your PNA.

- A stable system should generate repeatable error terms over about six months.
- A sudden shift in error terms over the same frequency range, power, and receiver settings, may indicate the need for troubleshooting system components. For information on troubleshooting error terms, see Appendix A, "Error Terms", of the Service Guide.
- A subtle, long-term shift in error terms often reflects drift or connector and cable wear. The cure is often as simple as cleaning and gauging connectors or inspecting cables.

**Viewing Cal Set Data**

- Existing measurement traces are unaffected by the Cal Set Viewer.
- The Cal Set data trace is presented in the highest unused channel number (usually 32) in the active window.
- The Cal Set data trace is labeled as S11 in the status bar regardless of the type of error term or standard.
- Only one Cal Set error term or standard data can be viewed at a time. However, a data trace can be stored into memory and then compared to other data traces.
How to access Cal Set Viewer

Using front-panel HARDKEY [softkey] buttons

For **N5230A** and **E836xA/B** models

1. Navigate using MENU/DIALOG

For **PNA-X** and **'C'** models

1. Press **CAL**
2. then **[Manage Cals]**
3. then **[Cal Set Viewer]**

Using a mouse with PNA Menus

1. Click **Calibration**
2. then **Cal Set Viewer**
1. Click **Response**
2. then **Cal**
3. then **Manage Cals**
4. then **Cal Set Viewer**

How to use Cal Set Viewer

![Cal Set Viewer interface](image)

1. Use the down arrow to select a Cal Set. Then click either:
   - **Error Terms** - calculated data.
   - **Standards** - the raw measurement data of the Standard. **ONLY** available with Unguided Cal (not ECal or Guided Cal).

2. Use the down arrow to select an error term or standard to view.
3. Select the **Enable** check box to view the data on the PNA screen.

**Port numbering convention** for error terms is the same as for S-Parameters:

**E Term (Receiver, Source)** with the following exceptions:

- Load Match (2,1) - The match of port 2 which is measured by making an S11 measurement.
- Load Match (1,2) - The match of port 1 which is measured by making an S22 measurement.
- Transmission Tracking (2,1) - The port 2 receiver relative to the port 1 reference. (source=port 1).
- Transmission Tracking (1,2) - The port 1 receiver relative to the port 2 reference. (source=port 2).
• And so forth for multiport calibrations.
Modify Calibration Kits

You can create or modify calibration kit files using Advanced Modify Cal Kits.

- About Modifying Calibration Kits
- Creating a New Cal Kit from an Existing Cal Kit
- Creating Custom Calibration Kits using a New Connector Family
- How to Modify Cal Kits
- Calibration Class Assignments
- Waveguide Cal Kits

Note: For a detailed discussion of Cal Kits and standards, see App Note 1287-11.

See other Calibration Topics

About Modifying Calibration Kits

You can modify calibration kit files or create a custom one.

Note: You CAN modify Data-based Cal Kits. Learn how.

For most applications, the default calibration kit models provide sufficient accuracy for your calibration. However, several situations exist that may require you to create a custom calibration kit:

- Using a connector interface different from those used in the predefined calibration kit models.
- Using standards (or combinations of standards) that are different from the predefined calibration kits. For example, using three offset SHORTs instead of an OPEN, SHORT, and LOAD to perform a 1-port calibration.
- Improving the accuracy of the models for predefined kits. When the model describes the actual performance of the standard, the calibration is more accurate. (Example: A 7 mm LOAD is determined to be 50.4O instead of 50.0O.)
- Modifying the THRU definition when performing a calibration for a non-insertable device.
- Performing a TRL calibration.

Creating a New Cal Kit from an Existing Cal Kit

You can create a new custom Cal Kit using a copy of an existing Cal Kit as a starting point. Here is how:

1. From the Edit PNA Cal Kits dialog, click Import Kit to load the Cal Kit you want to use as a starting point. A "Duplicate Name..." message appears. Click OK to load a duplicate copy of the Cal kit into the last position of the Edit PNA Cal Kits dialog.
2. Select the imported kit.

3. Click Edit Kit, then change the Cal Kit Name and Description.

4. Click Installed Kits - Save As to save the new Cal Kit to a .ckt file.

5. Recommended: Also click Edit PNA Cal Kits - Save As to save the entire collection of Cal Kits to a .wks file.

6. If using a new or modified connector, click Change Family to change the connector family.

7. Click Add or Edit to change connector descriptions and parameters.

8. Make modifications to your new custom Cal Kit as required. Save your work by clicking Installed Kits - Save As

**Note:** Custom Cal Kits must be imported after a firmware upgrade. [Learn more](#)

**Creating Custom Calibration Kits using a New Connector Family**

To create a custom calibration kit that uses a new connector type, you must first define the connector family. The connector family is the name of the connector-type of the calibration kit, such as:

- APC7
- 2.4 mm
- Type-N (50O)

Although more than one connector family is allowed, it is best to limit each calibration kit to only one connector family.

If you are using a connector family that has male and female connectors, include definitions of both genders. If you are using a family with no gender, such as APC7, only one connector definition is required.

Use the following steps to create a custom calibration kit:

1. In the Edit PNA Cal Kits dialog box, click Insert New to add the new connector family.

2. In the Edit Kit dialog box:
   - Type the Kit Description for the custom cal kit.
   - Click Add in the Connectors section of the dialog box.

3. In the Add Connector dialog box:
   - Type a Connector Family name.
   - Type a Description of the connector.
   - Select the Gender of one of the connectors.
   - Type the minimum and maximum Frequency Range.
• Type the Impedance.
• Click the down-arrow to select the Media.
• Type the cut-off frequency.
• Click Apply.
• Click OK.

4. If you need to add another connector gender, in the Edit Kit dialog box:
   • Click Add in the Connectors section again for the next connector gender.

5. If you are adding another connector gender, repeat step 3.

Note: If you have male and female versions of the connector family, you probably do NOT also have a NO GENDER version.

6. Now that the connector family is added to the custom cal kit, you are ready to add new calibration standards. In the Edit Kit dialog box:
   • Under the list of standards, click Add.

7. In the Add Standard dialog box:
   • Select the type of standard (OPEN, SHORT, LOAD, or THRU).
   • Click OK.

8. In the Edit/Add Standards dialog box:
   • Complete the information in the dialog box for the standard you selected. Note that for banded standards, the start and stop frequency may be different than the frequency range of the specified connector. Edit the start and stop frequencies as needed.
   • Click OK when all the settings are correct.

9. Repeat steps 6 - 8, as necessary, to add all standards and definitions to the new custom cal kit.

10. Assign each of the standards to a calibration class. This is done through the Modify Calibration Class Assignment dialog box.

11. Click File, PrintToFile. PrintToFile will generate a .prn file (ascii file with comma delimiters) that can be imported into a spreadsheet.

12. Import the .prn file into an application such as Microsoft Excel, and print the results.
13. Use the spreadsheet to verify that each standard in the kit belongs to the same connector family and the gender of each standard is properly specified. It is important that the connectors and genders for your standards are correctly defined and verified in order for your SmartCal (guided calibrations) to work properly.

How to Modify Cal Kits

The series of dialog boxes that follow allow you to modify the standard definitions or class assignments of calibration kit files.

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY (softkey) buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Calibration</td>
</tr>
<tr>
<td></td>
<td>2. then Advanced Modify Cal Kit</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press CAL</td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then [More]</td>
<td>2. then Cal</td>
</tr>
<tr>
<td>3. then [Cal Kit]</td>
<td>3. then More</td>
</tr>
<tr>
<td></td>
<td>4. then Cal Kit</td>
</tr>
</tbody>
</table>

Edit PNA Cal Kits
Edit PNA Cal Kits dialog box help

Provides access to all Agilent cal kits and allows modification of their standard definitions.

### PNA Cal Kits and Firmware Upgrades

- The default "factory" cal kits are overwritten when new firmware is installed. Your custom cal kits (files with custom filenames) are NOT overwritten. However, the custom cal kits must be imported (click **Import Kit**) into the new firmware.

- All PNA cal kits can only be imported by the current firmware revision and later. They can NOT be imported by PAST firmware revisions. Once a Cal Kit has been imported by a later firmware revision, it cannot be imported by the previous version of firmware from which it originated.

- When a firmware upgrade takes place, ALL cal kits, both factory and custom, that are present on the PNA are saved to a single *.wks file using a unique filename. These files are NOT Excel spreadsheet files. They are opened using the Open button (see below). They can be used as archives of cal kits from previous firmware versions.

**Open** Opens an archive of cal kits from past firmware upgrades and 'Save As' operations.

**Save As** Saves ALL cal kits in the PNA to a *.wks file.

**Restore Defaults** Re-installs the default factory contents of all Agilent cal kits from the PNA hard drive. The factory Agilent cal kits are stored on the PNA hard drive at C:/Program Files/Agilent/Network Analyzer/PnaCalKits/factory.

### Installed Kits

- **Import Kit** Invokes the Import Kit dialog box.
- **Save As** Saves the selected calibration kit and definitions (using .ckt file type).
- **Insert New** Invokes a blank Edit Kit dialog box to create new calibration kit definitions.
- **Print to File** Prints the contents of the selected cal kit to a .prn file.
- **Edit Kit...** Invokes the Edit Kit dialog box to modify selected calibration kit definitions.

**Note:** You CAN NOW modify Data-based Cal Kits. [Learn more.](#)

- **Delete** Deletes selected calibration kit file.
- ^ Selects previous / next calibration kit in list.

For more information see [Creating Custom Calibration Kits using a New Connector Family](#).
**Import Kit dialog box help**

**Note:** No more than 50 cal kits can be imported.

Imports calibration kit definitions from hard disk or other drive that are saved in the various formats. With PNA version 4.0 or later, four kit types can be imported.

**Note: See PNA Cal Kits and Firmware Upgrades**

**Files of type**  Select the file type of your Cal Kit

<table>
<thead>
<tr>
<th>Cal Kit Format</th>
<th>File Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current PNA Series Cal Kit</td>
<td>*.ckt</td>
</tr>
<tr>
<td>Old PNA Series Cal Kit (Version 1)</td>
<td>*.ck1</td>
</tr>
<tr>
<td>8510 Cal Kit</td>
<td>CK_*</td>
</tr>
<tr>
<td>8753, 8752, 8719, 8720, or 8722 Cal Kit</td>
<td>*.ck</td>
</tr>
</tbody>
</table>

**File name**   Navigate and select your cal kit file.

**Open**  Imports the selected file. The kit is added at the end of the list of cal kits.

**Importing Kits other than current PNA Series Kits**

Cal kit files from Agilent "legacy" network analyzers (listed above) may not contain information that the PNA requires. Therefore, the PNA may modify the cal kit name and description, the cal standards, and the cal class assignments in a best effort manner. You may need to correct these modifications after importing your legacy cal kit to meet your specific requirements.

- "Legacy" cal kit files are based on the analyzer test port sex; PNA cal kits are based on the Device Under Test (DUT) connector sex. Therefore, when the kit is imported the standard's label and description are reversed and are noted as F- (female) and M- (male).
When a Coaxial standard is detected in the kit file, a pair of male/female connectors is typically created.

Waveguide standards that are created as connector have no gender.

---

**Edit Kit** dialog box help

**Identification**
- **Kit Number**  Number of the selected calibration kit.
- **Kit Name**  Allows you to change the Name of the selected calibration kit.
- **Kit Description**  Allows you to change the description of the selected calibration kit.

**Connectors**

**Note:** You can NOT use a connector with a new or modified name to perform an ECal User Characterization.

Click the down arrow to change the connector type.

- **Add or Edit**  Invokes the Add or Edit Connector dialog box which allows you to add new connector type to the calibration kit or edit the connector properties.
- **Change Family**  Invokes the Change Connector Family dialog box which allows you to rename the entire connector family name.
**Class Assignments**

Click the down arrow to change the Class Assignment.

**Edit**  Invokes the Modify Calibration Class Assignments dialog box.

**Standards in Kit**

Lists the current standards and descriptions in the cal kit.

**Add...**  Invokes the Add Standard dialog box that allows you to add definitions for a standard.

**Edit...**  Invokes the Edit dialog box that allows you to modify standard definitions for the selected standard: either Open, Short, Load, or Thru.

**Delete**  Deletes selected standard from calibration kit.

---

**Add or Edit Connector** dialog box help

**Identification**

**Note:** You can NOT use a connector with a new or modified name to perform an ECal User Characterization.

**Connector Family**  Allows you to Add or Edit a specific connector name. If you change Connector Family to a unique name, the name and selected Gender is ADDED to the list of connectors in that kit.

**Note:** To change the Connector Family Name of all connectors in the Kit, click Change Family on the previous dialog box.

**Description**  Displays connector type and gender.

**Frequency Range**

**Min**  Allows you to define the lowest frequency at which the standard is used for calibration.

**Max**  Allows you to define the highest frequency at which the standard is used for calibration.
Gender
Allows you to define the connector gender.

Impedance
Allows you to define the impedance of the standard.

Media
Allows you to define the medium (or ‘geometry’) of the connector: COAX or WAVEGUIDE.

Waveguide Cal Kits
If modifying or creating a waveguide cal kit, be sure to make the following settings. You can create a custom waveguide cal kit using an existing factory waveguide Cal kit as a starting point. The factory cal kits already have these settings.

- Frequency Range: Min. frequency = Cutoff frequency.
- Gender: No Gender
- Impedance Z0: 1 ohm
- Media: Waveguide
- Cutoff Frequency enter the low-end cutoff frequency.
- Height/Width Ratio Used to calculate waveguide loss. This value is usually on the data sheet for waveguide devices. For more information see App Note 1287-11.

Other waveguide settings

- If performing an Unguided Cal, change System Impedance to 1 ohm.
- For waveguide, choose TRL (Thru-Reflect-Line) calibration type. These calibration types are more accurate and take fewer steps than SOLT.
Note: You can NOT use a connector with a new or modified name to perform an ECal User Characterization.

Performs a text "Search and Replace" function. Within the description field of each of the standards of the current Cal Kit, it searches for the Previous Connector Name and replaces it with the New Connector Name.

Specify New Connector Name  Allows you to replace the primary connector-family name from the selected kit with the new connector-family name. The PNA allows multiple connector-families per kit.

Previous Connector Name  Displays the primary connector-family name. All occurrences of the previous connector name will be replaced throughout calibration dialog boxes. This includes calibration kit labels and description fields.

Notes:

- String replacement requires an exact match and is case sensitive. For example, "Type N" does not match "type N", and "apc 7" does not match "APC 7".

- Some calibration kits may include connector names that do not match strings within labels or description fields. You may reuse the Change Connector Name dialog to standardize the name within the kit, and then to replace the standard name with the new name.

Example:
Select the 85056A calibration kit. The default connector-family name is "APC 2.4". However, many standard description files are labeled "2.4 mm". You may want to replace the connector family name with a new name and update the standard descriptions to match the new name. For this kit, use a two step procedure.

1. Use the Change Connector Name dialog to replace "APC 2.4" with "2.4 mm".
2. Use the Change Connector Name dialog to replace "2.4 mm" with the new name, "PSC 2.4 mm".

See Also Creating a New Cal Kit from an existing Cal Kit
Modify Calibration Class Assignments dialog box help

Allows you to assign single or multiple standards to Calibration Classes.

There are two ways to get here:

1. Click Calibration
2. Click Advanced Modify Cal Kit..
3. Select the Cal Kit, then click Edit Kit
4. Under Class Assignments, select the Cal Method (SOLT, TRL), then click Edit

You can also get here during a SmartCal Calibration.

1. From the Select DUT Connectors and Cal Kits dialog, check Modify Cal, then click Next.
2. At the Modify Cal dialog, click a Mod Stds button.
3. At the View/Modify Properties Dialog, select the Cal Method (SOLT, TRL), then click View/Modify

To assign a standard to a calibration class:

1. Select the Calibration Kit Class
2. Select the standard from the Unselected Standards field
3. Click the right arrow to move the standard to the Selected Standards field.

Notes:
During an Unguided Cal all of the **Selected Standards** are presented. You then choose which of these standards to measure.

The MATCH standards must be assigned to the FWD MATCH, REV MATCH, and LINE classes. See [TRL calibrations](#) to learn more about TRL standards.

Use MOVE UP and MOVE DOWN to change the ORDER of the standard. The order is used during a [SmartCal](#) to determine overlap priorities when:

- **Multiple standards are valid for a frequency** - standards are presented in the order in which they appear.

- **Using two sets of standards** - modify the order in which standards appear to reflect the configuration of your DUT. For example, for a DUT with a male connector on port 1 and a female connector on port 2, order the devices within the S11 classes (A, B, and C) such that the MALE standards are first in the list. Then order the S22 classes specifying the FEMALE standards as the first in the list.

To Add or Edit standards, click Calibration then, click [Advanced Modify Cal Kit](#).

- See [TRL Class Assignments](#)

- [Learn more about Calibration Classes](#)

**Calibration Class Label**

The label that appears on the Unguided Cal - Measure Mechanical Standards dialog box. For example, the Calibration Class Label "Modified OPEN" would yield the following prompt:

```
Modified OPEN
```

The following selections in this dialog box depend on your Class Assignment selection (**SOLT** or **TRL**) in the [Edit Kit](#) dialog box.

**SOLT ONLY**

**Link FWD TRANS, FWD MATCH, REV TRANS, and REV MATCH** Check to automatically assign the standard definition for FWD TRANS to FWD MATCH, REV MATCH, and REV TRANS. Clear to separately assign FWD MATCH, REV MATCH and REV TRANS classes (**SOLT** calibrations only).

**Expanded Calibration**

The following two check boxes **apply ONLY during Guided Calibrations**. For Unguided Calibration, these check boxes are ignored, including the case where the multiple standards dialog box is presented.

- **Measure all mateable standards in class** Check this box to attain the very highest accuracy possible. For
example, if a cal kit contains several load standards, during the calibration process you will be prompted to measure each of the standards. This could require a significant amount of calibration time. When checked, the "Use expanded math when possible" box is also checked automatically.

**Use expanded math when possible** Some kits contain multiple calibration standards of the same type that together cover a very wide frequency range. (For example: multiple shorts, or a lowband load and a sliding load.) If a calibration requires more than one standard to cover the calibration frequency range, there can be regions of overlapping measurements. When this checkbox is selected, the PNA automatically computes the most accurate measurement in the overlap regions using a "weighted least squares fit" algorithm. This function improves accuracy without slowing the calibration speed.

- Manually select this checkbox only when using a cal kit that contains multiple standards of the same type. (For example: multiple shorts, or a lowband load and a sliding load.)
- The checkbox is cleared by default when a polynomial model is selected from the cal kit menu.
- The checkbox is selected by default when the 85058B or 85058E data-based model is selected from the cal kit menu.

**TRL ONLY**

If TRL is selected as Class Assignment in the Edit Kit dialog box, the following changes appear in this dialog:

**Calibration Kit Class**

- Learn more about TRL standards.
- Isolation calibration is not usually necessary in the PNA.

**LRL line auto characterization**

**Note:** This setting ONLY applies if an LRL Cal Kit is being modified AND Testport Reference Plane is set to Thru Standard AND the TRL Thru class standard and the TRL Line/Match class standard both have the same values for Offset Z0 and Loss. Otherwise, this setting is ignored.

- Check the box to allow the PNA to automatically correct for line loss and dispersion characteristics.
- Clear the box if anomalies appear during a calibrated measurement which may indicate different loss and impedance values for the Line standards.

**Calibration Reference Z0 (TRL only)**

**System Z0** The system impedance is used as the reference impedance. Choose when the desired test port
impedance differs from the impedance of the LINE standard. Also, choose when skin effect impedance correction is desired for coax lines.

**Line Z0** The impedance of the line standard is used as the reference impedance, or center of the Smith Chart. Any reflection from the line standard is assumed to be part of the directivity error.

**Testport Reference Plane (TRL only)**

**Thru Standard** The THRU standard definition is used to establish the measurement reference plane. Select if the THRU standard is zero-length or very short.

**Reflect Standard** The REFLECT standard definition is used to establish the position of the measurement reference plane. Select if the THRU standard is not appropriate AND the delay of the REFLECT standard is well defined.

Also, select if a flush short is used for the REFLECT standard because a flush short provides a more accurate phase reference than a Thru standard.

---

**Add Standard** dialog box help

Allows you to add standards to the calibration kit file.

**OPEN** Adds an open to the calibration kit file.

**SHORT** Adds a short to the calibration kit file.

**LOAD** Adds a load to the calibration kit file.

**THRU** Adds a thru to the calibration kit file.

**DATA BASED STANDARD** Adds a data-based standard to the calibration kit file.

**OK** Invokes a blank Edit Standards: Open, Short, Load, Thru, or Data-Based dialog box.

For more information see Creating Custom Calibration Kits using a New Connector Family.
Edit / Add Standards (Open, Short, Load, Thru, or Data-based)

Note: For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

The boxed areas of the previous graphic applies to all standard types.

The other areas change depending on the type of standard selected.

**Identification**
- **Standard ID**  Number in list of standards
- **Label**  Type of standard.
- **Description**  Description of standard.

**Frequency Range**
- **Min**  Defines the lowest frequency at which the standard is used for calibration.
- **Max**  Defines the highest frequency at which the standard is used for calibration.

**Connector**
Indicates the type and gender (Male, Female, None) of the standard.

**Delay Characteristics**
- **Delay**  Defines the one-way travel time from the calibration plane to the standard in seconds.
- **Z0**  Defines the impedance of the standard.
- **Loss**  Defines energy loss in Ohms, due to skin effect, along a one-way length of coaxial cable.

The following applies to standard types **Open, Short, Load, Thru, and Data-based**
Open Standard

<table>
<thead>
<tr>
<th>Open Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0 89.939 F(e-12)</td>
</tr>
<tr>
<td>C1 2536.7999 F(e-27)/Hz</td>
</tr>
<tr>
<td>C2 264.9901 F(e-36)/Hz²</td>
</tr>
<tr>
<td>C3 13.4 F(e-45)/Hz³⁶</td>
</tr>
</tbody>
</table>

C0, C1, C2, C3 Specifies the fringing capacitance.

Short Standard

<table>
<thead>
<tr>
<th>Short Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>L0 0.7563 H(e-12)</td>
</tr>
<tr>
<td>L1 459.8799 H(e-24)/Hz</td>
</tr>
<tr>
<td>L2 52.429 H(e-33)/Hz²</td>
</tr>
<tr>
<td>L3 1.5846 H(e-42)/Hz³</td>
</tr>
</tbody>
</table>

L0, L1, L2, L3 Specifies the residual inductance.

Load Standard

<table>
<thead>
<tr>
<th>Load Type</th>
<th>Arbitrary Impedance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed Load</td>
<td></td>
</tr>
<tr>
<td>Sliding Load</td>
<td></td>
</tr>
</tbody>
</table>

Complex Impedance

<table>
<thead>
<tr>
<th>Real</th>
<th>Imag</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>0</td>
</tr>
</tbody>
</table>

Delay Characteristics

<table>
<thead>
<tr>
<th>Delay</th>
<th>pSec</th>
<th>Loss</th>
<th>Gohms/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Offset Load Definition

<table>
<thead>
<tr>
<th>First Offset Standard</th>
<th>Second Offset Standard</th>
<th>Load Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>THRU</td>
<td>7-32 Line</td>
<td>ADPTR/LOAD -M-</td>
</tr>
</tbody>
</table>

Allows you to select the type of load.

Load Type

Fixed Load Specifies the load type as Fixed. The fixed load is assumed to be a perfect termination without reflection.

Sliding Load A sliding load is defined by making multiple measurements of the device with the sliding load element positioned at various marked positions of a long transmission line. The transmission line is assumed to have zero reflections and the load element has a finite reflection that can be mathematically removed using a least squares circle fitting method.

A sliding load cal can be very accurate when performed perfectly. It can also be very inaccurate when not using proper technique. For accurate results, closely follow the users manual instructions for the sliding load.

Arbitrary Impedance Specifies the load type to be have an impedance value different from system Z0. An arbitrary impedance device is similar to a fixed load except that the load impedance is NOT perfect. Early firmware releases of the PNA series used a fixed resistance value. A complex terminating impedance has
been added to allow for more accurate modeling of circuit board or on-wafer devices.

The following Complex Impedance settings are available ONLY when Arbitrary Impedance is selected.

- **Real**: The real portion of the impedance value.
- **Imaginary**: The imaginary portion of the impedance value.

**Offset Load**

In Jan 2006, Offset Load definitions were added to TRL and Waveguide Cal Kit files. Using an Offset Load standard results in a more accurate calibration than with a Broadband Load. Therefore, when performing a calibration using one of the modified Cal Kit definitions, you may be prompted to connect more standards than before this change. To revert to using the Broadband Load Standard without offset, do the following:

1. **Click** Calibration, then **Advanced Modify Cal Kit**
2. **Select** the kit, then **click** Edit Kit
3. **Under Class Assignments**, click **Edit**
4. **Select** Calibration Kit Class **S11C** (Loads)
5. **Under Selected Standards**, select **Broadband Load**, then **click** Move Up until the standard is at the top of the list. This will ensure that the Broadband Load is used first.

**About Offset Load**

An offset load is a compound standard consisting of a load element and two known offset elements (transmission lines) of different length. The shorter offset element can be a zero-length (Flush-thru) offset. The load element is defined as a 1-port reflection standard. An offset load standard is used when the response of the offset elements are more precisely known than the response of the load element. This is the case with waveguide. Measurement of an offset load standard consists of two measurements, one with each of the two offset elements terminated by the load element. The frequency range of the offset load standard should be set so that there will be at least a 20 degree separation between the expected response of each measurement.

To specify more than two offset elements, define multiple offset load standards. In cases where more than two offsets are used, the frequency range may be extended as the internal algorithm at each frequency will search through all of the possible combinations of offsets to find the pair with the widest expected separation to use in determining the actual response of the load element.

The following Offset Load settings are available ONLY when Offset Load is selected.

- First Offset Standard
- Second Offset Standard
- Load Standard

**Thru Standard**
Connectors
Defines connector type and gender at both ports.

Data-Based Standard

**Frequency Range**

<table>
<thead>
<tr>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>70000 MHz</td>
</tr>
</tbody>
</table>

**Upload Data From File**

- **Browse...**

**Connectors**
- **One Port Standard**

**File Information**
- Package Name = DATA
- Number of Data Variables = 2
- index = 0: Number of Data Variable Values = 4
- Data Variable Name = S11
- index = 1: Number of Data Variable Values = 4
- Data Variable Name = U11
- Number of IVars = 1
- index = 0: Number of Independent Variable Values = 4
- Independent Variable Name = Freq

**Note:** To learn how to modify data-based standard files, visit [http://na.tm.agilent.com/pna/dbcal.html](http://na.tm.agilent.com/pna/dbcal.html)
The modified file can then be uploaded into the PNA.

**Upload Data From File**
- Click Browse to load data from a file.

**Connectors**
- **One Port Standard** Currently only 1-port standards can be modified.
- **Port 1** Select the type of connector.

**File Information** Information about the standard that is read from the uploaded file.
<table>
<thead>
<tr>
<th>Date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-Jan-2008</td>
<td>Added limit for imported kits</td>
</tr>
<tr>
<td>26-Oct-2007</td>
<td>Added Height/Width for Add connector.</td>
</tr>
<tr>
<td></td>
<td>Moved waveguide settings.</td>
</tr>
<tr>
<td>2-Feb-2007</td>
<td>MX Added UI</td>
</tr>
<tr>
<td>9/12/06</td>
<td>Added link to programming commands</td>
</tr>
</tbody>
</table>
Power Calibration

Source and Receiver Power Calibrations work together to provide very accurate power levels from the source, and very accurate power measurements from the PNA receiver.

- **Source Power Calibration Overview**
- **Supported Power Meters and Sensors**
- **How to perform Source Power Calibration**
- **Copy a Source Power Calibration to other Channels**
- **Saving a Source Power Calibration**
- **Reducing Time to Complete a Source Power Calibration**
- **Receiver Power Calibration**
- **Saving Receiver Cals**

---

**Source Power Calibration Overview**

Perform Source Power Calibration when you need accurate power levels at some point in the measurement path between the PNA test ports. For example, you need to characterize the gain of an amplifier across a frequency range at a specified input power. You would perform a source power cal at the input of the amplifier to ensure the exact power level into the amplifier across the frequency range.

Using a Source Power Cal, you can expect the power at the point of calibration to be within the range of the uncertainty of the power meter and sensor that is used.

**Note:** You may not be allowed to perform a Source Power Cal unless you are logged on to the PNA with an Administrator user account.

**Source Power Calibration:**

- Is independent of measurement type. It corrects the PNA source regardless of which receivers are being used in a measurement. Therefore, it can be used with both ratio or non-ratio measurements.

- Applies ONLY to those measurements on the selected channel that use the test port that was specified as the Source for the calibration. For example, if you specify Channel 1 and Port 1 as the source to be calibrated, only those measurements on channel 1 that use port 1 as the source will be corrected.

- Can be used in conjunction with other measurement calibrations, such as a full 2-port calibration. For highest accuracy, perform the measurement calibration AFTER the source calibration.

- Can be used with Power Sweep type. Source Power Cal will correct the power at all power levels across the power sweep.
Can be used with Port Power Uncoupled.

- Forces sweep mode to Stepped on measurements with source power correction turned ON.
- Beginning with PNA Rev. 7.50, an external source can be calibrated using Source Power Cal.

**Overview of How it works:**

[Click to see the detailed procedure](#)

1. Specify the measurement settings (frequency range, IFBW and so forth).


   **Note:** When using an Agilent 848X power sensor (sensors that do NOT have built-in calibration factors), enter the Cal Factors using the Power Sensor Settings dialog, because the PNA instructs the power meter to NOT use the Cal Factor tables internal to the power meter.

3. Connect a power meter sensor to the point at which you want a known power level. This may be at the input or output of your device, or some other point between the test ports.

4. The PNA source is stepped through the specified frequency range, and power is measured with the power meter. At each data point, the source power is adjusted until the measured power is within your specified accuracy level.

5. When complete, the power meter is preset. The source power calibration can be saved as part of the instrument state.

6. The power meter is removed and the measurement path reconnected.

7. The calibration is automatically applied to the channel. All measurements on that channel using that source port benefit from the source power cal.

**Verify** the source power calibration using the following procedure.

1. Connect the power meter as it was during the source power calibration.

2. Set the PNA to Point Trigger mode.

3. Trigger the PNA across the trace. Read about the behavior of the sweep indicator.

4. At each data point, the power meter should read the corrected power level within the specified tolerance.

**Supported Power Meters and Sensors**

**Power Meters**

The following power meters

All Agilent Power Meters are supported for use in a Source Power Calibration.
See the current list of power meters at: [www.agilent.com/find/powermeters](http://www.agilent.com/find/powermeters)

See a list of [compatible power meter / sensor combinations](http://www.agilent.com/find/powermeters).

The [82357A USB/GPIB Interface](http://www.agilent.com/find/powermeters) can be used to control the power meter.

[LAN connectivity](http://www.agilent.com/find/powermeters) can ONLY be used with the Agilent P-series power meters.

In addition, you can [Create a Custom Power Meter Driver](http://www.agilent.com/find/powermeters) for use with other power meters.

### Power Sensors

You can perform a Source Power Calibration with ALL power sensors that are supported by the above power meters. However, Source Power Calibration, operates slowly with the Agilent E930x and E932x power sensors, as the two calibrations are not optimized for use with those sensors.

Up to two sensors can be used to cover the frequency span of the measurement.

#### USB power sensors

USB power sensors are supported beginning with PNA Rev 7.50.

Only one USB power sensor can be used to cover the entire frequency span.

To select a USB power sensor:

1. Connect the sensor directly to one of the PNA USB ports.
2. From the main Source Power Cal dialog, click Power Meter Config.
3. On the Power Meter Settings dialog, select USB.

See note about [Zeroing USB Power Sensors](http://www.agilent.com/find/powermeters).

---

**How to perform Source Power Calibration**

1. Setup your measurement (sweep type, frequency range, IFBW, and so forth). By default, a Source Power Cal is performed on the source port of the active measurement.

2. Connect coax cable, GPIB cable, and power sensors to the PNA as shown in graphic below.

This image does NOT apply to USB power sensors, which are connected directly to a PNA USB port.
3. Apply power to the power meter and allow 30 minutes warm-up time before beginning calibration.

4. Select **Source Power Cal** as follows:

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td></td>
<td>See programming examples in SCPI and COM</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click <strong>Calibration</strong></td>
</tr>
<tr>
<td></td>
<td>2. then <strong>Source Power Calibration</strong></td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td></td>
<td>See programming examples in SCPI and COM</td>
</tr>
<tr>
<td>1. Press <strong>CAL</strong></td>
<td>1. Click <strong>Response</strong></td>
</tr>
<tr>
<td>2. then <strong>[Power Cal]</strong></td>
<td>2. then <strong>Cal</strong></td>
</tr>
<tr>
<td>3. then <strong>[Source Cal]</strong></td>
<td>3. then <strong>Power Cal</strong></td>
</tr>
<tr>
<td></td>
<td>4. then <strong>Source Cal</strong></td>
</tr>
</tbody>
</table>

5. Complete the Source Power Cal dialog box (below), including **Loss Compensation** and **Power Sensor Settings**, as needed.

**Note:** When using an Agilent 848X power sensor (sensors that do NOT have built-in calibration factors), enter the Cal Factors using the **Power Sensor Settings** dialog, because the PNA instructs the power meter to NOT use the Cal Factor tables internal to the power meter.

6. When complete, click **Take a Cal Sweep** in the Source Power Cal dialog box.
7. Follow the prompts to connect the sensors as required.

8. At this time you can change the Source Port setting and perform a Source Power Cal on a different port.

9. When calibration is finished, click **OK**. Correction is then applied and turned ON for the calibrated ports on the active channel.

10. Remove sensor.

11. **SrcPwrCal** is displayed in the status bar when Source Power Correction is applied to the Active Measurement.

---

**To turn Source Power Correction OFF:**

- On the **Calibration** menu, point to **Power Calibration**, then click **Source Power Correction on/OFF**.
- ONLY correction for the source port of the ACTIVE MEASUREMENT is turned OFF (regardless of port power coupling setting.)

---

**Interpolation**

If the original stimulus settings are changed, Interpolation or EXTRAPOLATION is applied and **SrcPwrCal** is displayed in the status bar. This is different from **measurement calibration interpolation**. For example, if the frequency span is increased, the PNA will extrapolate new correction values rather than turn correction off. This is to protect your test device from being overpowered by the source. If the original settings are restored, then source power calibration returns to full correction.

---

**Source Power Cal** dialog box help

**Note:** Be sure that the frequency range of your power sensor covers the frequency range of your measurement.
This does NOT occur automatically.

**Power**

**Cal Power**  The calculated power (in dBm) at the calibration point. This value is the specified PNA source power plus the Power Offset value.

**Power Offset**  Allows you to specify a gain or loss (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier in the path to your DUT. Following the calibration, the PNA power readouts are adjusted to this value.

**Channel and Port Selection**

**Channel**  Specifies the channel on which to perform the calibration. This setting defaults to the active channel.

**Source Port**  Specifies the source port to be corrected. This setting defaults to the source port for the active measurement.

Beginning with PNA Rev. 7.22, external sources can be controlled from this dialog. Learn more.

**Accuracy**

At each data point, power is measured using the specified Power Meter and adjusted, until the reading is within this Accuracy Tolerance or the Max Number of Readings has been met. The last power reading is plotted on the screen against the Tolerance limit lines.

**Tolerance**  Sets the maximum desired deviation from the specified Cal Power level.

**Max Number of Readings**  Sets the maximum number of readings to take at each data point for iterating the source power.

**Calibration Status**

Allows you to turn Source Power Cal ON | OFF and view Cal data for each port, regardless of the active measurement. This feature allows the Internal Second Source to be calibrated and turned ON | OFF, even when being used as an incidental source in a measurement, such as an LO.

**Calibration ON**  Check to turn Source Power Calibration ON for the specified source port.

The displayed text indicates when interpolation is applied for the calibration.

**Buttons**

**Options**  Invokes the Source Power Cal Options dialog. Label to the left of the button displays the current ‘Options’ setting.

**Power Meter Config**  Invokes the Power Meter Settings dialog box

**Take Cal Sweep**  Begins source power calibration measurement.

**OK**  Applies calibration. This button is disabled until the Take Cal Sweep has been pressed.

**Cancel**  If a sweep is in progress, cancels the sweep. Press again to close the dialog.

**See Also**

- Learn more about Source Power Cal
Learn about External Testsets and Source Power Cal.

**Source Power Calibration Options** dialog box help

Provides options for measurement of the source power.

- **Use power meter only**  Traditional source power calibration using only a power meter to measure the source power at each data point. Most accurate and slowest method.

  **Note:** Because the following two settings use PNA receivers to make power measurements, they do NOT work correctly when a **Frequency Offset** value is being used.

- **Use power meter for first iteration**  When checked, the first reading at each data point uses a power meter to calibrate the reference receiver. Subsequent readings, if necessary to meet your accuracy requirement, are measured using the reference receiver. This technique is much faster than using the power meter with almost no degradation in accuracy.

  **Note:** Do NOT use the "first iteration" feature if there is a component before the power sensor that exhibits non-linear behavior, such as a power amplifier in compression.

- **Use PNA receiver only**  This feature assumes that the receiver to be used has already been calibrated by a source power cal using a power meter, then a **receiver cal**. That receiver can then be used to quickly calibrate other PNA source ports, or on another channel with different stimulus settings.

  This would be useful, for example, if the power level of the measurement was below the sensitivity of the power sensor. Calibrate the PNA receiver using a source power cal that is within the sensitivity of the sensor. Then, use the calibrated receiver to perform a second source power cal at the reduced power level.

  The receiver is specified using **logical receiver notation**.

  - It is best to use the reference receiver for the source port to be calibrated. For example, if calibrating source port 2, specify the "a2" receiver.
To ensure an accurate source power cal, the frequency range over which the receiver was calibrated must be the same or larger than the "receiver only" source power calibration.

All accuracy and settling tolerance and number of reading settings apply just as they do with a power meter reading.

**Produce receiver power calibration of PNA reference receiver** Check to calibrate the appropriate reference receiver to the power level that is measured at the calibration plane. Do this to make very accurate measurements using the calibrated reference receiver. This cal is done in addition to the standard source power cal using any of the methods listed above. At the end of the source power cal measurement sweep, you can optionally save the reference receiver cal to a Cal Set to be recalled at a later time. The Cal is saved when the **OK** button is clicked to close the Source Power Cal dialog.

### Power Meter Settings dialog box help

This dialog appears when you click the **Power Meter Config** button on the main **Source Power Cal dialog**.

**Communication**

- **GPIB / Address** Select GPIB power meter. Then select the address for the power meter. Default is 13. The PNA will search VISA interfaces that are configured in the Agilent IO Libraries on the PNA.

- **USB** PNA scans the USB for connected power sensors. Select a power sensor from the list. Only ONE USB power sensor can be configured to cover the entire frequency range of the calibration.

- **LAN** Specify the Hostname or IP address of the Power Meter. This setting can ONLY be used with the Agilent P-series power meters.

**Sensors** Invokes the [power sensor settings] dialog box.

**Settling**
These Settling settings do not apply when a PNA receiver is the power measurement device. Each power meter reading is "settled" when either:

- two consecutive meter readings are within this **Tolerance** value or
- when the **Max Number of Readings** has been met.

The readings that were taken are averaged together to become the "settled" reading. The settled reading is then compared to the **Accuracy Tolerance requirements** (tolerance and max readings) specified on the Source Power Cal dialog box.

**Tolerance**  When consecutive power meter readings are within this value of each other, then the reading is considered settled.

**Max Number of Readings**  Sets the maximum number of readings the power meter will take to achieve settling.

**Sensor Loss Compensation**

**Use Loss Table**  Select this checkbox to apply loss data to Source Power calibration correction (such as for an adapter on the power sensor).

**Edit Table**  Invokes the **Power Loss Compensation** dialog box.

### Power Loss Compensation dialog box help

Compensates for losses that occur when using an adapter or coupler to connect the power sensor to the measurement port.

**Delete Table Segment**  Deletes row indicated in the field.

**Delete All**  Deletes all data in the table.

**Note: To Add a Row** to the table, click on a row in the table and press the down arrow on either the PNA front panel or keyboard.

- If you enter a single frequency/loss segment, the analyzer applies that value to the entire frequency range.
- You can enter up to 100 segments to achieve greater accuracy.
**Power Sensor Settings dialog box help**

This dialog appears when you click the **Sensors** button on the Power Meter Settings dialog.

**Note:** Be sure that the frequency range of your power sensor covers the frequency range of your measurement. This does NOT occur automatically.

**Sensor A (B)** Displays one of the following messages depending on type of sensor.

- **Not connected** The PNA is not detecting a power sensor.
- **Cal factors are contained within this sensor** The PNA detects an Agilent E-Series power sensor. Reference Cal Factor and Cal Factor data are loaded automatically.
- **Sensor Data** Allows entry for power sensor data:
  - **Reference Cal Factor** Specifies the sensor's Reference Cal Factor.
  - **Cal Factor Table** Specifies the frequency and corresponding Cal Factor for the sensor.
  - **Delete Cal Factor** Deletes the indicated row in the table.
  - **Delete All** Deletes all data in the table.
  - **To Add a Row** to the table, click on a row in the table and press the down arrow on either the PNA front panel or keyboard. A row is added to the bottom of the table. The table is automatically sorted by frequency when OK is pressed.
  - **Use this sensor only** Check this box to use this sensor over the entire frequency span of the measurement, even if two sensors are connected to power meter. Clear to allow entry of minimum and maximum frequencies for the sensor.
  - **Minimum Frequency** Specifies the minimum frequency range for the sensor when using dual sensors.
  - **Maximum Frequency** Specifies the maximum frequency range for the sensor when using dual sensors.
  - **Perform Sensor Zeroing and Calibration** Zero and calibrate the power sensor before taking data.

**Note:** There is no calibration needed in U2000 Series **USB power sensors**. Zeroing those sensors does NOT require disconnecting them from the source port or DUT except, for highest accuracy, when the power level is...
below -30 dBm. For more information, please read the USB power sensor documentation.

Copy a Source Power Calibration to other Channels

A macro application is now available that copies a Source Power Calibration to other channels. Once downloaded and installed on a PNA, the macro is automatically configured up. To learn more, click Help on the application main dialog. Get the application from http://na.tm.agilent.com/pna/apps/applications.htm.

Saving a Source Power Calibration

Because Source Power Cal calibrates source hardware, the calibration data is saved as part of the Instrument State, in either a .sta file or a .cst file. This correction is applied to all measurements on the channel that uses the calibrated source. See Save Instrument State.

Reducing Time to Complete a Source Power Calibration

The time required to perform a Source Power Calibration depends on source power, number of points, and number of readings taken. You can reduce this measurement time with the following methods:

- **Reduce number of points before calibration.** You can reduce the number of points before the measurement, then return the number of points to its original value after calibration is complete and correction is ON. The analyzer will perform a linear interpolation, although with some loss in accuracy.

- **Use an Agilent E-Series sensor.** You can obtain 40+ readings per second over GPIB with this type of sensor on the PNA.

- **Increase power to the sensor.** Lower power may have longer settling time with some sensors.

- Check Use Reference Receiver for Iteration.

Receiver Power Calibration

Receiver power calibration mathematically removes frequency response errors in the specified PNA receiver, and adjusts readings to the same, or a value offset from, the source power calibration level. It is the same as doing a Response Cal or Data / Memory, (Normalization) but with the data shifted to the Cal Power value.

Use Receiver Power Calibration to make very accurate absolute power (amplitude) measurements.

Receiver Power Calibration:

- Is ONLY allowed when making absolute power (unratioed) measurements.

- Is most accurate when a source power calibration was performed first.

- Applies to all unratioed measurements in the active channel using that receiver.

- Can be saved in a Cal Set and later reapplied to a like measurement.

Interpolation
Like other calibration types, if the original stimulus settings are narrowed, interpolation is applied and \( C^* \text{ Rcvr Pwr} \) is displayed in the status bar. If the original stimulus settings are made wider, the PNA will turn Receiver Power Correction OFF.

If the original settings are restored, then receiver power calibration returns to full correction.

**How to perform a Receiver Power Calibration**

1. Perform a [Source Power Calibration](#).
2. Set the active measurement to unratioed. [Learn How](#).
3. Connect a THRU line from the source port to the receiver port.
   - When performing a receiver power cal on a reference receiver, no connection is necessary as the receiver is internally connected to the source.
   - When the receiver port and the source port are the same (receiver A, source port 1), then connect an open or short to get maximum power to the receiver. This practice is not recommended. It is best to use different ports for the source and receiver.
4. Ensure correction for Source Power Calibration is ON as indicated by [Src Pwr Cal](#) or [Src Pwr Cal*](#) in the status bar.
5. Start the [Calibration Wizard](#)

### Using front-panel HARDKEY [softkey] buttons

**For N5230A and E836x A/B models**

1. **Navigate using MENU/DIALOG**

**For PNA-X and 'C' models**

1. Press [CAL]
2. then [Power Cal]
3. then [Receiver Cal]

### Using a mouse with PNA Menus

**Programming Commands**

1. Click [Calibration]
2. then [Cal Set]
3. Click [Response]
4. then [Cal]
5. Power Cal
6. then [Receiver Cal]

Complete the following dialog box, then click [Next].
Select Calibration Type for Unratioed Measurement dialog box help

**Cal Type Selection**  Select **Receiver Power**

**Receiver Power Configuration**

**Cal Power**  Specifies the power level to be displayed on the measurement when complete. (Source Port Power + Power Offset).

**Source Port Power**  Test port Power set for the measurement.  Learn how to change Test Port Power

**Power Offset**  Allows you to specify a gain or loss (in dB) to account for components you connect between the source and the reference plane of your measurement AFTER a source power cal has been performed. Following the calibration, the PNA power readouts are adjusted to the Cal Power value.

**Next**  Click to continue the Calibration Wizard.

**Notes:**

- When Receiver Power Cal is finished, C RcvrPwr is displayed in the status bar and correction data is applied to subsequent sweeps.
- To turn correction **OFF**, click **Calibration**, point to **Power Calibration**, then set **Receiver Power Correction** to **OFF**.

Learn more about Receiver Power Cal (scroll up).

---

**Saving a Receiver Power Calibration**

Beginning with PNA Revision 5.0, Receiver Power Cal is saved to a **Cal Register** and optionally to a **User Cal Set**. It can be applied to measurements in the same way as other Cal Types. Previously, Receiver Power Cal data was saved as part of an Instrument State and was only applied to the measurement on which it was performed.

Learn more about Saving PNA files types.

---

Last modified:
<table>
<thead>
<tr>
<th>Date</th>
<th>Change Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>21-Feb-2008</td>
<td>Added 848x note</td>
</tr>
<tr>
<td>4-Jan-2008</td>
<td>Added Cal note for USB sensors</td>
</tr>
<tr>
<td>30-Oct-2007</td>
<td>Added link to supported Power meters/ sensors</td>
</tr>
<tr>
<td>20-Jul-2007</td>
<td>Added USB / LAN support and Apply macro</td>
</tr>
<tr>
<td>21-January 21, 2007</td>
<td>MX Added UI</td>
</tr>
<tr>
<td>14 Sept-2006</td>
<td>MQ Added Receiver-only SPC.</td>
</tr>
</tbody>
</table>
Fixture Simulator

The following features allow you to mathematically add (embed) or remove (de-embed) circuits to, or from, your PNA measurements. The mathematical models are applied to specific ports for all measurements on the channel.

See Also

- "De-embedding and Embedding S-Parameter Networks Using a Vector Network Analyzer" App note, for more conceptual information on Fixture Simulation.
- Characterize Adaptor Macro can be used to create S2P files from Cal Sets.
- To Embed or De-embed? and the associated procedures

Order of Fixture Operations

- The fixturing operations are applied to the measurement results in the following order. This order can NOT be changed.
- In the PNA data processing chain, the Fixture Simulator functions occur at the same time as the Apply Error Terms block.

First, the following Single-ended measurement functions are processed in this order:

1. Port Extensions
2. 2-Port De-embedding
3. Port Z (Impedance) Conversion
4. Port Matching Circuit Embedding
5. 4-Port Network (single-ended) Embed/De-embed

Then, Balanced measurement functions are processed in this order:

6. Balanced Conversion
7. Differential / Common Mode Port Z Conversion
8. Differential Matching Circuit Embedding
Note: Port Impedance (Z) conversion uses values in the following prioritized order:

1. Balanced (Differential or Common Mode) - if enabled, these values are always used.
2. Single Port Impedance - if enabled, this value is used if Balanced is not enabled.
3. System Impedance - if neither balanced or single port is enabled, this value is used.

See an example of how these functions can be used to de-embed unwanted effects of a test fixture, and then mathematically embed the DUT in the circuit in which it is used.

How to select Fixturing Simulator

About Fixturing ON/off

BOTH of the following must occur to turn a fixturing selection ON.

EITHER ONE will turn a fixturing selection OFF.

1. Check Fixturing ON/off
   Port Extensions is NOT affected by Fixturing ON/off.
2. Check Enable on the individual fixturing selection dialog box.

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>For N5230A and E836xA/B models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Calibration</td>
</tr>
<tr>
<td></td>
<td>2. then Fixturing Selections</td>
</tr>
<tr>
<td>For PNA-X and 'C' models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press CAL</td>
<td>1. Click Cal</td>
</tr>
<tr>
<td>2. then [More]</td>
<td>2. then More</td>
</tr>
<tr>
<td>3. then [Fixtures]</td>
<td>3. then Fixtures</td>
</tr>
</tbody>
</table>
**Port Matching** dialog box help

This function specifies a circuit to embed (add) to the measurement results. See Order of Fixture Operations.

**Enable Port Matching**  Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

**Port** - Select Port in which to apply simulation.

**Circuit Model for Matching** - Choose one of the following that best emulates your fixture at the selected PNA port:

1. **Series L - Shunt C**
   - PNA: \( L \), \( R \), \( G \), \( C \)
   - DUT: \( G \), \( C \)

2. **Shunt C - Series L**
   - PNA: \( G \), \( C \)
   - DUT: \( L \), \( R \)

3. **Shunt L - Series C**
   - PNA: \( L \), \( R \)
   - DUT: \( G \), \( C \)
User Defined (S2P File)  Load a file that is specified with User S2P File button.

None  Use no circuit model.

User S2P File  Click to specify an S2P file of the circuit model to embed at the selected port. If the normalized impedance value in a recalled User .S2P file is different from the port reference impedance setting of the PNA, the PNA setting is used.  Characterize Adaptor Macro can be used to create S2P files from Cal Sets.

Circuit Values

- **Capacitance (C)**, **Inductance (L)**, **Resistance (R)**, **Conductance (G)**  Values for the specific components of the circuit type that models your fixture.

Reset  Restores the default values.
This function **removes** the effects of a test fixture from the measurement results.

The de-embedding operation recalls an .s2p file (Touchstone format) for a 2-port test fixture. The file includes the electrical characteristics of a supplemental fixture or device. The file can be in any standard format (real-imaginary, magnitude-angle, dB-angle) and can represent any 2-port test fixture.

In the following image, the 2-port fixture would be either Fixture A OR Fixture B. To de-embed both, perform this operation twice.

**Note:** In all cases:

- Port 1 of the fixture is assumed to be connected to the PNA.
- Port 2 of the fixture is assumed to be connected to the DUT.

**Enable De-embedding** Check to apply the settings to the measurement results. Must also enable **Fixturing ON/off**.

**Port** - The PNA port to which the recalled de-embedding file is applied.

From the drop-down menu, select **User S2P**.

**User S2P File** Click to specify an existing .S2P file. If the normalized impedance value in a recalled User .S2P file is different from the port reference impedance setting of the PNA, the PNA setting is used. **Characterize Adaptor Macro** can be used to create S2P files from Cal Sets.
**Port Z (Impedance) Conversion** dialog box help

This function corrects the measurement and displays the results as if the measurement had been made into the specified impedance value. However, the physical port termination is still approximately 50 ohms.

The specified impedance value is applied to all of the measurements on ONLY the active channel.

See [Order of Fixture Operations](#).

**Enable Port Z Conversion**  Check to apply the settings to the measurement results. Must also enable [Fixturing ON/off](#).

**Z**  Resistance part of the desired reference impedance for the specified port and channel.

**Close**  Applies the entries and closes the dialog box

See note about Port Impedance priority.

---

### 4-Port Embed/De-embed dialog box help

This function specifies a single-ended 4-port circuit (*.S4P file) to embed (add) or de-embed (remove) from the measurement results. Computation takes place BEFORE Balanced conversion. See [Order of Fixture Operations](#).

There is a single normalized impedance value for each port in the *.S4P file. This impedance value must match the impedance of the previous Port Z setting, or the PNA port impedance.

The PNA will interpolate if the number of data points that are read is different from the current PNA setting.

**Enable 4-Port Embed/De-embed**  Check to apply the settings to the measurement results. Must also enable [Fixturing ON/off](#).

**Topology:**  Select a DUT topology.

Refer to the images on 4-port embed/De-embed dialog box.

- **A** - 2 PNA/DUT Ports
- **B** - 3 PNA/DUT Ports
C - 4 PNA/DUT Ports

Topology configurations that are not addressed with standard images in dialog box:

1. If you have a 4-port DUT; 4-port network on one side; None on the other side.

   Specify Topology C.

   Use 4-port Network on one side.

   Use 4-port Network on the other side; set to None.

2. If you have a 3-port DUT and networks as follows:

   Specify Topology B.

   Use 4-port Network1 on one side.

   Use 2-port network on the other side.

NA Ports - Select the PNA Port that is connected to each circuit port.

Note: The *.S4P file always assumes that:
Network ports 1 and 2 are connected to the PNA
Network ports 3 and 4 are connected to the DUT

None, Embed, De-embed  For Network1 and Network2, select:

- **None** - The same as disabling.
- **Embed** - Add the specified network circuit to the measurement results.
- **De-embed** - Remove the specified network circuit to the measurement results.

Browse  For both Network1 and Network2, navigate to find the *.S4P file to embed or de-embed.

OK  Applies the changes and closes the dialog box.

Cancel  Does NOT apply the changes and closes the dialog box.
**Differential Impedance Conversion** dialog box help

This function sets the Differential impedance value for each balanced port.

The default value for R: is the SUM of the impedance values for both ports that make the logical port. If **Port Z Conversion** is not enabled, then **System Z0** values for both ports are summed.

See Order of Fixture Operations.

**Enable Differential Z Conversion** Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

**Logical Port** Select the logical (balanced) port to receive impedance value. To see logical port numbers, see the measurement topology.

**R** Real part of the impedance value.

**jX** Imaginary part of the impedance value.

**Close** Closes the dialog box.

See note about Port Impedance priority.
This function sets Common Mode Impedance value for each balanced port.
The default value for R: is calculated as follows.

\[
\frac{Z1 \times Z2}{Z1 + Z2}
\]

Where ports 1 and 2 comprise the logical port:

- Z1 = the Port Impedance values for port 1
- Z2 = the Port Impedance values for port 2

If Port Z Conversion is not enabled, then System Z0 values for port 1 and 2 are used in the calculation.

See Order of Fixture Operations.

Enable Common Mode Z Conversion Check to apply the settings to the measurement results. Must also enable Fixturing ON/off.

Logical Port Select the logical (balanced) port to receive impedance value. To see logical port numbers, see the measurement topology.

R Real part of the impedance value.

jX Imaginary part of the impedance value.

Close Closes the dialog box.

See note about Port Impedance priority.
Differential Port Matching dialog box help

This function allows the embedding of a differential matching circuit at a balanced port.  
See Order of Fixture Operations.

Enable Differential Port Matching  Check to embed the selected matching circuit to the measurement results. Must also enable Fixturing ON/off.

Logical Port  Choose Logical DUT port to receive the selected matching circuit.  To see logical port numbers,  see the measurement topology. 

Select Circuit  Select a matching circuit.  Choose from:

- **Shunt L - Shunt C**  Predefined circuit.

![Shunt L - Shunt C Circuit](image)

Circuit Values  Choose from:

- **C** Capacitance value
- **G** Conductance value
- **L** Inductance value
- **R** Resistance value

- **User defined**  Select an *.S2P file that represents the matching circuit.  Then click Browse to navigate to the *.S2P file.

**Note**: For the *.S2P file:

Port 1 of the circuit is assumed to be connected to the PNA  
Port 2 of the circuit is assumed to be connected to the DUT.

- **None**  No embedded circuit on selected port.

Close  Closes the dialog box.

Fixture Simulator Example

The following example shows a DUT and the matching circuit with which the DUT will be used in its intended application. When the DUT is tested in a high-volume manufacturing environment, multiple test fixtures are often required. The most accurate way to test the DUT and ensure measurement consistency between the different test
fixtures is to use a simple, repeatable, test fixture without the actual matching elements.

To get the desired performance data, the parasitic effects of the fixture must first be removed (de-embedded) from the measured data. Then a perfect "virtual" matching circuit must be simulated and added mathematically (embedded) to the corrected, measured data. The result is an accurate display of the DUT as though it was actually tested with a physical matching circuit, but without the uncertainties of using real components.

Test Device and the circuit in which it will be used.

Circuit Simulation

Port Z Conversion

Port Extensions/De-embed

Balanced Conversion

Port Matching

Diff Port Match

Balanced Z Conversion

This diagram does NOT refer to the order in which operations are performed.

See Order of Fixture Operations.

1. Create a balanced measurement using single-ended to balanced (SE-Bal) topology. Include all relevant measurement settings (IFBW, number of points, and so forth). Once the measurement is created and calibrated, the measurement parameter can be easily changed. For example, Sdd22 to Sds21.

2. Calibrate the measurement at the point where the simple test fixture is connected to the PNA. Use accurate calibration standards and definitions.

3. Remove the effects of the three uncalibrated transmission lines of the simple test fixture. This can be done in several different methods. The easiest is to use manual or automatic Port Extensions to move the calibration reference plane to the DUT. This removes the electrical length and loss of the fixture’s transmission lines, but does not account for fixture mismatch. Another method is to de-embed previously-created *.S2p files of the 3
transmission lines. The files can be created using external ADS modeling software. Another alternative is to create the *.S2P files by independently measuring all 3 ports of the test fixture and saving the results of each to an S2P file.

4. With the test fixture connected to the PNA and a DUT inserted, the measurement results now appear as though calibration was performed at the connections to the DUT, and the device was measured in a 50-ohm single-ended test environment. The following steps will cause the results to reflect the performance of the device as though the device is embedded in the circuit in which it will be used.

5. Port 1 of the device is a single-ended port and sees a source impedance the same as the PNA system impedance, so no change is required. However, if Rs were a value other than 50 ohms, Port 1 Impedance Conversion would be used to simulate the different impedance.

6. Port Matching is used to simulate L1 inductance. Select any of the Shunt L circuits to embed (add) to the measurement results. Enter the value of L and R. The C and G values can be entered as 0 (zero).

7. Port Matching is used to simulate C1 and C2 capacitance. For both port 2 and port 3, select any of the Series C circuits to embed (add) to the measurement results. Enter the value of C and G. The L and R values can be entered as 0 (zero).

8. Balanced Conversion mathematically simulates the measurement in balanced mode.

9. Differential Port Matching is used to simulate L2 inductance. Select Shunt L- Shunt C and enter the inductance / resistance value. The C and G values can be entered as 0 (zero).

10. Finally, Differential Z Conversion is used to simulate a circuit termination of 200 ohms. If you are making Common Mode measurements, specify Common Mode Z Conversion.

Last modified:

March 10, 2008   MX Added UI
Sept 12, 2006   Added link to programming commands
Port Extensions

Port extensions allow you to electrically move the measurement reference plane after you have performed a calibration. The following two scenarios show how port extensions can be useful.

1. You have already performed a calibration, and then decide that you need to add a length of transmission line in the measurement configuration. Use port extensions to "tell" the analyzer you have added the length to a specific port.

2. You are unable to perform a calibration directly at your device because it is in a test fixture. Use port extensions to compensate for the time delay (phase shift), and optionally the loss, caused by the added transmission line of the fixture.

See Also

Fixture Compensation features
Phase Accuracy
Comparing the PNA Delay Functions

How to launch the Port Extensions toolbar

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Calibration</td>
</tr>
<tr>
<td></td>
<td>2. then Port Extension Toolbar</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press CAL</td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then [Port Ext Tool]</td>
<td>2. then Cal</td>
</tr>
<tr>
<td></td>
<td>3. then Port Extension Tool</td>
</tr>
</tbody>
</table>

Port Extensions toolbar help

Port extensions settings affect all measurements on the active channel that are associated with a particular port.
Learn about Port Extensions (scroll up)

- If you know the electrical length of additional transmission line, enter the value directly to the Delay setting.

- If you know the physical length of additional transmission line, increase the Delay setting until the physical length setting (directly above Delay) is achieved.

- If you do not know the electrical or physical length of additional transmission line, you must be able to connect an OPEN or SHORT to the new reference plane at the point of the DUT. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane. Port Extensions can then be added manually (as follows), or by using Automatic Port Extensions.

Manual Port Extensions Procedure

1. Select a calibrated S11 measurement.

2. Select Phase format.

3. With an OPEN or SHORT at the calibration reference plane, verify that the phase across the frequency span is at or near zero.

4. Connect the added transmission line or fixture and attach an OPEN or SHORT in place of the DUT. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane. On the Port Extension toolbar, increase Delay until the phase response is flat across the frequency span of interest.

5. If you know the loss of the additional transmission line, enter the Loss Compensation values using either one or two data points.

Note: Most OPEN and SHORT standards have delay. Therefore, adjusting delay with this method results in a delay equal to two times the delay of the OPEN or SHORT.

Port Extensions Settings

Port Extension  Turns ON and OFF port extensions on all ports.

Port  Select a PNA port for delay and loss values. Port Extensions settings affect ALL measurements on the active channel that are associated with a particular port.

Delay  The amount of port extension delay in time. To compensate for delay in additional transmission line, enter a positive value.

Loss Compensation

The following settings, along with Loss at DC, allows the entire frequency span to be corrected for loss using a curved-fit algorithm.

To compensate for loss in additional transmission line, enter a positive value which causes the trace to shift in the positive (up) direction.

Loss1  Loss in dB at Freq1.

Use1  Check calculate and apply port extension Loss1 @Freq1 values. Also, check if using Loss at DC value.
**Loss2** Loss in dB at **Freq2**.

**Use2** ONLY available if Use1 is checked. Calculate and apply port extension Loss2 @Freq2 values.

Loss is calculated for each frequency data point \( f \) as follows.

If Use1 is checked and NOT Use2 then:

\[ \text{Loss}(f) = \text{Loss1} \times \left( \frac{f}{\text{Freq1}} \right)^{0.5} \]

If Use1 AND Use2 are checked, then:

\[ \text{Loss}(f) = \text{Loss1} \times \left( \frac{f}{\text{Freq1}} \right)^n \]

Where:

\[ n = \log_{10} \left[ \frac{\text{abs} (\text{Loss1} / \text{Loss2})}{\log_{10} \left( \frac{\text{Freq1}}{\text{Freq2}} \right)} \right] \]

Note: abs = absolute value

**More** Invokes the More Port Extensions Settings dialog box.

**Auto Ext.** Invokes the Automatic Port Extensions dialog box

**Note:** Individual receiver port extensions (A,B, and so forth) can no longer be set. (Sept. 2004)

Learn about Port Extensions (scroll up)
Automatic Port Extension AUTOMATICALLY performs the same operation as Manual Port Extension. By connecting a SHORT or OPEN, the reference plane is automatically moved to the point at which the standard is connected. In addition, Automatic Port Extension will optionally measure and compensate for the loss of the additional transmission line.

**Auto Port Extensions Procedure**

1. Connect the added transmission line or fixture. Attach an OPEN or SHORT to all affected ports at the new reference plane. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane.

2. On the Port Extension toolbar, click **Auto Port Ext**. Click **Show Configuration** to make additional settings.

3. Click **Measure** to perform the port extension calculations. The resulting delay and loss settings are entered into the port extension toolbar. These settings are saved with Instrument Save or you can manually record the values and enter them again when required.

**Settings**

**Measure either OPEN, SHORT, or both** Press a button to make the measurement of the reflection standard. Measure either OPEN or SHORT depending on which is most convenient. An ideal OPEN and SHORT, with zero loss and delay, is assumed. Therefore, accuracy is most affected by the quality of the standard. In most cases, removing the DUT will leave a suitable OPEN at the new reference plane. When measuring both OPEN and SHORT standards, the average of the two is used and will slightly improve accuracy.

**Selected Ports** Indicates the ports that currently have automatic port extension enabled. By default, ALL PNA ports are enabled. To disable a port, see **Measure on Port Number** below.

**Note:** Port Extensions settings affect ALL measurements on the active channel that are associated with a
particular port.

**Show/Hide Configuration**  Press to either show or hide the following configuration settings in the dialog box.

**Measure on Port Number**
Select port number to enable or disable automatic port extension.

**Enable**  Check to enable the specified port. All enabled ports will have their reference plane automatically adjusted after performing Automatic Port Extension.

**Include Loss**  Check to automatically measure the loss in the additional transmission line and apply compensation. To calculate loss compensation, frequencies at 1/4 and 3/4 through the frequency range are usually used as Freq1 and Freq2 values. See [Learn more about Loss Compensation](#).

**Adjust for Mismatch**  Only available when **Include Loss** is checked. During the measurement of the OPEN or SHORT standard, mismatch could cause ripple in the magnitude (loss) response. The Loss compensation curved-fit algorithm allows half of the ripple to be positive and half negative. When measuring low-loss devices, it is possible that some magnitude responses could become slightly positive, indicating gain rather than loss.

- **Check**  - Offsets the trace to cause all of the data points to be at or below zero.
- **Clear**  - Most accurate application of the curve-fit calculation, but allows positive responses.

**Prompt for Each Standard**  Check to invoke a prompt when the Measure OPEN or SHORT button is pressed. The prompt will indicate which standard to connect to which port.

**Method**
Select the span of data points which will be used to determine correction values for phase and loss (optional). If a portion of the current frequency span does not have flat or linear response, you can eliminate this portion from the calculations by using a reduced User Span.

To calculate loss compensation, Current Span and User Span methods usually use frequencies at 1/4 and 3/4 through the frequency range as Freq1 and Freq2 values. See [Loss Compensation](#) to learn more about how loss is calculated.

- **Current Span**  Use the entire frequency span to determine phase and loss values.
- **Active Marker**  Use only the frequency at the active marker, and one data point higher in frequency, to calculate phase and loss values. If a marker is not present, one will be created in the center of the frequency span.
- **User Span**  Use the following User Span settings to determine phase and loss values.

**User Span**

- **Start**  Enter start frequency of the user span.
- **Stop**  Enter stop frequency of the user span.

Learn about [Port Extensions](#) (scroll up).

See also [Comparing the PNA Delay Functions](#)
Delta Match Calibration

A TRL Cal, QSOLT, or Unknown Thru Cal requires a reference receiver for each test port. The 4-port PNA-L model does NOT have a reference receiver for each test port.

A Delta Match Calibration can be thought of as a software method which provides a reference receiver for each test port when not otherwise available in the hardware. The Delta Match Calibration measures the source match and load match of the PNA test ports, and then calculates the differences, or "delta", of the two match terms. The results are then used to correct subsequent TRL, QSOLT, or Unknown Thru calibrations.

There are several ways to acquire the Delta Match Calibration:

1. **From an existing User Cal Set** that meets the following Delta Match criteria: (Not allowed for use with external test sets.)

   - Must have been performed using ECal or as a guided mechanical Cal (not Unguided).
   - Must have the same start frequency, stop frequency, and number of points as the channel being calibrated.
   - Must calibrate the ports that require the delta match terms.

2. **From a Global Delta Match Calibration**.

3. **From a 'Self Delta Match'** when other portions of the calibration fully characterize all ports using SOLT with Defined Thru or Flush Thru. For example, when calibrating all four ports of a PNA-L, perform a SOLT between ports 1 and 2, and also between ports 3 and 4, then Unknown Thru could be used between any combination of the remaining ports. This is allowed with an external test set.

**Which to use?** A Self Delta Match Cal will always be used when possible. Otherwise, the Cal Wizard will use a Global Delta Match Cal when available unless you select Choose Delta Match.

Global Delta Match Cal

A Global Delta Match Cal is an "all-inclusive" calibration that can be applied whenever the delta match terms are required.

A Global Delta Match Cal differs from a standard SOLT Cal in the following ways:

- It is always performed using a Flush Thru, a Known Thru, or an insertable ECal module. You can NOT use an Unknown Thru in the calibration process.

- Only two Thru connections are required to characterize the delta match terms on a 4-port PNA. This is less than the minimum number of Thrus of a standard 4-port Cal.

- Upon completion, the Global Delta Match Cal is stored as a special type of Cal Set and should be used ONLY as a Delta Match Cal. It provides Delta Match error terms, but does NOT provide all of the standard error correction terms.

- To attain the highest accuracy, the following settings are automatically used to perform a Global Delta Match Cal. When applied, it will likely be interpolated.
Performed over the entire frequency range of the PNA.

Uses very dense data points, particularly at low frequencies.

Uses 100 Hz IF Bandwidth.

**Note:** For highest accuracy, perform Global Delta Match Cal using an insertable ECal module and select Flush-thru as the Calibration Thru method.

### How to perform a Global Delta Match Cal

These selections will only be available if the PNA hardware requires a Delta Match Calibration.

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Calibration</td>
</tr>
<tr>
<td></td>
<td>2. then Global Delta Match Cal</td>
</tr>
<tr>
<td><strong>For PNA-X - Not required</strong></td>
<td>Programming Commands</td>
</tr>
<tr>
<td><strong>For PNA 'C' Models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press CAL</td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then [Start Cal]</td>
<td>2. then Cal</td>
</tr>
<tr>
<td>3. then [Global Delta Match]</td>
<td>3. then Start Cal</td>
</tr>
<tr>
<td></td>
<td>4. then Global Delta Match</td>
</tr>
</tbody>
</table>

![Delta Match Calibration: Select DUT Connector and Cal Kit](image)

561
Delta Match Calibration. Select DUT Connectors and Cal Kit dialog box help

- Only one Cal Kit is specified and necessary to perform a Delta Match Cal. However, ALL of the PNA test ports are calibrated in a Delta Match Cal.

- You must configure ALL test ports to terminate in the specified connector / gender using the necessary adapters. The errors from adapters are removed during calibration, but the Thru connections must be made as specified.

- If you select an ECal module that does NOT cover the entire frequency range of the PNA, your selection will change to a different Cal Kit. The Global Delta Match Cal covers the entire frequency range of the PNA. Your selected Cal Kit or ECal module must also cover the frequency range of the PNA.

Guided Calibration Steps dialog box help

Click Measure for each standard.

When all standards have been measured, click Done to complete the measurement steps.

Delta Match Calibration Complete dialog box help

Click Finish to store the Global Delta Match Calibration as a special type of Cal Set.

By default, it will be used when a Delta Match Calibration is required.

It should ONLY be used as a Delta Match Cal. It does NOT provide all of the standard error correction terms.
9-Nov-2007    Edits for requirements
23-Feb-2007    Modified requirements for multiport
9/12/06        Added link to programming commands
Markers

Markers provide a numerical readout of measured data, a search capability for specific values, and can change stimulus settings. There are 9 regular markers and one Reference marker (used with Delta markers) available per trace. This topic discusses all aspects of markers.

Note: Marker Readout can be turned ON / OFF and customized from the View/Display menu. See Marker Readout

- Creating and Moving Markers
- Delta Markers
- Searching with Markers
- Marker Functions (Change Instrument Settings)
- Advanced Marker Settings
- Marker Table

Other Analyze Data topics

How to Create Markers

Using front-panel HARDKEY [softkey] buttons

<table>
<thead>
<tr>
<th>For N5230A and E836xA/B models</th>
<th>For PNA-X and 'C' models</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press MARKER The first button press creates marker 1.</td>
<td>1. Press MARKER The first button press creates marker 1.</td>
</tr>
<tr>
<td>2. To create more markers, use the Active Entry toolbar</td>
<td>2. then [Marker n]</td>
</tr>
<tr>
<td>3. Or use the Marker toolbar</td>
<td>3. select a marker number</td>
</tr>
</tbody>
</table>

Using a mouse with PNA Menus

<table>
<thead>
<tr>
<th>Programming Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Click Marker</td>
</tr>
<tr>
<td>2. then Marker</td>
</tr>
</tbody>
</table>

Programming Commands

Moving a Marker
To move a marker, make the marker active by selecting its number in any of the previous 3 methods. The active marker appears on the analyzer display as . All of the other markers are inactive and are represented on the analyzer display as . Then change the stimulus value using any of the following methods:

- Type a value.
- Scroll to a stimulus value using the up / down arrows. The resolution can not be changed.
- Click the stimulus box, then use the front-panel knob.

Click and Drag Markers - PNA-X ONLY

Markers can also be moved across a trace using a finger (touchscreen) or by left-clicking and holding a marker symbol. Then drag the marker to any point on the trace. This feature is NOT allowed in Smith Chart or Polar display formats or with a Fixed Marker type.

![Marker dialog box help](image)

**Marker** dialog box help

**Marker**  Specifies the current (active) marker number.

**Stimulus**  Specifies the X-axis value of the active marker. To change stimulus value, type a value, use the up and down arrows, or click in the text box and use the front-panel knob.

**On**  Check to display the marker and corresponding data on the screen.

**Delta Marker**  Check to make the active marker display data that is relative to the reference (R) marker. There is only one reference marker per trace. All nine other markers can be regular markers or delta markers. When a delta marker is created, if not already displayed, the reference marker is displayed automatically.

A delta marker can be activated from the Marker dialog box or the Marker Toolbar.

**Advanced...**  Invokes the Advanced Markers dialog box.

**All Off**  Switches OFF all markers on the active trace.

**Searching with Markers**

You can use markers to search measurement data for specific criteria.

If there is no valid data match for any of the search types, the marker will not move from its current position.
How to Search with Markers

Using front-panel HARDKEY [softkey] buttons

For N5230A and E836xA/B models

1. Press MARKER SEARCH
2. Only Max, Min, Left Peak, and Right Peak search types are available from Active Entry Keys

For PNA-X and 'C' models

1. Press SEARCH

Using a mouse with PNA Menus

1. Click Marker
2. then Marker Search

Marker Search dialog box help

**Marker** Specifies the marker that you are defining.

**Search Domain** Defines the area where the marker can move or search. For full span, the marker searches for specified values within the full measurement span. For user span, the marker searches for specified values within a measurement span that you define. [Learn more about Search Domain](#).

**Search Type**
- **Maximum** Marker locates the maximum (highest) data value.
- **Minimum** Marker locates the minimum (lowest) data value.
- **Next Peak** Marker locates the peak with the next lower amplitude value relative to its starting position.
- **Peak Right** The marker locates the next valid peak to the right of its starting position on the X-axis.
- **Peak Left** The marker locates the next valid peak to the left of its starting position on the X-axis.

- **Threshold** Minimum amplitude (dB). To be considered valid, the peak must be above the threshold

Programming Commands
level. The valley on either side can be below the threshold level.

- **Excursion**  The vertical distance (dB) between the peak and the valleys on both sides. To be considered a peak, data values must "fall off" from the peak on both sides by the excursion value.

For more information, see [What is a Peak?](#)

**Target**  Enter the Target value. The marker moves to the first occurrence of the Target value to the right of its current position. Subsequent presses of the Execute button cause the marker to move to the next value to the right that meets the Target value. When the marker reaches the upper end of the stimulus range, it will "wrap around" and continue the search from the lower end of the stimulus range (left side of the window).

- **If Discrete Marker** is OFF, the marker locates the interpolated data point that equals the target value.
- **If Discrete Marker** is ON and there are two data points on either side of the target value, the marker locates the data point closest to the Target value.

**Bandwidth**  Four markers are automatically generated to find the first negative or positive bandpass in the selected search domain. Specify the level in dB from the peak or valley where bandwidth is measured.

- Bandwidth Search can be used ONLY with **Log Mag display format**.
- To use Bandwidth Search on a peak or valley other than the maximum or minimum values, change the Search Domain.

---

Enter a **Negative** number to search for a **Peak** bandpass, such as a filter S21 response:

- Marker 1: Maximum value within the **Search Domain**.
- Marker 2: Specified level DOWN the left of the peak.
- Marker 3: Specified level DOWN the right of the peak.
- Marker 4: Center frequency between markers 2 and 3.

Enter a **Positive** number to search for a **Valley** bandpass, such as a filter S11 response:

- Marker 1: Minimum value within the **Search Domain**.
- Marker 2: Specified level UP the left of the valley.
- Marker 3: Specified level UP the right of the valley.
- Marker 4: Center frequency between markers 2 and 3.

The following four values are displayed for Bandwidth Search:

- **BW**: (Marker 3 x-axis value) - (Marker 2 x-axis value) = width of the filter.
- **Center**  Mathematical midpoint between markers 2 and 3.
- **Q** Ratio of Center Frequency to Bandwidth (Center Frequency / Bandwidth).
- **Loss** Y-axis value of Marker 4. This is the loss of the filter at its center frequency. The ideal filter has no loss (0 dB) in the passband.

**Note** You must either press **Execute** or check **Tracking** to initiate all search types.

**Execute** Click to cause the marker to search for the specified criteria.

**Tracking** Check to cause the marker to search for the specified criteria with each new sweep. The searches begin with the first sweep after Tracking has been checked, based on the current search type and domain information. Therefore, make sure that the search criteria are in the desired state before using the data. You cannot manually change the stimulus setting for a marker if Tracking is selected for that marker.

**What Is a "Peak"?**

You define what the analyzer considers a "peak" by selecting the following two peak criteria settings:

- **Threshold** - Minimum amplitude (dB). To be considered valid, the peak must be **above** the threshold level. The valley on either side can be below the threshold level.

- **Excursion** - The vertical distance (dB) between the peak and the valleys on both sides. To be considered a peak, data values must "fall off" from the peak on both sides by the excursion value.

**Example:**

Threshold Setting: -10dB  
Excursion Setting: 1dB  
Scale = 1 dB / Division

**Mouse over the graphic to find a valid peak.**

- **Peak A** = Valid Peak (Above Threshold and Excursion Settings)
- **Peak B** = Invalid Peak (Below Excursion Setting)
- **Peak C** = Invalid Peak (Below Threshold Setting)
**Search Domain**

Search domain settings restrict the stimulus values (X-axis for rectangular format) to a specified span. Set the Start and Stop stimulus settings of these User spans. If Start is greater than Stop, the marker will not move.

- The default domain of each new marker is "full span".
- There are 16 user-defined domains for every channel.
- The user-defined domains can overlap.
- More than one marker can use a defined domain.

The graphic below shows examples of search domains.

![Search Domain Diagram](image)

**Marker Functions - Change Instrument Settings**

The following settings change the relevant PNA settings to the position of the active marker.

<table>
<thead>
<tr>
<th>How to change instrument setting using markers</th>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Using a mouse with PNA Menus</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Press Marker Function</td>
<td></td>
<td>1. Click Marker</td>
</tr>
<tr>
<td>2. Only Center, Ref Level, and Delay are available</td>
<td></td>
<td>2. then Marker Function</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Press Marker</td>
<td></td>
<td>1. Click Marker/Analysis</td>
</tr>
<tr>
<td>2. then [Marker Function]</td>
<td></td>
<td>2. then Marker Function</td>
</tr>
</tbody>
</table>
**Marker Function** dialog box help

**Note:** Marker Functions do not work with channels that are in **CW** or **Segment Sweep** mode.

**Marker =>Start**  Sets the start sweep setting to the value of the active marker.

**Marker =>Stop**  Sets the stop sweep setting to the value of the active marker.

**Marker =>Center**  Sets the center of the sweep to the value of the active marker.

**Marker =>Ref Level**  Sets the screen reference level to the value of the active marker.

**Marker =>Delay**  The phase slope at the active marker stimulus position is used to adjust the line length to the receiver input. This effectively flattens the phase trace around the active marker. (Additional Electrical Delay adjustments are required on devices without constant group delay over the measured frequency span.) You can use this to measure the electrical length or deviation from linear phase.

This feature adds phase delay to a variation in phase versus frequency; therefore, it is only applicable for ratioed measurements. (See Measurement Parameters.)

**Marker =>Span**  Sets the sweep span to the span that is defined by the delta marker and the marker that it references. Unavailable if there is no delta marker.

### How to select Advanced Marker settings

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Using a mouse with PNA Menus</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Programming Commands</strong></td>
<td></td>
</tr>
</tbody>
</table>

**For N5230A and E836xA/B models**

1. Press **Marker**

**Programming Commands**

1. Click **Marker**

2. then **Marker**

3. then **Advanced** on the Marker Dialog

**For PNA-X and 'C' models**

**Programming Commands**

1. Press **Marker**

2. then **Marker**

3. then **Advanced** on the Marker Dialog
1. Press **Marker**
2. then **[Properties]**
3. then **[Advanced Markers]**

### Advanced Marker

**Marker**  Specifies the marker number that you are defining.

**On**  Check to display the marker and corresponding data on the screen.

**Format**  Displays the marker data in a format that you choose. The marker format could be different from the grid format. In the default setting, the marker and grid formats are the same.

**Discrete Marker**  Check to display values at only the discrete points where data is measured. Clear to display values that are interpolated from the data points. The interpolated marker will report y-axis data from ANY frequency value between the start and stop frequency.

**Coupled Markers**  Check to couple markers by marker number, 1 to 1, 2 to 2 and so forth. The markers will remain coupled until this box in unchecked. [Learn more about coupled markers.](#)

**Marker Type**

**Normal**  Has a fixed stimulus position (X-axis) and responds to changes in data amplitude (Y-axis). It can be scrolled left and right on the X-axis by changing the marker stimulus value. Use this marker type with one of the marker search types to locate the desired data.

**Fixed**  Has a fixed X and Y-axis position based on its placement on the trace when it was set to fixed. It does NOT move with trace data amplitude. It can be scrolled left and right on the X-axis by changing the marker stimulus value.

Use this marker type to quickly monitor "before and after" changes to your test device. For example, you could use fixed markers to record the difference of test results before and after tuning a filter.

**Coupled Markers**

The coupled markers feature causes markers on different traces to line up with the markers on the selected trace. Markers are coupled by marker number, 1 to 1, 2 to 2, 3 to 3, and so forth. If the x-axis domain is the same (such as frequency or time), coupling occurs across all channels, windows, and traces. Trace markers in a different x-axis domain will not be coupled. If a trace marker has no marker to couple with on the selected trace, the marker remains independent.

**Coupled Markers Model**

This model simulates the use of coupled markers in the PNA.
1. Click **Trace A** or **Trace B**

2. Click **Coupled Markers**

3. Notice the following:
   * Markers on the unselected trace move to the x-axis position of the selected trace.
   * If a marker number on the unselected trace has no corresponding marker on the selected trace, no movement occurs for that marker.

4. Click **Reset** to run the model again. (There is no Reset for coupled markers on the PNA.)

Set Coupled Markers from the Advanced Markers dialog box.

**Marker Table**

You can display a table that provides a summary of marker data for the active trace. The marker data is displayed in the specified format for each marker.

**How to view the Marker Table**

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Marker</td>
</tr>
<tr>
<td></td>
<td>2. then Marker Table</td>
</tr>
<tr>
<td><strong>For PNA-X and ’C’ models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press DISPLAY</td>
<td>1. Click Response</td>
</tr>
<tr>
<td>2. then [More]</td>
<td>2. then Display</td>
</tr>
<tr>
<td></td>
<td>then [Tables]</td>
</tr>
<tr>
<td>---</td>
<td>------------------------------</td>
</tr>
<tr>
<td>3.</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td></td>
</tr>
</tbody>
</table>

Last Modified:

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-Jan-2008</td>
<td>Added bookmark to move marker</td>
</tr>
<tr>
<td>17-Jul-2007</td>
<td>Clarified bandwidth search</td>
</tr>
<tr>
<td>2-Feb-2007</td>
<td>MX Added UI</td>
</tr>
</tbody>
</table>
Using Math Operations

You can perform four types of math on the active trace versus a memory trace. In addition, three statistics (Mean, Standard Deviation, and Peak to Peak) can be calculated and displayed for the active data trace.

**Trace Math**

**Trace Statistics**

**Note:** Trace Math (described here) allows you to quickly apply one of four math operations using memory traces. Equation Editor allows you to build custom equations using several types of traces from the same, or different channels.

**Other Analyze Data topics**

**Trace Math**

To perform any of the math operations, you must first store a trace to memory. You can display the memory trace using the View options.

Trace math is performed on the complex data before it is formatted for display. See the PNA data processing map. Markers can be used while viewing a memory trace.

**How to select Trace Math**

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press MATH/MEMORY</td>
<td>1. Click Trace</td>
</tr>
<tr>
<td>2. Only the following are available from Active Entry</td>
<td>2. then Math</td>
</tr>
<tr>
<td>• Data&gt;&gt;Mem (stores Data trace into Memory)</td>
<td></td>
</tr>
<tr>
<td>• Data/Mem (performs math operation: Data divided by memory)</td>
<td></td>
</tr>
<tr>
<td>• Data (displays data trace with no math operation applied)</td>
<td></td>
</tr>
<tr>
<td>• Mem on/OFF (turns Memory trace on or off)</td>
<td></td>
</tr>
</tbody>
</table>

---

574
For PNA-X and 'C' models

1. Press MEMORY

Normalize, available only from the Memory menu, (not on the Math / Memory dialog), performs the same function as Data=>Memory, then Data / Memory.

**Math / Memory** dialog box help

**Normalize**, available only from the Memory menu, (not on the Math / Memory dialog), performs the same function as Data=>Memory, then Data / Memory.

**Data=>Memory** Puts the active data trace into memory. You can store one memory trace for every displayed trace.

**Data Math**
All math operations are performed on linear (real and imaginary) data before being formatted. See the PNA Data flow (below).

- **Data** Does no mathematical operation.
- **Data / Memory** - Current measurement data is divided by the data in memory. Use for ratio comparison of two traces, such as measurements of gain or attenuation. Learn more.
- **Data – Memory** - Data in memory is subtracted from the current measurement data. For example, you can use this feature for storing a measured vector error, then subtracting this error from the DUT measurement. Learn more.
- **Data + Memory** - Current measurement data is added to the data in memory. Learn more.
**Data * Memory** - Current measurement data is multiplied by the data in memory. [Learn more](#).

**8510 Mode** Check to simulate the Agilent 8510 data processing chain as it pertains to Trace Math and Memory. This setting applies to all channels. When the box is checked or cleared, the PNA performs an [Instrument Preset](#) and retains its setting through subsequent Instrument Presets.

This setting is saved as part of an [instrument state](#). However, when recalled, this setting is assumed only temporarily. When a subsequent PNA Preset is performed, the PNA reverts to the setting that was in effect before the state was recalled.

This represents the relevant portion of the data flow. [See the entire PNA data processing chain](#).

A settings change in any of the operations that occur after the Memory operation on the above PNA Data Flow diagram changes both the Data trace and the Memory trace. For example, after storing a data trace to memory, when you change the format for the Data Trace, the format for the Memory Trace is also changed to the same setting.

**Trace View Options**

- **Data Trace** Displays ONLY the Data trace (with selected math operation applied).
- **Memory Trace** Displays ONLY the trace that was put in memory.
- **Data and Memory Trace** Displays BOTH the Data trace (with selected math operation applied). and the trace that was put in memory.

[Learn more about Trace Math](#) (scroll up)

**Data / Memory** and **Data - Memory**

(Data / Memory) and (Data - Memory) math operations are performed on linear data before it is formatted. Because data is often viewed in log format, it is not always clear which of the two math operations should be used. Remember: dividing linear data is the same as subtracting logarithmic data. The following illustrates, in general, when to use each operation.

Use **Data / Memory** for normalization purposes, such as when comparing S21 traces "before" and "after" a change is made or measurement of trace noise. In the following table, the Data/Mem values intuitively show the differences between traces. It is not obvious what Data-Mem is displaying.
<table>
<thead>
<tr>
<th>S21 values to compare</th>
<th>Data/Mem</th>
<th>Data-Mem</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5 dB and 0.6 dB</td>
<td>0.1 dB</td>
<td>-39 dB</td>
</tr>
<tr>
<td>0.5 dB and 0.7 dB</td>
<td>0.2 dB</td>
<td>-33 dB</td>
</tr>
</tbody>
</table>

Use **Data - Memory** to show the relative differences between two signals. Use for comparison of very small signals, such as the S11 match of two connectors.

In the following table, Data/Mem shows both pairs of connectors to have the same 2 dB difference. However, the second pair of connectors have much better S11 performance (-50 and -52) and the relative significance is shown in the Data-Mem values.

<table>
<thead>
<tr>
<th>S11 values to compare</th>
<th>Data/Mem</th>
<th>Data-Mem</th>
</tr>
</thead>
<tbody>
<tr>
<td>-10 dB and -12 dB</td>
<td>2 dB</td>
<td>-24 dB</td>
</tr>
<tr>
<td>-50 dB and -52 dB</td>
<td>2 dB</td>
<td>-64 dB</td>
</tr>
</tbody>
</table>

**Data * Memory and Data + Memory**

Use **Data * Memory and Data + Memory** to perform math on an active data trace using data from your own formulas or algorithms rather than data from a measurement. For example, if you want to simulate the gain of a theoretical amplifier placed in series before the DUT, you could do the following:

1. Create an algorithm that would characterize the frequency response of the theoretical amplifier.
2. Enter complex data pairs that correspond to the number of data points for your data trace.
3. Load the data pairs into memory with SCPI or COM commands. The analyzer maps the complex pairs to correspond to the stimulus values at the actual measurement points.
4. Use the **data + memory** or **data * memory** function to add or multiply the frequency response data to the measured data from the active data trace.

**Note:** The data trace must be configured before you attempt to load the memory.

**Trace Statistics**

You can calculate and display statistics for the active data trace. These statistics are:

- Mean
- Standard deviation
- Peak-to-peak values

You can calculate statistics for the full stimulus span or for part of it with user ranges.

There are nine user ranges per channel. These user ranges are the same as the search domains specified for a
marker search in that same channel; they use the same memory registers and thus share the same stimulus spans. If you specified search domains with marker search for a channel, you can recall these same spans by selecting the corresponding user ranges. The user ranges for a channel can overlap each other.

A convenient use for trace statistics is to find the peak-to-peak value of passband ripple without searching separately for the minimum and maximum values.

The trace statistics are calculated based on the format used to display the data.

- **Rectangular data formats** are calculated from the scalar data represented in the display
- **Polar** or **Smith Chart** formats are calculated from the data as it would be displayed in **Log Mag** format

See how to make Trace Statistics display settings.

### How to activate Trace Statistics

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Trace</td>
</tr>
<tr>
<td></td>
<td>2. then Statistics</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press ANALYSIS</td>
<td>1. Click Marker/Analysis</td>
</tr>
<tr>
<td>2. then [Statistics]</td>
<td>2. then Analysis</td>
</tr>
<tr>
<td>3. then [Trace Statistics]</td>
<td>3. then Trace Statistics</td>
</tr>
</tbody>
</table>
Trace Statistics dialog box help

See how to make Trace Statistics display settings.

Statistics  Check to display mean, standard deviation, and peak to peak values for the active trace.

Span  Specifies the span of the active trace where data is collected for a math operation. You can define up to 9 user spans per channel with Start and Stop. You can also define the user spans from the Marker Search dialog box.

Start  Defines the start of a user span.

Stop  Defines the stop of a user span.

Learn more about Trace Statistics (scroll up)

Last Modified:

27-Aug-2007  Edited trace display settings

2-Feb-2007  MX added UI
Equation Editor

Equation Editor, new with PNA release 6.03, allows you to enter an algebraic equation that can mathematically manipulate measured data. The results are displayed as a data trace. Data that is used in the equation can be from the same or different channels.

**Note:** Equation Editor is NOT available with FCA measurements.

### Overview

*How to start Equation Editor*

*Using Equation Editor*

*Data that is used in Equation Editor*

*Trace Settings, Error Correction, and an Example*

*Functions and Constants*

*Operators*

*Example Equations*

*Saving Equation Editor Data*

---

### Other 'Analyze Data' topics

---

### Overview

Equation Editor allows you to enter an algebraic equation of standard mathematical operators and functions, referencing data that is available in the PNA. Once a valid equation is entered and enabled, the display of the active trace is replaced with the results of the equation, and updated in real-time as new data is acquired. For equations that can be expressed with Equation Editor's supported functions, operators, and data. There is no need for off-line processing in a separate program.

For example, enter the equation “S21 / (1 - S11)”. The resulting trace is computed as each S21 data point divided by one minus the corresponding S11 data point. For a 201 point sweep setup, the computation is repeated 201 times, once for each point.

As another example, suppose you want the PNA to make a directivity measurement of your 3-port DUT. This is not a “native” PNA measurement, but can be achieved using the Equation Editor. The desired result is the sum and difference of LogMag formatted traces, expressed as: S12 + S23 - S13.

Because Equation Editor operates on **unformatted complex data**, the required equation is:

\[
\text{DIR} = \frac{S12 \times S23}{S13}
\]

DIR becomes a display label to help you identify the computed data trace.

On the equation trace, set the format to LogMag.
### How to start Equation Editor

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Trace</td>
</tr>
<tr>
<td></td>
<td>2. then Equation</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press ANALYSIS</td>
<td>1. Click Marker/Analysis</td>
</tr>
<tr>
<td>2. then [Equation Editor]</td>
<td>2. then Analysis</td>
</tr>
<tr>
<td></td>
<td>3. then Equation Editor</td>
</tr>
</tbody>
</table>

#### Equation Editor

![Equation Editor](image)

**Equation:**

S21 / Mem

**Functions/Constants:**
- built-in
- arcsin
- atan
- atan2
- cos
- cpx

**Operators:**
- +
- *
- /

**Trace Data:**
- Tr1

**Ch Param Data:**
- Mem
- S11
- S12
- S13
- S14
- S21
- S22
- S23
- S24

**Import Functions**

---

581
Notes

- **Double-click**, or type, the Functions, Operators, and Data to build an Equation.
- Equation Editor is NOT available with FCA measurements.
- Scroll down to learn more about Using Equation Editor

**Equation**: The field in which equations are built. Click the down arrow to the right to use or modify equations that have been previously saved. This is where equations are saved when you press 'Store Equation'.

**Enabled** Check this box to enable the equation that is currently in the Equation field. If the Enabled box is not available, then the equation is not valid. If a data trace is used that is from a different channel than the Equation trace, the channels MUST have the same number of data points to be valid.

**<-Backspace** Moves the cursor to the left while erasing characters.

**<-** Moves the cursor to the left without erasing characters.

**-->** Moves the cursor to the right without erasing characters.

**Store Equation** Press to save the current equation. To later recall the equation, click the down arrow to the right of the equation.

**Delete Equation** Removes the current equation from the drop-down list.

**Functions/Constants**: See descriptions of Functions.

**Operators**: See descriptions of Operators.

**Trace Data**: Select from ALL of the currently **displayed** traces on ALL channels.

**Ch Param Data**: Select from **undisplayed** data that is available ONLY from the active channel (same channel as the equation trace).

**Note**: With an external test set enabled, only parameters involving ports 1 through 4 are listed. However, all available parameters can be typed directly into the **Equation** field.

See **Data that is used in Equations**.

**Show "Tr" annotation** Check to show the TrX annotation on PNA display and **Trace Status** buttons.

**Keypad**: Provided to allow navigation of the entire dialog with a mouse.

### Using Equation Editor

1. **Pick a trace in which to enter the equation**
   - Equation Editor works on the active trace.
   - Either create a new trace, or click the **Trace Status** button on an existing trace to make the trace active.

2. **Enter an equation**
   Start Equation Editor (click **Trace**, then **Equation**)
**Note:** Equation Editor is NOT available for FCA measurements.

- The equation text can be in the form of an expression \((S21)/(1-S11)\) or an equation \((\text{DIR} = S12 \times S23 / S13)\). This topic refers to both types as equations.
- Either type, or double-click the Functions, Operators, and Data to build an equation.
- Functions and Constants ARE case-sensitive; Data names are NOT case sensitive.
- [Learn more about referring to data traces.](#)

### 3. Check for a valid equation

When a valid equation is entered, the Enabled checkbox becomes available for checking. When the Enabled box is checked:

- The Equation Trace becomes computed data.
- The equation is visible on the Trace Status (up to about 10 characters).
- The equation is visible in the trace Title area (up to about 45 characters) when the Equation trace is active.
- The equation is visible in the Status Bar at the bottom of the display. This is updated only after the equation is entered and the Trace Status button is clicked.
- If an equation is NOT valid, and a trace from a different channel is used, make sure the number of data points is the same for both channels.

Learn more about the [Functions](#), [Operators](#), and [Data](#) that are used in Equation Editor.

### Data that is used in Equation Editor

**Definitions**

- **Equation trace** A trace in which an equation resides.
- **Referred trace** A trace that is used as data in an equation.

**Example:** `eq=Tr2+S11` is entered into \(\text{Tr1}\).
- \(\text{Tr1}\) becomes an equation trace.
- \(\text{Tr2}\) and \(\text{S11}\) are both referred traces because they are used in the equation trace.

**Notes**

- Referred traces are processed one data point at a time. For example, the expression “\(S11/S21\)” means that for each data point in \(S11\) and \(S21\), divide point \(N\) of \(S11\) by point \(N\) of \(S21\).
- Once an equation is enabled, the trace is no longer identified by its original measurement parameter. It becomes an equation trace.
An equation trace can NOT refer to itself. For example, an equation in Tr1 cannot refer to trace Tr1.

Referred traces can be selected from S-Parameters, Receiver data, and Memory traces.

See note regarding External Test Sets.

**There are three ways to refer to traces:**

The following distinction is important when discussing the three ways to refer to traces/data.

- **Trace** - a sequential collection of data points that are displayed on the PNA screen.
- **Data** - PNA measurements that are acquired but not displayed. When an equation trace refers to data that is not displayed, the PNA will automatically acquire the data.

1. **Using TrX Trace notation (for example, Tr2).**

   When a trace is created, check *Show Tr Annotation* to see the Tr number of that trace.

   - **Simple** - ALWAYS refers to displayed traces.
   
   - Must be used for referring to traces in a different channel as the equation trace.
   
   - All trace settings are preserved in the equation trace. If you do NOT want a trace setting to be used in the equation trace, you must disable it in the referred trace.
   
   - If the referred trace is error corrected, then that data is corrected in the equation trace.
   
   - Used to refer to a memory trace (it must already be stored in memory). Append .MEM to the TrX trace identifier. For example, Tr2.mem refers to the memory trace that is stored for Tr2.

2. **Using S-parameter notation (for example, S11/S21)**

   - **Convenient** - ALWAYS refers to data that is NOT displayed.
   
   - Refers to data that resides in the same channel as the equation.
   
   - NOT the same as referring to a displayed S11 trace using TrX notation. See Example.

   - The referred data includes NO trace settings.
   
   - If the channel has error correction available, then it can be applied by turning error correction ON for the Equation trace.

3. **Using Receiver notation (for example AB_2); NOT case sensitive.**

   At least one receiver is required, followed by an underscore and a number.

   - The letters before the underscore refer to the receivers.

   - Letters alone refer to physical receivers.
Letters immediately followed by numbers refer to logical receivers. Learn more.

If two receivers are referenced, they are ratioed.

The number after the underscore refers to the source port for the measurement.

Examples

- AR1_2 = physical receiver A / physical receiver R1 with 2 as the source port.
- a3b4_1 = reference receiver for port 3 / test port receiver for port 4 with 1 as the source port.

Learn more about ratioed and unratioed receiver measurements.

Receiver notation is like S-parameter notation in that:

- Refers to data that is NOT displayed and resides in the same channel as the equation.
- The referred data includes NO trace settings.
- If the channel has error correction available for that receiver, then it can be applied by turning error correction ON for the Equation trace.

Referring to Traces in a different channel

When the equation trace refers to a trace on a different channel:

- The trace must already be displayed.
- Must refer to the trace using TrX notation.
- The Equation trace and the referred trace MUST have the same number of data points or the Enable checkbox will not be available.
- The Equation trace is updated when the last referred data in the same channel is acquired. Therefore, to prevent 'stale' data from being used, the Equation trace must be on a higher numbered channel than the referred trace. This is because the PNA acquires data in ascending channel number order - first channel 1, then channel 2, and so forth. If the Equation trace is on channel 1, and it refers to a trace on channel 2, the Equation trace will update after channel 1 is finished sweeping, using 'old' data for the channel 2 trace.

Trace Settings, Error Correction, and an Example

This discussion highlights the differences between using S-parameter / Receiver notation and TrX notation when referring to traces. The key to understanding the differences is realizing that S-parameter / Receiver notation ALWAYS refers to data that is NOT displayed.
Trace Settings  Normalization, Trace Math, Gating, Phase and Mag Offset, Electrical Delay, Time Domain.

Equation Editor  processing occurs on the equation trace immediately after error correction.

Referred Data/Trace  (used in the equation) is taken from the following locations:

- When using TrX notation, data is taken immediately before formatting. These traces are always displayed and include Trace Settings.

- When using S-parameter / Receiver notation, data is taken immediately after error correction. This data is NOT displayed and includes NO trace settings (see example).

Error-correction and Equation Editor

Using TrX notation:

- The Trace Settings and Error-correction on the referred trace are used in the Equation trace.

- If error correction is NOT ON, then the raw, uncorrected data is used in the equation trace.

- To see if error correction is ON, make the trace active, then see the Correction level in the status bar.

- Turning error correction ON/OFF on the equation trace has no meaning. The referred data that is used in the equation is ALWAYS what determines its level of correction.

Using S-parameter and Receiver notation:

- Because the data is not displayed, NO trace settings are used in the Equation trace.

- Correction can be turned ON/OFF if corrected data is available for the referred data. Exception: When using S-parameter and Receiver notation to refer to a trace on a channel that has been calibrated with a Response Cal or Receiver Cal, correction can NOT be turned ON, even though the Status Bar indicates otherwise. For example: Tr1 is an S11 measurement with a Response Cal. Tr2 is an equation trace that refers to S11. The Tr2 equation trace is NOT corrected, even though the Status Bar may indicate that it is corrected. However, if Tr2 refers to Tr1 (not S11), the Tr2 equation trace is corrected.

Example

This example illustrates the differences when referring to a trace using S-parameter notation and TrX notation:
• **Tr1** is an S11 measurement with no equation, 2-port correction ON, and Time Domain transform ON.

• **Tr2** is an equation trace that refers to **Tr1**. Tr2 is corrected because Tr1 is corrected. Tr2 is transformed because Tr1 is transformed. If transform is turned ON for Tr2, the data will be transformed AGAIN, which results in "unusual" data.

• **Tr3** is an equation trace that refers to **S11**. This is NOT the same as referring to Tr1. The S11 trace that is referred to is a different instance of S11 that is NOT displayed, and has NO trace settings. Notice that Tr3 data is NOT transformed, although Tr1 is transformed. Correction for **Tr3** can be turned ON and OFF because a calibration was performed on the channel in which the S11 trace resides.

• **Note**: X-axis annotation of the Equation trace is completely independent of the data that is presented. ONLY the data values from a referred trace are used. For example, notice that the Equation trace **Tr2** has Frequency on the X-axis although the referred trace **Tr1** is presented in Time.

### Functions and Constants used in Equation Editor

ALL trace data that is used in Equation Editor is unformatted, complex data.

In the following table,

- Function(scalar x) means that an automatic conversion from a complex number to its scalar magnitude is performed before passing the value to the function.

- Function(complex x) means that the entire complex value is used.

- a, b, c, d are arguments that are used in the function.

<table>
<thead>
<tr>
<th>Function/Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acos(scalar a)</td>
<td>returns the arc cosine of a in radians</td>
</tr>
<tr>
<td>asin(scalar a)</td>
<td>returns the arc sine of a in radians</td>
</tr>
</tbody>
</table>
atan(scalar a) returns the arc tangent of a in radians

atan2 returns the phase of complex a = (re,im) in radians

has the following two argument sets:

- atan2(complex a) - returns the phase in radians
- atan2(scalar a, scalar b)

cpx(scalar a, scalar b) returns a complex value (a+i*b) from two scalar values

cos(complex a) takes a in radians and returns the cosine

e returns the constant =~ 2.71828...

exp(complex a) returns the exponential of a

im(complex a) returns the imag part of a as the scalar part of the result (zeroes the imag part)

kfac(complex a, complex b, complex c, complex d)

k-factor:
k = (1 - |a|^2 - |d|^2 + |a*d-b*c|^2) / (2 * |b*c|)

when entered in EE:
kfac(S11,S21,S12,S22)

ln(complex a) returns the natural logarithm of a

log10(complex a) returns the base 10 logarithm of a

mag(complex a) returns sqrt(a.re*a.re+a.im*a.im)

max(complex a, complex b, ...) returns the complex value that has the largest magnitude of a list of values.

median(complex a, complex b, ...)

- The median is determined by sorting the values by magnitude, and returning the middle one.
- If an even number of values is passed, then the smaller of the two middle values is returned.

min(complex a, complex b, ...)

returns the complex value that has the smallest magnitude of a list of values.

mu1(complex a, complex b, complex c, complex d)

mu1 = (1 - |a|^2) / ( |d - conj(a) * (a*d-b*c)| + |b*c| )

when entered in EE:
mu1(S11,S21,S12,S22)
\[
\mu_2( \text{complex } a, \text{complex } b, \\
    \text{complex } c, \text{complex } d ) \\
\text{when entered in EE:} \\
\mu_1(S_{11}, S_{21}, S_{12}, S_{22}) \\
\mu_2 = \frac{(1 - |d|^2)}{|a - \text{conj}(d) - (a^*d-b*c)| + |b*c|}
\]

for both \( \mu_1 \) and \( \mu_2 \) (Usually written with the Greek character \( \mu \) )

- \( \text{conj} \) is the complex conjugate. For scalars \( a \) and \( b \), \( \text{conj}(a+ib) = (a-ib) \)

- returns a scalar result - the imaginary part of the complex result is always 0

\( \text{phase}(\text{complex } a) \) returns \( \text{atan2}(a) \) in degrees

\( \text{PI} \) returns the numeric constant \( \pi \) (3.141592), which is the ratio of the circumference of a circle to its diameter

\( \text{pow}(\text{complex } a, \text{complex } b) \) returns \( a \) to the power \( b \)

\( \text{re}(\text{complex } a) \) returns the scalar part of \( a \) (zeroes the imag part)

\( \text{sin}(\text{complex } a) \) takes \( a \) in radians and returns the sine

\( \text{sqrt}(\text{complex } a) \) returns the square root of \( a \), with phase angle in the half-open interval \( (-\pi/2, \pi/2] \)

\( \text{tan}(\text{complex } a) \) takes \( a \) in radians and returns the tangent

\( \text{xAxisIndex}(\text{scalar } a) \) \textbf{New} returns the numeric data point \( a \) of the sweep

\( \text{xAxisTraceData}(\text{scalar } a) \) \textbf{New} for each point \( a \) on the sweep, returns the x-axis value on the selected channel.

---

**Operators used in Equation Editor**

---

589
<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Addition</td>
</tr>
<tr>
<td>-</td>
<td>Subtraction</td>
</tr>
<tr>
<td>*</td>
<td>Multiplication</td>
</tr>
<tr>
<td>/</td>
<td>Division</td>
</tr>
<tr>
<td>(</td>
<td>Open parenthesis</td>
</tr>
<tr>
<td>)</td>
<td>Close parenthesis</td>
</tr>
<tr>
<td>,</td>
<td>Comma - separator for arguments (as in S11, S22)</td>
</tr>
<tr>
<td>=</td>
<td>Equal (optional)</td>
</tr>
<tr>
<td>E</td>
<td>Exponent (as in 23.45E6)</td>
</tr>
</tbody>
</table>

**Example Equations**

The following examples may help you get started with Equation Editor.

**Offset each data point in Tr2 from Tr1 by 2dB**

Use the function: pow(complex a, complex b) -- returns a to the power b.

\[
20\log(a) + 2 = 20\log(x) \\
\log(a) + 2/20 = \log(x) // divide all by 20. \\
x = 10^{\log(a) + 2/20} // swap sides and take 10 to the power of both sides \\
x = 10^{\log(a)} * 10^{2/20} \\
x = a * 10^{2/20} \\
\]

The equation is entered into Tr2 as:

**Offset=Tr1*\text{pow}(10, 2/20)**

To offset by 5 dB

**Offset=Tr1*\text{pow}(10, 5/20)**.

**Balanced Match using a 2-port PNA**

\[
\text{SDD11} = (\text{S11}-\text{S21}-\text{S12}+\text{S22})/2
\]

**Conversion loss**

\[
\text{B}_1/\text{pow}(10, -15/20)
\]

- \(\text{B}_1\) is a receiver measurement;
• -15 is the input power in dBm

**Third-order intercept point (IP3 or TOI)**

\[ TR1 \times \sqrt{TR1/TR3} \]

- \( TR1 \) = input signal power
- \( TR3 \) = intermodulation power (both traces measured with single receivers)

**Harmonics in dBc**

\[ B_1/TR2 \]

- \( B_1 \) is tuned to a harmonic frequency
- \( TR2 \) = power at fundamental frequency, measured with \( B_1 \) receiver

**PAE (Power Added Efficiency)**

\( P_{out} - P_{in} / P_{dc} \)

Type the following equation into a new trace with an unratioed measurement, such as AI1. The data format is REAL:

\[ PAE = 100 \times (0.001 \times \text{pow(mag}(TR1), 2) - (0.001 \times \text{pow(mag}(TR1), 2) / \text{pow(mag}(TR2), 2))) / (TR3 \times TR4) \]

Where:

- \( TR1 \) - a trace that measures unratioed B receiver.
- \( TR2 \) - a corrected S21 trace (amplifier gain)
- \( TR3 \) - a trace that measures ADC voltage (AI1) across a sensing resistor.
- \( TR4 \) - an equation trace containing \( Isupp = (TR3 / \text{value of sensing resistor}) \).

Data is displayed in Real format with units actually being watts.

**1-port Insertion Loss**

When it is not possible to connect both ends of a cable to the PNA, a 1-port insertion loss measurement can be made. However, the measured loss must be divided by 2 because the result includes the loss going down and coming back through the cable. This assumes that the device is terminated with a short to reflect all of the power. The ‘divide by 2’ operation is performed as follows using Equation Editor:

- \( TR1 \) - an S11 trace
- \( TR2 \) - an equation trace containing \( 20 \times \text{log10}(TR1) / 2 \)

**Saving Equation Editor Data**
Equation data can be saved to the PNA hard drive in the following formats:

- **Citifile (.cti)** - Equation data is saved and recalled. The file header indicates the "underlying" s-parameter trace type.
- **Trace (.prn)** - read by spreadsheet software. Can NOT be recalled by the PNA.
- **Print to File** (bmp, jpg, png) - saves image of PNA screen.

Equation data is NOT saved in .SnP file format. When attempting to save an Equation trace in .SnP format, the "underlying" S-parameter data is saved; not Equation data.

---

Last Modified:

- 17-Oct-2007    Added new functions
- 30-Aug-2007     Added 1-port insertion loss
- 3-Jul-2007      Added PAE and other notes
- 18-Jun-2007     Added examples
Using Limit Lines

Limit lines allow you to compare measurement data to performance constraints that you define.

**Overview**

**Create and Edit Limit Lines**

**Display and Test with Limit Lines**

**Testing with Sufficient Data Points**

---

### Overview

Limit lines are visual representations on the PNA screen of the specified limits for a measurement. You can use limit lines to do the following:

- Give the operator **visual guides** when tuning devices.
- Provide **standard criteria** for meeting device specification.
- Show the **comparison** of data versus specifications.

Limit testing compares the measured data with defined limits, and provides optional **Pass or Fail** information for each measured data point.

You can have up to **100** discrete lines for each measurement trace allowing you to test all aspects of your DUT response.

Limit lines and limit testing are NOT available with **Smith Chart** or **Polar** display format. If limit lines are ON and you change to Smith Chart or Polar format, the analyzer will automatically disable the limit lines and limit testing.

### Create and Edit Limit Lines

You can create limit lines for all measurement traces. The limit lines are the same color as the measurement trace.

Limit lines are made up of discrete lines with four coordinates:

- **BEGIN and END stimulus** - X-axis values.
- **BEGIN and END response** - Y-axis values.
How to create, edit, and test with Limit Lines

All limit line settings are made with the limit table. Use one of the following methods to show the limit table:

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press LIMIT TABLE</td>
<td>1. Click Trace</td>
</tr>
<tr>
<td>2. then Active Entry keys</td>
<td>2. then Limit Test</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press ANALYSIS</td>
<td>1. Click Marker/Analysis</td>
</tr>
<tr>
<td>2. then [Limits]</td>
<td>2. then Analysis</td>
</tr>
<tr>
<td>3. then [Limit Test]</td>
<td>3. then Limit Test</td>
</tr>
</tbody>
</table>

Limit Table

<table>
<thead>
<tr>
<th>Type</th>
<th>BEGIN STIMULUS</th>
<th>END STIMULUS</th>
<th>BEGIN RESPONSE</th>
<th>END RESPONSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN</td>
<td>1.0000000 Hz</td>
<td>1.0000000 Hz</td>
<td>50.000000 dB</td>
<td>5.000000 dB</td>
</tr>
<tr>
<td>MAX</td>
<td>1.0000000 Hz</td>
<td>1.0000000 Hz</td>
<td>50.000000 dB</td>
<td>5.000000 dB</td>
</tr>
<tr>
<td>OFF</td>
<td>0.0000000 Hz</td>
<td>0.0000000 Hz</td>
<td>0.000000 dB</td>
<td>0.000000 dB</td>
</tr>
</tbody>
</table>

**Note:** To ADD a limit line to the table, change the last limit line to either MAX or MIN

1. In the Type area of the Limit Table, select MIN or MAX for Limit Line 1.
   - The MIN value will fail measurements BELOW this limit.
   - The MAX value will fail measurements ABOVE this limit.
2. Click BEGIN STIMULUS for Limit Segment 1. Enter the desired value.
3. Click END STIMULUS for Limit Segment 1. Enter the desired value.
4. Click BEGIN RESPONSE for Limit Segment 1. Enter the desired value.
5. Click END RESPONSE for Limit Segment 1. Enter the desired value.
6. Repeat Steps 1-5 for each desired limit line.

Displaying and Testing with Limit Lines

After creating limit lines, you can then choose to display or hide them for each trace. The specified limits remain valid even if limit lines are not displayed.
Limit testing cannot be performed on memory traces. You can choose to provide a visual and/or audible PASS/FAIL indication.

With limit testing turned ON:

- Any portion of the measurement trace that **fails** is displayed in red.
- Any portion of the measurement trace that does **NOT fail** remains unchanged and silent.

**PASS is the default mode of Pass/Fail testing.** A data point will FAIL only if a measured point falls outside of the limits.

- If the limit line is set to OFF, the entire trace will PASS.
- If there is no measured data point at a limit line stimulus setting, that point will PASS.

---

**Limit Test** dialog box help

**Show Table**  Shows the table that allows you to create and edit limits.
**Hide Table**  Makes the limits table disappear from the screen.

**Note:** To ADD a limit line to the table, change the last limit line to either MAX or MIN

---

**Limit Test**

- **Limit Test ON**  Check the box to compare the data trace to the limits and display PASS or FAIL.
- **Limit Line ON**  Check the box to make the limits visible on the screen. (Testing still occurs if the limits are not visible.)
- **Sound ON Fail**  Check the box to make the PNA beep when a point on the data trace fails the limit test.

**Global Pass/Fail**

The Pass/Fail indicator provides an easy way to monitor the status of ALL measurements.

- **Global pass/fail display ON**  Check to display the Global Pass/Fail status.
- **Policy:**  Choose which of the following must occur for the Global Pass/Fail status to display PASS:

---

595
- All Tests (with Limit Test ON) Must Pass - This setting reads the results from the Limit Tests. If all tests (with Limit Test ON) PASS, then the Global Pass/Fail status will PASS.

- All Measurements Must Pass - This more critical setting shows FAIL unless all measured data points fall within established test limits and Limit Test is ON. Note: In this mode, if one measurement does NOT have Limit Test ON, Global Pass/Fail will show FAIL.

Learn more about displaying and testing with Limits (scroll up)

**Testing with Sufficient Data Points**

Limits are checked only at the actual measured data points. Therefore, it is possible for a device to be out of specification without a limit test failure indication if the data point density is insufficient.

The following image is a data trace of an actual filter using 11 data points (approximately one every vertical graticule). The filter is being tested with a minimum limit line (any data point under the limit line fails).

Although the data trace is clearly below the limit line on both sides of the filter skirts, there is a PASS indication because there is no data point being measured at these frequencies.

The following image shows the exact same conditions, except the number of data points is increased to 1601. The filter now fails the minimum limit test indicated by the red data trace.
Last Modified:

2-Feb-2007  MX Added UI
Save and Recall a File

The PNA allows you to save and recall files to and from an internal or external storage device in a variety of file formats.

- **How to Save a File**
- **How to Recall a File**
- **Instrument / Calibration State Files** (.csa, .cst, .sta, .cal)
- **Measurement Data Files** (.prn, .sNp, .cti, .csv)
- **Define Data Saves**
- **Managing Files without a Mouse**

### Other Data Outputting topics

#### How to Save a File
Use one of the following methods:

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>SAVE</strong></td>
<td>1. Click <strong>File</strong></td>
</tr>
<tr>
<td>2. then <strong>Save, Save As</strong>, or <strong>Auto Save</strong></td>
<td>2. then <strong>Save, Save As</strong>, or <strong>Auto Save</strong></td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Press <strong>SAVE</strong></td>
<td>1. Click <strong>File</strong></td>
</tr>
<tr>
<td>2. then <strong>[Save], [Save As], or [Auto Save]</strong></td>
<td>2. then <strong>Save</strong> or <strong>Save As</strong></td>
</tr>
</tbody>
</table>

**Save**  Immediately saves the PNA state and possibly calibration data to the filename and extension you used when you last performed a Save. Only .cst, .sta, and .csa files are remembered when Save is performed. This file will be overwritten the next time you click **Save**. To prevent this, use one of the following methods.

**Save As**  Invokes the **Save As** dialog box.

**Auto Save**  (Only available from the Active Entry keys) Saves state and calibration data to the internal hard disk in the C:\Program Files\Agilent\Network Analyzer\Documents folder. A filename is generated automatically using the syntax "atxxx.csa"; where xxx is a number that is incremented by one when a new file is Auto
Saved.

**Note:** You can NOT save Frequency Converter Application .S2P files using this method. To learn how, see Using FCA, Save Data.

---

**Save As** dialog box help

**Save in** Allows you to navigate to the directory where you want to save the file.

**File name** Displays the filename that you either typed in or clicked on in the directory contents box.

**Save as type**

The following file types save **Instrument states and Calibration data**. You can save, and later recall, instrument settings and calibration data for **all channels** currently in use on the PNA.

These file types are only recognized by Agilent PNA Series analyzers. Learn more about these file types.

- **.csa** - save Instrument state and actual Cal Set data (cal/state archive) **Default selection**.
- **.cst** - save Instrument state and a link to the Cal Set data.
- **.sta** - save Instrument state ONLY (no calibration data)
- **.cal** - save actual Calibration data ONLY (no Instrument state)

**Note:** Before saving a .cst file (Instrument State and link to Cal Set), be sure that a User Cal Set is being used for the calibration; not a Cal Register. Cal Registers are overwritten with new data whenever a calibration is performed, and may not be accurate cal data when the .cst file is recalled. Learn more about Cal Sets.

The following file types save **Measurement data** for use in spreadsheet or CAE programs. Click to learn more about these file types.
- *.prn
- *.sNp
- *cti (citifile)
- *.csv (used to save 2D Gain Compression data).

**Note:** To save the PNA screen as .bmp, .jpg, or .png graphics file types, click **File / Print to File.** Learn more.

**Save** Saves the file to the specified file name and directory.

---

**How to Recall (open) a file**

Select a file from the 'most recently used' list. The list is saved when the PNA application exits.

Use one of the following methods:

**Using front-panel HARDKEY [softkey] buttons**

For **N5230A and E836xA/B models**

1. Press **RECALL**
2. then **Active Entry** keys

For **PNA-X and 'C' models**

1. Press **RECALL**
2. then **[Recall]**

**Using a mouse with PNA Menus**

1. Click **File**
2. then **Recall**

---

![Recall screen capture](image)
Recall dialog box help

Look in  Allows you to select the directory that contains the file that you want to recall.

File name  Displays the filename that you either typed in or clicked on in the directory contents box.

Files of type  Allows you view and select files that are listed in categories of a file type.

Recall  Recalls the file displayed in the file name box.

Note: *.sNp files cannot be recalled by the PNA.

Instrument State / Calibration Files

You can save, and later recall, instrument settings and calibration data for all channels currently in use on the PNA.

An Instrument State contains almost every PNA setting. The following PNA settings are NOT saved and recalled with Instrument State:

- GPIB address
- RF power ON/OFF (depends on current setting)
- Test set I/O settings

The following file types are used to save and recall instrument states and Cal Set information:

<table>
<thead>
<tr>
<th>File Types</th>
<th>Information that is stored for each channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>.cst</td>
<td>Instrument State Information</td>
</tr>
<tr>
<td>.sta</td>
<td>Channels/Traces</td>
</tr>
<tr>
<td>.csa</td>
<td>Windows</td>
</tr>
<tr>
<td>.cal</td>
<td>Triggering</td>
</tr>
<tr>
<td></td>
<td>Format</td>
</tr>
<tr>
<td></td>
<td>Scale</td>
</tr>
<tr>
<td></td>
<td>Stimulus Information:</td>
</tr>
<tr>
<td></td>
<td>Frequency range</td>
</tr>
<tr>
<td></td>
<td>Number of points</td>
</tr>
<tr>
<td></td>
<td>IF bandwidth</td>
</tr>
<tr>
<td></td>
<td>Sweep type</td>
</tr>
<tr>
<td></td>
<td>Sweep mode</td>
</tr>
<tr>
<td></td>
<td>Port powers</td>
</tr>
<tr>
<td></td>
<td>Source attenuators</td>
</tr>
<tr>
<td></td>
<td>Receiver attenuators</td>
</tr>
<tr>
<td></td>
<td>Test Set port map</td>
</tr>
<tr>
<td></td>
<td>Alternate sweep</td>
</tr>
<tr>
<td></td>
<td>Stimulus Information:</td>
</tr>
<tr>
<td></td>
<td>GUID (Globally Unique Identifier)</td>
</tr>
<tr>
<td></td>
<td>provides link to Cal Set</td>
</tr>
<tr>
<td></td>
<td>Name, Description, Modify date</td>
</tr>
<tr>
<td></td>
<td>Stimulus Information:</td>
</tr>
<tr>
<td></td>
<td>Frequency range</td>
</tr>
<tr>
<td></td>
<td>Alternate sweep</td>
</tr>
<tr>
<td></td>
<td>Number of points</td>
</tr>
<tr>
<td></td>
<td>Port powers</td>
</tr>
</tbody>
</table>
**File Type Descriptions and Recall**

The following describes each file type, and what occurs when the file type is recalled.

***.sta files**

- Contain ONLY instrument state information.
- When recalled, they always replace the current instrument state immediately.

***.cst files**

- Contain BOTH instrument state and a LINK to the Cal Sets.
- The **quickest and most flexible** method of saving and recalling a calibrated instrument state.

- Channels need not have cal data to save as .cst file.

- When recalled, the state information is loaded first. Then the PNA tries to **apply a Cal Set** as you would do manually. If the stimulus settings are different between the instrument state and the linked Cal Set, the usual choice is presented (**see Cal Sets**). If the linked Cal Set has been deleted, a message is displayed, but the state information remains in place.

- Because only a link to the Cal Set is saved, the Cal Set can be shared with other measurements.

***.cal files**

- Contain ONLY Cal Set information.

- When recalled, the Cal Set is NOT automatically applied. Apply the calibration data to a channel as you would **apply any Cal Set**.

- **Learn about Recalling**

***.csa files**

- Contain ALL instrument state and the actual Cal Set; not a link to the Cal Set.
- The **safest** method of saving and recalling a calibrated instrument state. However, the file size is larger than a *cst file, and the save and recall times are longer. In addition, because the actual Cal Set is saved, it is very difficult to share the cal data with other measurements.

- Channels need not be calibrated to save as .cst file.

- The Cal Set that is saved could have been a [Cal Register or a User Cal Set](#).

- [Learn about Recalling](#)

---

**Note:** *.pcs* files are the internal file format the PNA uses for storing cal sets. There is no reason for users to access or copy these files.

### Recalling Cal Sets

The calsets listed above will overwrite existing calsets.

To continue, press **OverWrite**
To create duplicates with unique names, press **Duplicate**
To cancel this operation, press **Cancel**.
Recalling Cal Sets dialog box help

Both .cal and .csa file types contain whole Cal Sets. When these file types are recalled, the PNA checks to see if the incoming Cal Set GUID matches an existing PNA Cal Set GUID. If it does, and if the rest of the Cal Set contents are different in any way, then both of these Cal Sets can NOT coexist in the PNA and you are offered the following choices.

Because all PNA channels are saved, there could be more than one Cal Set in either of these file types.

**Overwrite** The incoming Cal Set will replace the existing Cal Set.

**Duplicate** (Only available with .cal recalls.) Because the Cal Set is not automatically applied, you can choose to apply either the original or duplicate Cal Set. The original Cal Set remains in the .cal file.

**Cancel** Abandon the recall operation.

The PNA will offer a choice as described in each file type below. Learn more about Cal Sets.

---

Measurement Data Files

Measurement data is saved as ASCII file types for use in a spreadsheet or CAE programs.

**Note:** Before saving measurement data, always trigger a single measurement, and then allow the PNA channel to go into Hold. This ensures that the entire measurement trace is saved.

The following three file types are used by the PNA. You can select the content and the format of *.SnP files and *.cti files through the Define Data Saves dialog box.

- *.prn files
- *.sNp (Touchstone)
- *.cti (Citifile)
- *.csv

### *.prn Files

Prn files have the following attributes:

- Comma-separated data which can be read into rows and columns by spreadsheet software, such as Microsoft® Excel. To avoid the "delimiting" dialog boxes, change the filename extension from .prn to .csv. Then open directly into Microsoft Excel.

- Contain formatted and corrected stimulus and response data for the current active trace ONLY.

- Are Output only - they cannot be read by the analyzer.

- Beginning with Rev 6.2, FCA and Cal Set Viewer data can be saved to *.prn files

Example:
"S₁₁ Log Mag"

"Frequency (Hz)" , "dB"

3.000000e+005 , -3.528682e+001 ,
4.529850e+007 , -2.817913e+001 ,
9.029700e+007 , -3.216808e+001 ,
1.352955e+008 , -3.101017e+001 ,

.sNp Format (*.s₁p, *.s₂p, *.s₃p, *.s₄p, and so forth)

This file format is used by CAE programs such as Agilent's Microwave Design System (MDS) and Advanced Design System (ADS).

Note: Frequency Converter Application .S2P files are saved using a different method. See Using FCA, Save Data.

- .sNp data is Output only; it can ONLY be read by the PNA embed/de-embed functions.
- .sNp data can be saved in various formats. See Define Data Saves
- The amount of data that is saved depends on the file type that you specify and the amount of data that is available:
  - To save sNp data with an external test set enabled, at the File, Save As dialog, select Snp File(*.sⁿp), then complete the "Choose Ports " dialog.

<table>
<thead>
<tr>
<th>File Type</th>
<th># of Ports</th>
<th># of S-parameters saved</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.s₁p</td>
<td>1</td>
<td>1 S-parameter</td>
</tr>
<tr>
<td>*.s₂p</td>
<td>2</td>
<td>4 S-parameters</td>
</tr>
<tr>
<td>*.s₃p</td>
<td>3</td>
<td>9 S-parameters</td>
</tr>
<tr>
<td>*.s₄p</td>
<td>4</td>
<td>16 S-parameters</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>*.sNp</td>
<td>N</td>
<td>N^2 S-parameters</td>
</tr>
</tbody>
</table>

.sNp data is generally used to gather all S-parameters for a fully corrected measurement. The PNA uses the data that is available on the channel of the active measurement.

- If correction is applied, then valid data is returned for all corrected s-parameters.
- If requesting less data then is available, the Choose ports for sNp data dialog appears. Previous to PNA release 6.2, data was returned beginning with the first calibrated ports until your request if fulfilled.
If correction is NOT applied, the PNA returns as much applicable raw data as possible using S-parameter measurements on the selected channel. Data that is not available is zero-filled. For example, if correction is NOT applied and the active measurement is S11, and an S21 measurement also exists on the channel, then data is returned for the S11 and S21 measurements. Data for S12 and S22 is not available and therefore returned as zeros.

**IMPORTANT** - ALL valid data is saved using the same format and settings (trace math, offset, delay, and so forth) as the active measurement. This can cause the data that is saved for the non-active measurements to be dramatically different from the data that is displayed. For example, when saving an S2P file, if the active S11 measurement is set to Data/Mem (data divided by memory), then ALL 4 S-parameters are saved using Data/Mem. The memory trace that is used in the Data/Mem operation is the same as that used in the active (S11) measurement.

Before saving measurement data, always trigger a single measurement, and then allow the PNA channel to go into Hold. This ensures that the entire measurement trace is saved.

**.sNp Data Output**

.sNp files contain header information, stimulus data, a response data pair for EACH S-parameter measurement. The only difference between .s1p, s2p, and so forth, is the number of S-parameters that are saved.

The following is a sample of **Header information**:

```
!Agilent Technologies,E8362B,US42340026,Q.03.54
!Agilent E8362B: Q.03.54
!Date: Friday, April 25, 2003 13:46:41
!Correction: S11(Full 2 Port SOLT,1,2) S21(Full 2 Port SOLT,1,2) S12(Full 2 Port SOLT,1,2) S22(Full 2 Port SOLT,1,2)
!S2P File: Measurements:S11,S21,S12,S22:
# Hz  S RI R 50
```

**Note:** Although the following shows Real / Imag pairs, the format could also be LogMag / Phase or LinMag / Phase.

**.s1p Files**

Each record contains 1 stimulus value and 1 S-parameter (total of 3 values)

Stim  Real(Sxx)  Imag(Sxx)

**.s2p Files**

Each record contains 1 stimulus value and 4 S-parameters (total of 9 values)

Stim  Real(S11)  Imag(S11)  Real(S21)  Imag(S21)  Real(S12)  Imag(S12)  Real(S22)  Imag(S22)

**.s3p Files**

Each record contains 1 stimulus value and 9 S-parameters (total of 19 values)

Stim  Real(S11)  Imag(S11)  Real(S12)  Imag(S12)  Real(S13)  Imag(S13)  Real(S21)  Imag(S21)  Real(S22)  Imag(S22)  Real(S12)  Imag(S12)  Real(S13)  Imag(S13)  Real(S21)  Imag(S21)  Real(S22)  Imag(S22)  Real(S31)  Imag(S31)  Real(S32)  Imag(S32)  Real(S33)  Imag(S33)

**.s4p Files (and so forth...)**

Each record contains 1 stimulus value and 16 S-parameters (total of 33 values)

Stim  Real(S11)  Imag(S11)  Real(S12)  Imag(S12)  Real(S13)  Imag(S13)  Real(S14)  Imag(S14)  Real(S21)  Imag(S21)  Real(S22)  Imag(S22)  Real(S23)  Imag(S23)  Real(S24)  Imag(S24)

```
Choose ports for SNP File dialog box help

This dialog appears when selecting File, Save As, Trace sNp, and you request less data than is available, or you want data for more than 4 ports. This dialog allows you to choose which S-parameter data to save.

**Number of ports**  Select the number of ports for which data will be saved.

**Arrow buttons**  Click to Add and Remove ports for the following columns:

- **Available Ports**  The PNA / External test set ports. There may NOT be valid data available for all of these ports. Learn more.
- **Chosen Ports**  When OK is clicked, sNp data is saved for these ports.

**OK**  Becomes available when the number of Chosen ports = the Number of ports to save. Click to save to sNp file.

With **Number of ports** = 2, .s2p data is saved; with **Number of ports** = 3, .s3p data is saved, and so forth. Learn more about sNp files

.cti CitiFiles

Citifile format is compatible with the Agilent 8510 Network Analyzer and Agilent's Microwave Design System (MDS).

**Note:** Before saving measurement data, always trigger a single measurement, and then allow the PNA channel to go into Hold. This ensures that the entire measurement trace is saved.

You can do the following using citifiles:

- save the active trace, or all traces. (see Define Data Saves
- save formatted or unformatted citifile data
Save Formatted data

1. Set the format using Define Data Saves.
2. Click File then Save As
3. Select Citifile Formatted Data (*.cti)

FCA, GCA, and NFA traces can NOT be saved as formatted Citifile data.
On the data access map, Formatted data is taken from location 2 or 4.

Save Unformatted data

1. Click File Save As
2. Select Citifile Data Data (*.cti)

On the data access map, Unformatted data is taken from the block just before Format.

Recalling Citifiles into the PNA

- To recall citifiles, click File then Recall. Specify (*.cti)
- Recalled citifile data is ALWAYS displayed on the PNA using LogMag format, regardless of how the file was stored.

Citifile traces are recalled into the same window / channel configuration as when they were saved. However, the new recalled channel numbers begin with channel 32 and decrement for each additional channel.

For example, when a citifile is saved, two traces are in window 1, channel 1 and two additional traces are in window 2, channel 2. When recalled into a factory preset condition (1 trace in window 1, channel 1), the first two recalled traces appear in window 2, channel 32, and the second two traces appear in window 3 channel 31.

If a channel is in use, you are prompted to create a new channel.

- Yes - skip down to the next available channel.
- No - add recalled data to the existing channel.

See also Traces, Channels, and Windows on the PNA

*.csv Files

This file format is available ONLY for saving 2D Gain Compression data. This data type can be read by spreadsheet programs, such as Microsoft Excel. Learn about Gain Compression App (Opt 086).

Note: Before saving measurement data, always trigger a single measurement, and then allow the PNA channel to go into Hold. This ensures that the entire measurement trace is saved.

When a 2D Gain Compression trace is active, the following is saved:
The data are organized by frequency regardless of the 2D method used to acquire the data. The above image shows 5 power points at each frequency. For 201 frequency points and 5 power points, there are 1005 rows of data.

If calibration is turned on when the file is saved, then all data are calibrated. Otherwise, raw data is saved.

**Define Data Saves**

**How to select Define Data Saves**

Using front-panel HARDKEY [softkey] buttons

Using a mouse with PNA Menus

**For N5230A and E836xA/B models**

1. Navigate using MENU/DIALOG

**For PNA-X and 'C' models**

1. Press SAVE
2. then [Define Data Saves]
Define Data Saves dialog box help

**CitiFile Contents**  Determines what is saved to a .cti file.

- **Auto** - Saves the active trace. Additional traces are saved if correction is ON and Full 2-port or Full 3-port calibration is performed. For Full 2-port calibration, 4 traces are saved. For Full 3-port calibration, 9 traces are saved.

- **Single Trace** - Saves the active trace in the currently selected window.

- **Displayed Traces** - Saves all displayed data traces

**Citifile Formatted Data**

- **Auto** - Data is saved in LogMag or LinMag if one of these is the currently selected display format. If format is other than these, then data is saved in Real/Imag.

- **LogMag, LinMag, Real/Imag** - Select output format.

  The imaginary portion for all *LogMag* and *LinMag* data is output in degrees.

**SnP Formatted Data**  (.s1p, .s2p, .s3p) [Learn more about SnP files.]

- **Auto** - Data is saved in LogMag or LinMag if one of these is the currently selected format. If format is other than these, then data is saved in Real/Imag.

- **LogMag, LinMag, Real/Imag** - Select output format.

  The imaginary portion for all *LogMag* and *LinMag* data is output in degrees.

---

**Manage Files without a Mouse**

The Manage Files dialog box is designed to be used from the front panel. It performs the same function as Windows Explorer, but can be used without the use of a mouse or keyboard.

[Learn more about using the Front-panel interface.]


Manage Files dialog box help

- **Recall**  Opens a Network Analyzer file already stored in memory.
- **Rename**  Renames a file that is selected in the open folder.
- **Delete**  Removes a selected file from the open folder.
- **Delete All**  Removes all files of the file type selected that appear in the open folder.
- **New folder**  Create a new folder and give it a name

Last modified:

- 17-Oct-2007  Added note for MM
- 24-Jul-2007  Added clarification to define data save
- 10/23/06  Added pcs note
- 9/18/06  MQ Added choose ports for snp
- 9/12/06  Added link to programming commands
Drive Mapping

Drive mapping allows you to share disk drives between the PNA and an external computer. You can either map from the PNA, or from your PC, to the other.

- **From the PNA, map to a drive on an External PC**
- **From an External PC, map to a drive on the PNA**

To prepare for Drive Mapping:

1. Both the PC and PNA must be connected to a shared computer network
2. You must know the full computer name of the PC (or analyzer) you are mapping TO. **Tell me how**
3. Your logon and password on the analyzer must be the same as that on the external PC. You can add your PC logon to the analyzer. **Tell me how**

**Note:** These procedures require a mouse and keyboard. Also, the external PC must have Windows NT 4.0 (or later).

**From the Analyzer, map to a drive on the External PC**

1. On the external computer desktop, go to **Windows Explorer**. In the listing of drives, right click on the drive you want to share. Click **Sharing**.
2. In the dialog box, select **Shared As**. In the **Share Name** box, use the arrow key or type in a share name for the drive. For example: C$. Click **OK**.
3. On the analyzer desktop, click **Windows Explorer**. From the **Tools** menu, click **Map Network Drive**. (To get to the analyzer desktop, click **View**, then click **Title Bars**)
4. If you would like to connect to your external PC using a different logon, click **Connect using a different Logon**. This logon must be registered on the analyzer and you must be currently logged on the external PC using this logon.
   1. In the **Connect as** box, type your logon name. The logon name and password must be exactly the same on both the external PC and the analyzer.
   2. In the **Password** box, type the logon password that you use on the external computer. Click **OK**. The logon name and password must be exactly the same on both the external PC and the analyzer.
5. In the **Folder** box, type \(\text{full computer name of analyzer}\)\text{share name (from step 2). (For example: }\text{SLT1234}\text{C$ )}
6. Click **Finish**.
From an External PC, map to a drive on the Analyzer

1. On the analyzer desktop, click **Windows Explorer**. Right click on the drive you want to share. Click on **Sharing**.

2. In the dialog box, select **Shared this folder**. In the **Share Name** box, type in a share name for the drive. For example: **C$**. Click **OK**.

3. On the external PC desktop, click **Windows Explorer**. From the **Tools** menu, click **Map Network Drive**.

4. If the current logon on your PC is different from the current logon on the analyzer, click **Connect using a different Logon** to connect to using the current analyzer logon. This logon must be registered on the external PC. To see the current logon on either the PC or analyzer, hold **Ctrl - Alt**, and press **Delete**.
   
   1. In the **Connect as** box, type the logon currently being used by the analyzer.
   
   2. In the **Password** box, type the logon password that you use on the external computer. Click **OK**

5. In the **Folder** box, type **\computername (prep1)\share name** (from step 2). (For example: **\SLT1234\C$**)

6. Click **Finish**.
Print a Displayed Measurement

The analyzer allows you to print a displayed measurement to a printer or to a file. The printer can be either networked or local.

- Connecting a Printer
- Printing

Other Outputting Data topics

Connecting a Printer

You can connect your printer to the PNA using three different connector types:

- Parallel connector
- Serial connector
- USB

In addition to connecting the printer directly to the PNA via one of the above interfaces, LAN connected printers are also usable by the PNA.

Note: Early PNAs have a Centronics connector for connecting a printer. An adapter (36-pin male - 1284-C - to 25-pin female) was shipped with those PNAs to allow connection with a standard parallel printer cable.

CAUTION: Do NOT connect your printer to the 25-pin female port labeled Ext. Test Set Interface. Voltage levels of signal lines may damage the printer's I/O.

To Add a Printer

Note: If you try to print from the PNA application and the Add Printer Wizard appears, click Cancel and add the printer using the following procedure.

1. From the PNA application, click View then click Minimize Application
2. On the Windows taskbar, click Start, point to Settings, then click Printers.
3. Double-click Add Printer.
4. Follow the instructions in the Add Printer Wizard.

For more information, refer to Microsoft Windows Help or your printer documentation.

Printing

- Print a Hardcopy
The measurement information on the screen can be printed to any local or networked printer that is connected to the PNA. The graphic below shows an example of how a screen-capture image appears when printed. The Page Setup settings allows you to customize the printed form of the measurement information.

---

**How to Print a Hardcopy**

**Using front-panel HARDKEY [softkey] buttons**

<table>
<thead>
<tr>
<th>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press</td>
</tr>
</tbody>
</table>

**Using a mouse with PNA Menus**

| 1. Click **File**                     |
| 2. then **Print**                     |

For **PNA-X** and **‘C’ models**
1. Press Print
2. then [Print]

1. Click File
2. then Print
3. then Print

Note: For information on the choices in the Print dialog box, see Windows Help.

Page Setup

The Page Setup dialog allows flexibility in the appearance that measurement data is printed. After setting up the page, click File, then Print... to obtain a hard-copy.

How to select Page Setup

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click File</td>
</tr>
<tr>
<td></td>
<td>2. then Page Setup</td>
</tr>
<tr>
<td><strong>For PNA-X and ‘C’ models</strong></td>
<td></td>
</tr>
</tbody>
</table>

Programming Commands
1. Press **PRINT**
2. then **[Page Setup]**

1. Click **File**
2. then **Print**
3. then **Print Options**

---

### Page Setup dialog box help

**Note:** See Windows Help for information on the choices on the left side of this dialog.

#### Windows

- **Minimum vertical size** Adjust to change the amount of a page that the measurement window fills. The adjustment range is from 40 to 100%.
- **One window per page** Check to print one window per page. Clear to print all selected windows without a forced page break.
- **Only active window** Check to print only the active window. Clear to print all windows.
- **Agilent logo** Check to print the Agilent logo to the header.
- **Data and Time** Check to add the current date and time to the header.
- **Global Pass/Fail** Check to add the Global Pass/Fail status to the header.
- **Page Numbers** Check to add page numbers (1 of n) to the header.

#### Channel Settings Table

- **Print** Check to print the channel settings table.
- **Expand segment data** Check to print segment sweep data.

#### Trace Attributes Table

- **Print** Check to print the Trace Attributes Table. The Trace Attributes are measurement type, correction factors ON or OFF, smoothing, options, and marker details. The Trace Attributes are listed by Trace ID# for each window.
Each Trace ID# can have multiple entries depending on the number of markers associated with the trace. The marker details are marker number, position and response. If there are multiple markers on a trace, the trace attributes are only shown for the first marker. However, the trace attributes for the first marker apply to all other markers on that trace.

The options column can have one or more options. D for Delay, M for Marker, G for Gating. Multiple options selected would appear as follows: DMG.

**Print marker data**  Check to print all marker data. The amount of data depends on how many markers are created.

---

### Print to a File

The analyzer can save a screen-capture image in any of the following formats:

- **.png** (preferred format)
- **.bmp** (bitmap)
- **.jpg**

The analyzer automatically saves the file to the current path. If not previously defined, the analyzer automatically selects the default path C:/Program Files/Agilent/Network Analyzer/Documents/.

A .bmp file, like a .prn file, can be imported into software applications such as Microsoft Excel, Word, or Paint to display a screen-capture image.

See Save and Recall files for more information.

---

### How to Print to a File

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click File</td>
</tr>
<tr>
<td></td>
<td>2. then Print to File</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press PRINT</td>
<td>1. Click File</td>
</tr>
<tr>
<td>2. then [Print to File]</td>
<td>2. then Print</td>
</tr>
<tr>
<td></td>
<td>3. then Print to File</td>
</tr>
</tbody>
</table>

---

Last modified:
PNA Application Notes

The following links require an Internet connection.

Note: Check out the multimedia PNA Demo presentations, including ‘Network Analyzer Basics’.

Calibrations

Improving Measurement and Calibration Accuracy Using the Frequency Converter Application (5988-9642EN)
On-Wafer Calibration Using a 4-port, 20 GHz PNA-L Network Analyzer (N5230A Option 240/245) (5989-2287EN)

ECal

Agilent Electronic vs. Mechanical Calibration Kits: Calibration Methods and Accuracy (5988-9477EN)
User Characterization: Electronic Calibration Feature Allows Users to Customize to Specific Needs (5988-9478EN)

Embedding / De-embedding

De-embedding and Embedding S-Parameter Networks Using a Vector Network Analyzer (5980-2784EN)

Amplifier Measurements

High-power measurements using the PNA (5989-1349EN)
Amplifier Linear and Gain Measurements (5988-8644EN)
Amplifier Swept-Harmonic Measurements (5988-9473EN)
Amplifier and CW Swept Intermodulation-Distortion Measurements (5988-9474EN)

Antenna Measurements

Triggering PNA Microwave Network Analyzers for Antenna Measurements (5988-9518EN)
New Network Analyzer Methodologies in Antenna/RCS Measurements (5989-1937EN)
Pulsed Antenna Measurements Using PNA Network Analyzers (5989-0221EN)

Balanced Measurements (Although the following refer to the ENA, they are also relevant to the PNA.)
On-wafer Balanced Component Measurement with the Cascade Microtech Probing System (5988-5886EN)
Network De-embedding/Embedding and Balanced Measurement (5988-4923EN)

Mixer Measurements

Mixer Transmission Measurements Using the Frequency Conversion Application (5988-8642EN)
Mixer Conversion-Loss and Group Delay Measurement Techniques and Comparisons (5988-9619EN)
Comparison of Mixer Characterization using New Vector Characterization Techniques (5988-7827EN)
Novel Method for Vector Mixer Characterization and Mixer Test System Vector Error Correction (5988-7826EN)
Measuring Absolute Group Delay of Multistage Converters Using PNA Microwave Network Analyzers (5989-0219EN)

Pulsed Measurements
Pulsed-RF S-Parameter Measurements Using Wideband and Narrowband Detection (AN 1408-12)
Accurate Pulsed Measurements (5989-0563EN)
Pulsed Antenna Measurements Using PNA Network Analyzers (5989-0221EN)

Other Measurements

New Time Domain Analysis
Using the PNA Series to Analyze Lightwave Components (5989-3385EN)
Using the PNA for Banded Millimeter-Wave Measurements (5989-4098EN)
PNA MM-Wave Network Analyzers: Analysis of Cable Length on VNA System Performance (5989-1941EN)
Basics of Measuring the Dielectric Properties of Materials (5989-2589EN)

Automation
Connectivity Advances for Component Manufacturers (5980-2782EN)
Introduction to Application Development using the PNA (5980-2666EN)
The 'Need for Speed' in Component Manufacturing Test (5980-2783EN)
Network Analyzer Basics

This self-paced two hour video discusses the basic concepts of Network Analysis.

The files are installed and should work on older PNA models. If the PNA link does not work, then use the internet link, which requires an internet connection.

- **From the PNA:** Proceed with Network Analyzer Basics.
- **From the Internet:** [http://wireless.agilent.com/networkanalyzers/pnademo.htm](http://wireless.agilent.com/networkanalyzers/pnademo.htm) in both streaming and downloadable format.

Last modified:

10/18/06 Added link to pnademo.
**Connector Care**

Proper connector care is critical for accurate and repeatable measurements. The following information will help you preserve the precision and extend the life of your connectors - saving both time and money.

- **Connector Care Quick Reference Guide**
- **Connector Cleaning Supplies**
- **Safety Reminders**
- **About Connectors**
- **Gaging Fundamentals**
- **Connector Care Procedures**

See also mmWave Connector Care at [http://na.tm.agilent.com/pna/connectorcare/Connector_Care.htm](http://na.tm.agilent.com/pna/connectorcare/Connector_Care.htm)

### Preventing Test Port Connector Damage

#### Handling and Storing Connectors

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keep connectors clean</td>
<td>Touch mating-plane surfaces</td>
</tr>
<tr>
<td>Protect connectors with plastic end caps</td>
<td>Set connectors contact-end down</td>
</tr>
<tr>
<td>Keep connector temperature same as analyzer</td>
<td>Store connectors loose in box or drawer</td>
</tr>
</tbody>
</table>

#### Visual Inspection

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inspect connectors with magnifying glass.</td>
<td>Use a connector with a bent or broken center conductor</td>
</tr>
<tr>
<td>Look for metal debris, deep scratches or dents</td>
<td>Use a connector with deformed threads</td>
</tr>
</tbody>
</table>

#### Cleaning Connectors

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clean surfaces first with clean, dry compressed air</td>
<td>Use high pressure air (&gt;60 psi)</td>
</tr>
<tr>
<td>Use lint-free swab or brush</td>
<td>Use any abrasives</td>
</tr>
<tr>
<td>Use minimum amount of alcohol</td>
<td>Allow alcohol into connector support beads</td>
</tr>
<tr>
<td>Clean outer conductor mating surface and threads</td>
<td>Apply lateral force to center conductor</td>
</tr>
</tbody>
</table>
Gaging Connectors

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inspect and clean gage, gage master and device tested</td>
<td>Use an out of specification connector</td>
</tr>
<tr>
<td>Use correct torque wrench</td>
<td>Hold connector gage by the dial</td>
</tr>
<tr>
<td>zero gage before use</td>
<td></td>
</tr>
<tr>
<td>Use multiple measurements and keep record of readings</td>
<td></td>
</tr>
</tbody>
</table>

Making Connections

<table>
<thead>
<tr>
<th>Do</th>
<th>Do Not</th>
</tr>
</thead>
<tbody>
<tr>
<td>Align connectors first</td>
<td>Cross thread the connection</td>
</tr>
<tr>
<td>Rotate only the connector nut</td>
<td>Twist connector body to make connection</td>
</tr>
<tr>
<td>Use correct torque wrench</td>
<td>Mate different connector types</td>
</tr>
</tbody>
</table>

Connector Care and Cleaning Supplies

<table>
<thead>
<tr>
<th>Description</th>
<th>Web Site</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lint Free Cloths- Air dusters</td>
<td><a href="http://www.ccrwebstore.com">http://www.ccrwebstore.com</a></td>
</tr>
<tr>
<td>Isopropyl</td>
<td><a href="http://www.techspray.com">http://www.techspray.com</a></td>
</tr>
<tr>
<td>Nitrilite Gloves and Finger Cots</td>
<td><a href="http://www.techni-tool.com">http://www.techni-tool.com</a></td>
</tr>
</tbody>
</table>

Safety Reminders

When cleaning connectors:

- Always use protective eyewear when using compressed air or nitrogen.
- Keep isopropyl alcohol away from heat, sparks and flame. Use with adequate ventilation. Avoid contact with eyes, skin and clothing.
- Avoid electrostatic discharge (ESD). Wear a grounded wrist strap (having a 1 MΩ series resistor) when cleaning device, cable or test port connectors.
- Cleaning connectors with alcohol shall only be done with the instruments power cord removed, and in a well-ventilated area. Allow all residual alcohol moisture to evaporate, and the fumes to dissipate prior to energizing the instrument.

About Connectors
Connector Service Life

Even though calibration standards, cables, and test set connectors are designed and manufactured to the highest standards, all connectors have a limited service life. This means that connectors can become defective due to wear during normal use. For best results, all connectors should be inspected and maintained to maximize their service life.

Visual Inspection should be performed each time a connection is made. Metal particles from connector threads often find their way onto the mating surface when a connection is made or disconnected. See Inspection procedure.

Cleaning the dirt and contamination from the connector mating plane surfaces and threads can extend the service life of the connector and improve the quality of your calibration and measurements. See Cleaning procedure.

Gaging connectors not only provides assurance of proper mechanical tolerances, and thus connector performance, but also indicate situations where the potential for damage to another connector may exist. See Gaging procedure.

Proper connector care and connection techniques yield:

- Longer Service Life
- Higher Performance
- Better Repeatability

Connector Grades and Performance

The three connector grades (levels of quality) for the popular connector families are listed below. Some specialized types may not have all three grades.

- **Production** grade connectors are the lowest grade and the least expensive. It is the connector grade most commonly used on the typical device under test (DUT). It has the lowest performance of all connectors due to its loose tolerances. This means that production grade connectors should always be carefully inspected before making a connection to the analyzer. Some production grade connectors are not intended to mate with metrology grade connectors.

- **Instrument** grade is the middle grade of connectors. It is mainly used in and with test instruments, most cables and adapters, and some calibration standards. It provides long life with good performance and tighter tolerances. It may have a dielectric supported interface and therefore may not exhibit the excellent match of a metrology grade connector.

- **Metrology** grade connectors have the highest performance and the highest cost of all connector grades. This grade is used on calibration standards, verification standards, and precision adapters. Because it is a high precision connector, it can withstand many connections and disconnections and, thus, has the longest life of all connector grades. This connector grade has the closest material and geometric specifications. Pin diameter and pin depth are very closely specified. Metrology grade uses an air dielectric interface and a slotless female contact which provide the highest performance and traceability.
**Note:** In general, Metrology grade connectors should not be mated with Production grade connectors.

**Adapters as Connector Savers**

Make sure to use a high quality (Instrument grade or better) adapter when adapting a different connector type to the analyzer test ports. It is a good idea to use an adapter even when the device under test is the same connector type as the analyzer test ports. In both cases, it will help extend service life, and protect the test ports from damage and costly repair.

The adapter must be fully inspected before connecting it to the analyzer test port and inspected and cleaned frequently thereafter. Because calibration standards are connected to the adapter, the adapter should be the highest quality to provide acceptable RF performance and minimize the effects of mismatch.

**Connector Mating Plane Surfaces**

An important concept in RF and microwave measurements is the reference plane. For a network analyzer, this is the surface that all measurements are referenced to. At calibration, the reference plane is defined as the plane where the mating plane surfaces of the measurement port and the calibration standards meet. Good connections (and calibrations) depend on perfectly flat contact between connectors at all points on the mating plane surfaces (as shown in the following graphic).

![Mating Plane Surfaces](image)

**Gaging Fundamentals**

Connector gages are important tools used to measure center conductor pin depth in connectors. Connector pin depth, measured in terms of recession or protrusion, is generally the distance between the mating plane and the end of the center conductor, or the shoulder of the center conductor for a stepped male pin.

**Typical Connector Gage**
Recession and Protrusion

Pin depth is negative (recession) if the center conductor is recessed below the outer conductor mating plane, usually referred to as the "reference plane". Pin depth is positive (protrusion) if the center conductor projects forward from the connector reference plane.

Pin Depth

1. Recession of female contact
2. Recession of male pin shoulder

Difference with Type-N Connectors
Type-N connectors have the mating plane of the center conductors offset from the connector reference plane. In this case the zero setting "gage masters" generally offset the nominal distance between the center conductor mating plane and the connector reference plane.

**When to Gage Connectors**

- Before using a connector or adapter the first time.
- When visual inspection or electrical performance suggests the connector interface may be out of range.
- After every 100 connections, depending on use.

**Connector Gage Accuracy**

Connector gages (those included with calibration and verification kits), are capable of performing coarse measurements only. This is due to the repeatability uncertainties associated with the measurement. It is important to recognize that test port connectors and calibration standards have mechanical specifications that are extremely precise. Only special gaging processes and electrical testing (performed in a calibration lab) can accurately verify the mechanical characteristics of these devices. The pin depth specifications in the Agilent calibration kit manuals provide a compromise between the pin depth accuracy required, and the accuracy of the gages. The gages shipped with calibration and verification kits allow you to measure connector pin depth and avoid damage from out-of-specification connectors.

**Note:** Before gaging any connector, the mechanical specifications provided with that connector or device should be checked.

**To Gage Connectors**

1. Wear a grounded wrist strap (having a 1 M series resistor).
2. Select proper gage for device under test (DUT).
3. Inspect and clean gage, gage master, and DUT.
4. Zero the connector gage.
   a. While holding gage by the barrel, carefully connect gage master to gage. Finger-tighten connector nut only.
   b. Use proper torque wrench to make final connection. If needed, use additional wrench to prevent gage master (body) from turning. Gently tap the barrel to settle the gage.
   c. The gage pointer should line up exactly with the zero mark on gage. If not, adjust "zero set" knob until gage pointer reads zero. On gages having a dial lock screw and a movable dial, loosen the dial lock screw and move the dial until the gage pointer reads zero. Gages should be zeroed before each set of measurements to make sure zero setting has not changed.
   d. Remove gage master.
5. Gage the device under test.
   a. While holding gage by the barrel, carefully connect DUT to gage. Finger-tighten connector nut only.
   b. Use proper torque wrench to make final connection and, if needed, use additional wrench to prevent DUT (body) from turning. Gently tap the barrel to settle the gage.
c. Read gage indicator dial for recession or protrusion and compare reading with device specifications.

**Caution:** If the gage indicates excessive protrusion or recession, the connector should be marked for disposal or sent out for repair.

6. For maximum accuracy, measure the device a minimum of three times and take an average of the readings. After each measurement, rotate the gage a quarter-turn to reduce measurement variations.

7. If there is doubt about measurement accuracy, be sure the temperatures of the parts have stabilized. Then perform the cleaning, zeroing, and measuring procedure again.

**Connector Care Procedures**

- **Inspecting Connectors**
- **Cleaning Connectors**
- **Making Connections**
- **Using a Torque Wrench**
- **Handling and Storing Connectors**

**To Inspect Connectors**

Wear a grounded wrist strap (having a 1 MΩ series resistor).

Use a magnifying glass (≥10X) and inspect connector for the following:

- Badly worn plating or deep scratches
- Deformed threads
- Metal particles on threads and mating plane surfaces
- Bent, broken, or mis-aligned center conductors
- Poor connector nut rotation

**Caution:** A damaged or out-of-specification device can destroy a good connector attached to it even on the first connection. Any connector with an obvious defect should be marked for disposal or sent out for repair.

**To Clean Connectors**

1. Wear a grounded wrist strap (having a 1 MΩ series resistor).

2. Use clean, low-pressure air to remove loose particles from mating plane surfaces and threads. Inspect connector thoroughly. If additional cleaning is required, continue with the following steps.

4. Clean contamination and debris from mating plane surfaces and threads. When cleaning interior surfaces, avoid exerting pressure on center conductor and keep swab fibers from getting trapped in the female center conductor.

5. Let alcohol evaporate—then use compressed air to blow surfaces clean.

6. Inspect connector. Make sure no particles or residue remains.

7. If defects are still visible after cleaning, the connector itself may be damaged and should not be used. Determine the cause of damage before making further connections.

To Make Connections

1. Wear a grounded wrist strap (having a 1 MΩ series resistor).

2. Inspect, clean, and gage connectors. All connectors must be undamaged, clean, and within mechanical specification.

3. Carefully align center axis of both devices. The center conductor pin—from the male connector—must slip concentrically into the contact finger of the female connector.

4. Carefully push the connectors straight together so they can engage smoothly. Rotate the connector nut (not the device itself) until finger-tight, being careful not to cross the threads.
5. Use a torque wrench to make final connection. Tighten until the "break" point of the torque wrench is reached. Do not push beyond initial break point. Use additional wrench, if needed, to prevent device body from turning.

To Separate a Connection

1. Support the devices to avoid any twisting, rocking or bending force on either connector.
2. Use an open-end wrench to prevent the device body from turning.
3. Use another open-end wrench to loosen the connector nut.
4. Complete the disconnection by hand, turning only the connector nut.
5. Pull the connectors straight apart.

To Use a Torque Wrench

1. Make sure torque wrench is set to the correct torque setting.
2. Position torque wrench and a second wrench (to hold device or cable) within 90° of each other before applying force. Make sure to support the devices to avoid putting stress on the connectors.
2. **Hold**

Correct Method

INCORRECT METHOD (Too Much Lift)

3. Hold torque wrench lightly at the end of handle—then apply force perpendicular to the torque wrench handle. Tighten until the "break" point of the torque wrench is reached. Do not push beyond initial break point.

**Torquing Direction**

STOP WHEN HANDLE BEGINS TO YIELD

**To Handle and Store Connectors**

- Install protective end caps when connectors are not in use.
- Never store connectors, airlines, or calibration standards loose in a box. This is a common cause of connector damage.
- Keep connector temperature the same as analyzer. Holding the connector in your hand or cleaning connector with compressed air can significantly change the temperature. Wait for connector temperature to stabilize before using in calibration or measurements.
- Do not touch mating plane surfaces. Natural skin oils and microscopic particles of dirt are difficult to remove from these surfaces.
- Do not set connectors contact-end down on a hard surface. The plating and mating plane surfaces can be damaged if the interface comes in contact with any hard surface.
- Wear a grounded wrist strap and work on a grounded, conductive table mat. This helps protect the analyzer and devices from electrostatic discharge (ESD).
Protection against electrostatic discharge (ESD) is essential while removing or connecting cables to the network analyzer. Static electricity can build up on your body and can easily damage sensitive internal circuit elements when discharged. Static discharges too small to be felt can cause permanent damage. To prevent damage to the instrument:

- **Always** have a grounded, conductive table mat in front of your test equipment.

- **Always** wear a grounded wrist strap, connected to a grounded conductive table mat, having a 1 MO resistor in series with it, when making test setup connections.

- **Always** wear a heel strap when working in an area with a conductive floor. If you are uncertain about the conductivity of your floor, wear a heel strap.

- **Always** ground yourself before you clean, inspect, or make a connection to a static-sensitive device or test port. You can, for example, grasp the grounded outer shell of the test port or cable connector briefly.

- **Always** ground the center conductor of a test cable before making a connection to the analyzer test port or other static-sensitive device. This can be done as follows:

  1. Connect a short (from your calibration kit) to one end of the cable to short the center conductor to the outer conductor.

  2. While wearing a grounded wrist strap, grasp the outer shell of the cable connector.

  3. Connect the other end of the cable to the test port and remove the short from the cable.

See [Analyzer Accessories](#) for ESD part numbers.
Absolute Output Power

An absolute output-power measurement displays absolute power versus frequency.

- **What is Absolute Output Power?**
- **Why Measure Absolute Output Power?**
- **Accuracy Considerations**
- **How to Measure Absolute Output Power**

See other Amplifier Parameters topics

**What is Absolute Output Power?**

An absolute-output power measurement displays the power present at the analyzer's input port. This power is absolute-it is not referenced (ratioed) to the incident or source power. In the log mag format, values associated with the grid's vertical axis are in units of dBm, which is the power measured in reference to 1 mW.

- 0 dBm = 1 mW
- -10 dBm = 100 mW
- +10 dBm = 10 mW

In the linear mag format, values associated with the grid's vertical axis are in units of watts (W).

**Why Measure Absolute Output Power?**

Absolute output power is measured when the amplifier's output must be quantified as absolute power rather than a ratioed relative power measurement. For example, during a gain compression measurement, it is typical to also measure absolute output power. This shows the absolute power out of the amplifier where 1-dB compression occurs.

**Accuracy Considerations**

The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:

- Damage the analyzer receiver
- Exceed the input compression level of the analyzer receiver, resulting in inaccurate measurements.

Attenuation of the amplifier's output power can be accomplished using either attenuators or couplers.

The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

**How to Measure Absolute Power**
Do the following to measure absolute output power:

1. Preset the analyzer.
2. Select an unratioed power measurement (receiver B).
3. Set the analyzer's source power to 0 dBm.
4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port-2.
5. Connect the amplifier as shown in the following graphic, and provide the dc bias.

![Amplifier Diagram]

* Direct Connection

6. Select the analyzer settings for your amplifier under test.
7. Remove the amplifier and connect the measurement ports together. Store the data to memory. Be sure to include the attenuator and cables in the test setup if they will be used when measuring the amplifier.
8. Save the instrument state to memory.
9. Reconnect the amplifier.
10. Select the data math function Data/Memory.
11. Scale the displayed measurement for optimum viewing and use a marker to measure the absolute output-power at a desired frequency.
12. Print or save the data to a disk.
AM-PM Conversion

The AM-PM conversion of an amplifier is a measure of the amount of undesired phase deviation (PM) that is caused by amplitude variations (AM) inherent in the system.

- What Is AM-PM Conversion?
- Why Measure AM-PM Conversion
- Accuracy Considerations
- How to Measure AM-PM Conversion

What Is AM-PM Conversion?

AM-to-PM conversion measures the amount of undesired phase deviation (PM) that is caused by amplitude variations (AM) of the system. For example, unwanted phase deviation (PM) in a communications system can be caused by:

Unintentional amplitude variations (AM)

- Power supply ripple
- Thermal drift
- Multipath fading

Intentional modulation of signal amplitude

- QAM
- Burst modulation

AM-to-PM conversion is usually defined as the change in output phase for a 1-dB increment in the power-sweep applied to the amplifier's input (i.e. at the 1 dB gain compression point). It is expressed in degrees-per-dB (°/dB). An ideal amplifier would have no interaction between its phase response and the power level of the input signal.
Why Measure AM-PM Conversion

AM-to-PM conversion is a critical parameter in systems where phase (angular) modulation is used, such as:

- FM
- QPSK
- 16QAM

It is a critical parameter because undesired phase deviation (PM) causes analog signal degradation, or increased bit-error rates (BER) in digital communication systems. While it is easy to measure the BER of a digital communication system, this measurement alone does not help you understand the underlying causes of bit errors. AM-to-PM conversion is one of the fundamental contributors to BER, and therefore it is important to quantify this parameter in communication systems.

Refer to the I/Q diagram below for the following discussion on how AM-to-PM conversion can cause bit errors.
The desirable state change is from the small solid vector to the large solid vector.

With AM-to-PM conversion, the large vector may actually end up as shown with the dotted line. This is due to phase shift that results from a change in the input power level.

For a 64QAM signal as shown (only one quadrant is drawn), we see that the noise circles that surround each state would actually overlap, which means that statistically, some bit errors would occur.

**Accuracy Considerations**

With this method of measuring AM-to-PM conversion, the modulation frequency is approximately the inverse of the sweep time. Even with the fastest power sweep available on most network analyzers, the modulation frequency ends up being fairly low (typically less than 10 Hz). This could cause a slight temperature change as the sweep progresses, especially if the amplifier has low thermal mass, typical of an unpackaged device. Results using this method could differ slightly if the nonlinear behavior of an amplifier is extremely sensitive to thermal changes. (The PNA series analyzers can make power sweeps <1 ms.)

- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

- The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:
  - damage the analyzer receiver
  - exceed the input compression level of the analyzer receiver, resulting in inaccurate measurements

**Attenuation** of the amplifier’s output power can be accomplished using:

- Attenuators
- Couplers

The frequency-response effects of the attenuators and couplers must be accounted for during calibration since they are part of the test system. Proper error-correction techniques can reduce these effects.

- The frequency response is the dominant error in an AM-to-PM conversion measurement setup. Performing a
thru-response measurement calibration significantly reduces this error. For greater accuracy, perform a 2-port measurement calibration.

**How to Measure AM-PM Conversion**

1. Preset the analyzer.
2. Select an S21 measurement in the power-sweep mode.
3. Enter the start and stop power levels for the analyzer's power sweep. The start power level should be in the linear region of the amplifier's response (typically 10-dB below the 1-dB compression point). The stop power should be in the compression region of the amplifier's response.
4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port 2.
5. Connect the amplifier as shown in the following graphic, and provide the dc bias.

![Diagram of amplifier setup](image)

6. Select the analyzer settings for your amplifier under test in order to perform a swept-power gain compression measurement at a chosen frequency. See [Gain Compression](#).
7. Remove the amplifier and perform a measurement calibration. Be sure to include the attenuator and cables in the calibration setup if they will be used when measuring the amplifier.
8. Save the instrument state to memory.
9. Reconnect the amplifier.
10. Use a reference marker to target the amplifier's input power at the 1-dB gain compression point. Select a second marker and adjust its stimulus value until its response is 1-dB below the reference marker.
11. Change the S21 measurement from a log magnitude format to a phase format (no new calibration is required).
12. Find the phase change between the markers. The value is the AM-to-PM conversion coefficient at the 1-dB gain compression point.
13. Print the data or save it to a disk.
Gain

\[ \text{Gain (dB)} = 20 \log_10 \left( \frac{V_{\text{trans}}}{V_{\text{inc}}} \right) \]

The ratio of the amplifier's output power (delivered to a Z₀ \text{load}) to the input power (delivered from a Z₀ source). Z₀ is the \text{characteristic impedance}, in this case, 50 Ω.

For small signal levels, the output power of the amplifier is proportional to the input power. Small signal gain is the gain in this linear region.

As the input power level increases and the amplifier approaches saturation, the output power reaches a limit and the gain drops. Large signal gain is the gain in this nonlinear region. See \text{Gain Compression}.

Gain Flatness

The variation of the gain over the frequency range of the amplifier. See \text{Small Signal Gain and Flatness}.

Reverse Isolation

The measure of transmission from output to input. Similar to the gain measurement except the signal stimulus is applied to the output of the amplifier. See \text{Reverse Isolation}.

Gain Drift versus Time (temperature, bias)
The maximum variation of gain as a function of time, with all other parameters held constant. Gain drift is also observed with respect to other parameter changes such as temperature, humidity or bias voltage.

**Deviation from Linear Phase**

The amount of variation from a linear phase shift. Ideally, the phase shift through an amplifier is a linear function of frequency. See Deviation from Linear Phase.

**Group Delay**

\[
\tau_G (\omega) = - \frac{\Delta \phi}{\Delta \omega} \\
= - \frac{1}{360} \cdot \frac{\Delta \phi}{\Delta f}
\]

The measure of the transit time through the amplifier as a function of frequency. A perfectly linear phase shift would have a constant rate of change with respect to frequency, yielding a constant group delay. See Group Delay.

**Return Loss (SWR, \rho)**

\[
\Gamma = \frac{V_{ref}}{V_{inc}} = \rho \angle \delta \\
\text{Reflection coefficient} = \rho \\
\text{Return loss (dB)} = -20 \log_{10} \rho \\
\text{SWR} = \frac{1+\rho}{1-\rho}
\]

The measure of the reflection mismatch at the input or output of the amplifier relative to the system Z_0 characteristic impedance.

**Complex Impedance**

\[
Z = \frac{1+\Gamma}{1-\Gamma} \cdot Z_0 \\
= - R + jX
\]

Complex impedance (1+G). The amount of reflected energy from an amplifier is directly related to its impedance. Complex impedance consists of both a resistive and a reactive component. It is derived from the characteristic impedance of the system and the reflection coefficient. See Complex Impedance.

**Gain Compression**

See Gain Compression Application.

**AM-to-PM Conversion Coefficient**

\[
\frac{\Delta \phi}{\Delta P}
\]
The amount of phase change generated in the output signal of an amplifier as a result of an amplitude change of the input signal.

The AM-to-PM conversion coefficient is expressed in units of degrees/dB at a given power level (usually $P_{1\text{dB}}$, which is the 1 dB gain compression point). See AM-PM Conversion.
Antenna Measurements

This topic describes how to setup the PNA to make S21 measurements on an array of antennas. Measurements can be made on up to 100 antenna arrays (Ports) and up to 15 discrete frequencies.

Measurement Sequence

1. The PNA is set to a start frequency.
2. As the antenna moves, the PNA responds to each external trigger signal by measuring an antenna port.
3. When all ports are measured, the PNA increments to the next frequency.
4. Again the PNA measures all ports, and so forth until all ports are measured at all frequencies in the forward direction.
5. As the antenna begins moving in the opposite direction, the same sequence occurs, except the PNA decrements in frequency until all ports are measured at all frequencies and the PNA is set back to the original start frequency.

Once setup, only external trigger signals are sent to the PNA. After each trigger, measurement data is stored in internal PNA memory.

How to set up the PNA

See the Antenna Macro to learn how to do this automatically.

1. On the System menu click Preset
2. On the Sweep menu point to Trigger then click Trigger
3. In Trigger Source click External
4. In Trigger Scope click Channel
5. Click OK

Forward Sweep

1. On the Trace menu click New Trace
2. Click S21 then Channel Number 1
3. On the Sweep menu point to Trigger then click Trigger
4. In Channel Trigger State check Point Sweep
5. Click OK
6. On the Sweep menu click **Sweep Type:** then **Segment Sweep**

7. Click **OK**

8. On the **View** menu point to **Tables** then click **Segment Table**

9. Do this 15 times - Sweep menu point to **Segment Table** then **Insert Segment**

10. For each Segment in the Segment table:

    1. Click **State:** and select **ON**
    2. Double click both **START** and **STOP** Frequency: (each new segment ascends in frequency)
    3. Double click **Points:** type Number of Ports (elements)

**Reverse sweep**

Repeat the following steps for each frequency: (up to 15)

- Increment the channel number (X) Starting with Channel 2
- Decrement the frequency (F)

1. On the **Trace** menu click **New Trace...**

2. Click S21 then Channel Number X

3. When a window contains four traces, check **Create in New Window.**

4. Click **OK**

5. On the **Sweep** menu point to **Trigger** then click **Trigger**

6. In Channel Trigger State check **Point Sweep**

7. Click **OK**

8. On the Sweep menu click **Sweep Type:** then **Segment Sweep**

9. Click **OK**

10. On the **View** menu point to **Tables** then click **Segment Table**

11. In the Segment table

    1. Click **State:** and select **ON**
    2. Double click both **START** and **STOP** Frequency **F**
    3. Double click **Points:** type Number of Ports (elements)
Balanced Measurements

- **What are Balanced Devices?**
- **Differential and Common Modes Model**
- **Measuring Mixed Mode (Balanced) S-Parameters**
- **Measuring Imbalance Parameters**
- **Measuring CMRR**
- **Port Mapping**
- **How the PNA makes Balanced Measurements**

**Other Measurement Setup Topics**

**New** Check out the True Mode Stimulus Application (TMSA) available for download at http://na.tm.agilent.com/pna/apps/applications.htm

**What are Balanced Devices?**

Standard **Single-ended devices** generally have one input port and one output port. Signals on the input and output ports are referenced to ground.

Balanced devices have two pins on either the input, the output, or both. The signal of interest is the difference and average of the two input or output lines, not referenced to ground.

**Differential and Common Modes Model**

On balanced devices, the signal of interest is the difference and average of the two input or output lines. In balanced device terminology, these signals are known as the Differential and Common modes.

The following model shows how two signals (A and B) combine to create Differential and Common mode signals:
- **Signal A** is fixed at 1V peak
- **Signal B** is selectable
- **Differential** is calculated as \( A \) minus \( B \)
- **Common** is calculated as the **AVERAGE** of \( A \) and \( B \)

**Note:** Click **Signal B** selections to see various Differential and Common signals.

### Signal A = 1V

<table>
<thead>
<tr>
<th><strong>Differential (A - B)</strong></th>
<th><strong>Common (Avg) (A + B) / 2</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Signal A Waveform" /></td>
<td><img src="image2" alt="Common Waveform" /></td>
</tr>
</tbody>
</table>

### Signal B = SELECTABLE

<table>
<thead>
<tr>
<th><strong>Calculations</strong></th>
<th><strong>Signal B</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-ended 0V</td>
<td>1 - 0 = 1</td>
</tr>
<tr>
<td>1 - 0/2 = .5</td>
<td></td>
</tr>
<tr>
<td>180° Out of Phase 1V</td>
<td>1 - (-1) = 2</td>
</tr>
<tr>
<td>1 - 1/2 = 0</td>
<td></td>
</tr>
<tr>
<td>180° Out of Phase 2V</td>
<td>1 - (-2) = 3</td>
</tr>
<tr>
<td>1 - (-2)/2 = -.5</td>
<td></td>
</tr>
<tr>
<td>In Phase 1V</td>
<td>1 - 1 = 0</td>
</tr>
<tr>
<td>1 - 1/2 = 1</td>
<td></td>
</tr>
<tr>
<td>In Phase 2V</td>
<td>1 - 2 = -1</td>
</tr>
<tr>
<td>1 + 2/2 = 1.5</td>
<td></td>
</tr>
</tbody>
</table>

**Notes:**

- Even when Signal B is 0V, like a Single-ended signal, there is still a unique Differential and Common mode representation of the two individual signals.
- The above model does not show a DUT. The difference and average of two signals can be calculated for both the balanced INPUT and balanced OUTPUT of a device.

**Measuring Mixed Mode (Balanced) S-Parameters**

Mixed mode S-parameters combine traditional S-parameter notation with balanced measurement terminology. Some balanced devices are designed to amplify the differential component and reject the common component. This allows noise that is common to both inputs to be virtually eliminated from the output. For example, a balanced...
device may amplify the differential signal by a factor of 5, and attenuate the common signal by a factor of 5. Using traditional S-parameter notation, an S21 is a ratio measurement of the device Output / device Input. Mixing this with balanced terminology, we could view the amplifier's Differential Output signal / Differential Input signal. To see this parameter on the PNA, we would select an Sdd21 measurement using the following balanced notation:

\[ \text{Sabxy} - \]

Where

- **a** - device output mode
- **b** - device input mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

- **x** - device output "logical" port number
- **y** - device input "logical" port number

**See Also**

- Logical port mapping
- Port mapping with External Test Sets

**Measuring Imbalance Parameters**

Imbalance is a measure of how well two physical ports that make up a balanced port are matched. With a perfectly balanced port, the same amount of energy flows to both ports and the magnitude of the ratio of these ports is 1.

The notation is similar to traditional S-parameters. In the following diagrams, the letters a, b, c, and d are used because any PNA port can be assigned to any logical port using the port mapping process.

For example, in the following single-ended - balanced formula, \( S_{ba} \) indicates the device output port is logical port b and the input port is logical port a.

![Diagram showing Imbalance parameter when measuring a single-ended - balanced device.](image)
Imbalance1 and Imbalance2 parameters when measuring a balanced - balanced device.

\[
\text{Imbal1} = -\frac{S_{ac} - S_{ad}}{S_{bc} - S_{bd}} \\
\text{Imbal2} = -\frac{S_{ca} - S_{cb}}{S_{da} - S_{db}}
\]

Imbalance1 and Imbalance2 parameters when measuring a single-ended - single-ended - balanced device.

\[
\text{Imbal1} = -\frac{S_{ac} - S_{ad}}{S_{bc} - S_{bd}} \\
\text{Imbal2} = -\frac{S_{ca} - S_{cb}}{S_{da} - S_{db}}
\]

**Measuring CMRR (Common Mode Rejection Ratio)**

CMRR is a ratio of the transmission characteristic in differential mode over the transmission characteristic in the common mode of the balanced port as the measurement parameter. A high value indicates more rejection of common mode, which is desirable in a device that transmits information in the differential portion of the signal. The table below shows the CMRR parameter you can select when measuring each balanced device.
**Device Topology and Port Mapping**

As we have seen on balanced inputs and outputs, the signal of interest is the difference or average of two BALANCED input or BALANCED output lines. It is also possible to have single-ended ports AND balanced ports on the same device. The two balanced input or output lines are referred to as a single "logical" port.

When configuring a balanced measurement on the PNA, select a device 'topology'. Then map each PNA test port to the DUT ports. The PNA assigns "logical ports". See how to set device topology in the PNA.

The following device topologies can be measured by a 4-port PNA.

- **Balanced / Balanced**
  (2 logical ports - <4 actual ports>)
- **Single-ended / Balanced**
  (2 logical ports - <3 actual ports>)

  ![Single-ended / Balanced Diagram]

- **Single-ended - Single-ended / Balanced**
  (3 logical ports - <4 actual ports>)

  ![Single-ended - Single-ended / Balanced Diagram]

These topologies can be used in the reverse (⇐⇒) direction to measure:

- **Balanced / Single-ended** topology
- **Balanced / Single-ended - Single-ended** topology

For example, to measure a **Balanced / Single-ended** topology, measure the S12 (reverse direction) of a **Single-ended / Balanced** topology.

**How the PNA makes Balanced Measurements**

The PNA does not provide true balanced measurements by stimulating both balanced inputs together and measuring both outputs relative to one another. Instead, the PNA makes only Single-ended measurements. On a Balanced/ Balanced device, it stimulates each input and measures each output individually. From the output data, the PNA calculates the Differential and Common outputs from the DUT using the same math formulas as the above model. However, all measurements and calculations on the PNA are performed in frequency domain using complex (magnitude and phase) data. The Balanced S-parameter display data is then calculated from the Differential and Common inputs and outputs.
Complex Impedance

When making an S11 or S22 measurement of your device under test, you can view complex-impedance data such as series resistance and reactance as well as phase and magnitude information. Complex impedance data can be viewed using either the Smith Chart format or the Polar format.

- **What Is Complex Impedance?**
- **Accuracy Considerations**
- **How to Measure Complex Impedance**

**What Is Complex Impedance?**

Complex-impedance data is information that can be determined from an S11 or S22 measurement of your device under test, such as:

- Resistance
- Reactance
- Phase
- Magnitude

The amount of power reflected from a device is directly related to the impedances of both the device and the measuring system. For example, the value of the complex reflection coefficient \( G \) is equal to 0 only when the device impedance and the system impedance are exactly the same (i.e. maximum power is transferred from the source to the load). Every value for \( G \) corresponds uniquely to a complex device impedance (as a function of frequency), according to the equation:

\[
Z_L = [(1 + G) / (1 - G)] Z_0
\]

where \( Z_L \) is your test device impedance and \( Z_0 \) is the measuring system's characteristic impedance.

Complex Impedance is best viewed using either Polar or Smith Chart format.

**Accuracy Considerations**

- The Smith chart is most easily understood when used with a full scale value of 1.0.

- For greater accuracy when using markers in the Smith chart or polar formats, activate the discrete marker mode.

- The uncertainty of reflection measurements is affected by:
  - Directivity
  - Reflection tracking
  - Source match
  - Load match (with 2-port devices)
With a 2-port calibration, the effects of these factors are reduced. A 1-port calibration provides the same accuracy if the output of the device is well terminated. Refer to the graphic below for the following discussion.

- If you connect the device between both analyzer ports, it is recommended that you use a 10 dB pad on the output of the device to improve measurement accuracy. This is not necessary if you use a 2-port calibration since it corrects for load match.

- If you connect a two-port device to only one analyzer port, it is recommended that you use a high-quality load (such as a calibration standard) on the output of the device.

**How to Measure Complex Impedance**

1. Connect the device as shown in the previous graphic.
2. Preset the analyzer.
3. Set up, calibrate, and perform an S11 or S22 measurement.
4. View impedance data:
   a. Select the Smith Chart format.
   b. Scale the displayed measurement for optimum viewing.
   c. Position the marker to read the resistive and reactive components of the complex impedance at any point along the trace.
   d. Print the data or save it to a disk.
5. View the magnitude and phase of the reflection coefficient:
   a. Select the Smith chart format or the Polar format.
   b. Select either Lin Marker or Log Marker formats.
   c. Scale the displayed measurement for optimum viewing.
   d. Position the marker to read the frequency, magnitude, and phase of the reflection coefficient ( ) at any point along the trace.
   e. Print the data or save it to a disk.
Comparing the PNA "Delay" Functions

The PNA has three Delay functions which are similar but are used in different ways.

1. **Group Delay format** is used to display the Group Delay of a network. Group Delay is defined as:
   \[-d(\phi)/d(\omega)\] -- where \(\phi\) is radian angle, and \(\omega\) is radian frequency.

   Since it is defined by a derivative, the value must be determined from an analytic function. However, the PNA makes discrete measurements, so we approximate the group delay by taking the finite difference:
   \[-(1/360)\delta(\phi)/\delta(f)\] -- where \(\phi\) is degree angle and \(f\) is frequency in Hz. The \(1/360\) does the proper conversion of degrees to radians and Hz frequency to radian frequency.

   From this we can see that, if the phase response of a network varies with frequency, then the Group Delay must vary as well. In fact, many filters are specified by the variation of their Group Delay.

   If we measure the phase response of a lossless cable, it should be a straight line. But, of course, nothing is perfect. The phase response will have a small amount of noise. This is due to trace noise of the PNA, and the loss with real cables or transmission lines, which causes a small amount of non-linear phase change with frequency. So, if we look at the Group Delay of a cable, we will see a small amount of variation. Also, if the frequency spacing is small enough when you make the measurement, the \(\delta(f)\) in the denominator becomes very small, so the delay can have wide swings with just a little noise.

   To overcome this issue, we sometimes add smoothing to a phase trace, which widens the effective \(\delta(f)\), called the aperture, and provides a less noisy Group Delay response. The Group Delay of a device is only valid for a given frequency aperture. Learn more about Group Delay.

2. **Electrical Delay** function. On many filters, the passband response is specified for a maximum value of "Deviation from Linear Phase". When looking at the passband of a multi-pole filter, one sees the phase changing very rapidly. This makes it difficult to determine the linearity of the phase response. The Electrical Delay function subtracts out a "LINEAR PHASE" equivalent to the delay time value computed as above. When you use this function, you dial in the Linear Delay such that a CONSTANT PHASE SLOPE is removed from the phase trace, until the phase trace is mostly flat. The remaining variation is the deviation from linear phase.

   To make this task a little less tedious, the PNA has a marker function called Marker =>> Delay. This function computes the Group Delay value at the marker position, using a 20% smoothing aperture, then changes the Electrical Delay value to this value. Obviously, if the phase trace is not perfectly linear, moving the marker and recomputing the delay will result in different values. The phase slope added by the electrical delay function applies only to the current measurement. That is, each measurement (S11, S22, S12, S21) can have its own value of electrical delay. Learn more about Deviation from Linear Phase.

3. **Port Extension** is a function that is similar to calibration. It applies to all the traces in a given channel. It compensates for the phase response change that occurs when the calibration reference plane is not the same as the measurement plane of the device.

   Let's look at an example of a DUT that is mounted on a PCB fixture with SMA connectors. We can easily calibrate at the SMA connectors. But if we add the fixture to measure the board-mounted device, the apparent phase of the DUT is changed by the phase of the PCB fixture. We use port extensions to add a LINEAR PHASE (constant delay) to the calibration routines to shift the phase reference plane to that of the DUT. This is ONLY valid if the fixture consists of a transmission line with linear phase response, and this limitation is usually met in practice. The main reason that it is NOT met is that there is mismatch at the SMA-to-PCB interface. This mismatch was not removed with the error correction because it occurs AFTER the SMA connector. Ripple can be seen on the display as signals bounce back and forth between the mismatch and the DUT. If the DUT is well matched, the ripple effect is very small. However, when we use Automatic Port Extension (APE), and we leave the fixture open (the DUT removed), the reflection is large and we see larger ripples. That is why APE uses a curve fitting process to remove the effects of the ripple. For best effect, the wider the IF Bandwidth, the better we can "smooth-out" the ripples with...
curve fitting. Still, we are fitting a LINEAR PHASE SLOPE to the phase response, and thus we use only a single Port Extension Delay value to represent the phase slope.

The method used by older VNAs to get this same functionality was to add a mechanical line stretcher to the reference channel, which removed a fixed delay amount from the port. Port extensions give 1x the delay for transmission at each port, and 2x the delay for reflection, so it differs somewhat from Electrical Delay above, in that the math function depends upon the measurement being made. The signal passes twice through the fixture for reflection (out and back), but only once for each port on transmission. For S21, the phase slope added is the sum of the port 1 and port 2 Port Extension Delay values.

The "User Range" APE function is used in cases where a fixture has limited bandwidth, perhaps due to tuning elements or bias elements. In this case, the model of constant delay for the fixture over the whole bandwidth is not valid, so a narrower "User Range" of frequencies can be selected to compute the delay. Since the aperture is smaller, there is more uncertainty in the delay computation for port extension. Also, for those who had been using the Marker => Delay function to estimate the delay, we added the "Active Marker" selection to APE, which works exactly the same as Marker->Delay. Learn more about Automatic Port Extensions.
Deviation from Linear Phase

Deviation from linear phase is a measure of phase distortion. The electrical delay feature of the analyzer is used to remove the linear portion of the phase shift from the measurement. This results in a high-resolution display of the non-linear portion of the phase shift (deviation from linear phase).

- **What Is Linear Phase Shift?**
- **What Is Deviation from Linear Phase?**
- **Why Measure Deviation from Linear Phase?**
- **Using Electrical Delay**
- **Accuracy Considerations**

See also [Comparing the PNA Delay Functions](#)

### What Is Linear Phase Shift?

Phase shift occurs because the wavelengths that occupy the electrical length of the device get shorter as the frequency of the incident signal increases. Linear phase-shift occurs when the phase response of a device is linearly proportional to frequency. Displayed on the analyzer, the phase-versus-frequency measurement trace of this ideal linear phase shift is a straight line. The slope is proportional to the electrical length of the device. Linear phase shift is necessary (along with a flat magnitude response) for distortionless transmission of signals.

### What Is Deviation from Linear Phase?

In actual practice, many electrical or electronic devices will delay some frequencies more than others, creating non-linear phase-shift (distortion in signals consisting of multiple-frequency components). Measuring deviation from linear phase is a way to quantify this non-linear phase shift.

Since it is only the deviation from linear phase which causes phase distortion, it is desirable to remove the linear portion of the phase response from the measurement. This can be accomplished by using the electrical delay feature of the analyzer to mathematically cancel the electrical length of the device under test. What remains is the deviation from linear phase, or phase distortion.

### Why Measure Deviation from Linear Phase?

The deviation from linear phase measurement accomplishes the following:

- Presents data in units of phase rather than units of seconds (group delay). For devices that pass modulated signals, units of phase may be most practical.
- Provides a less noisy measurement than a group delay measurement.
Using Electrical Delay

The electrical delay feature is the electronic version of the mechanical "line stretcher" of earlier analyzers. This feature does the following:

- Simulates a variable-length lossless transmission line, which is effectively added to or removed from the reference signal path.
- Compensates for the electrical length of the device under test.
- Flattens the measurement trace on the analyzer's display. This allows the trace to be viewed at high resolution in order to see the details of the phase nonlinearity.
- Provides a convenient method to view the deviation from linear phase of the device under test. See the following graphic.

Learn how to set Electrical Delay.

Accuracy Considerations

The frequency response of the test setup is the dominant error in a deviation from linear phase measurement. To reduce this error, perform a 2-port measurement calibration.

How to Measure Deviation from Linear Phase:

1. Preset the analyzer.
2. If your device under test is an amplifier, it may be necessary to adjust the analyzer's source power:
   - Set the analyzer's source power to be in the linear region of the amplifier's output response (typically 10-dB below the 1-dB compression point).
   - Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port 2.
3. Connect the device under test as shown in the following graphic.
3. Select an S21 measurement.

4. Select the settings for your device under test, including the following:
   - **Format**: phase
   - **Scale**: autoscale

5. Remove the device and perform a calibration.

6. Reconnect the device.

7. Scale the displayed measurement for optimum viewing.

8. Create a marker in the middle of the trace.

9. Press the >Delay **Active Entry Key** to invoke the Marker to Electrical Delay function. This flattens the phase trace.

10. If desired, on the Scale menu, click Electrical Delay to fine-tune the flatness of the phase trace.

11. Use the markers to measure the maximum peak-to-peak deviation from linear phase.

12. Print the data or save it to a disk.
Synchronize PNA with an External (PSG) Source

Beginning with PNA Rev. 7.22, the PNA External Source Control feature can be used to automatically control external sources. However, this feature requires certain PNA options. Learn more.

Many PNA measurements require the use of at least one external source. For example, when measuring the insertion loss of a mixer, the LO must be swept at the same time as the RF input. This requires the PNA and external source to be synchronized.

The following procedure shows how to manually synchronize the PNA with an Agilent PSG Source. Although the settings will be different, the concept is useful with other sources.

**Hardware configuration**

- Connect the PNA and PSG Time Base ([PNA 10 MHz OUT](#) to PSG 10 MHz IN)

**PNA-L, E836xB Models**

Connect the PSG and [PNA Trigger Connectors](#) as follows:

- PNA Trigger IN to Source OUT
- PNA Trigger OUT to Source IN

**PNA-X Models**

Connect either pair (1 or 2) of the [AUX Trigger I/O connectors](#) as follows:

- PNA AUX Trig IN to Source Trigger OUT
- PNA AUX Trig OUT to Source Trigger IN

Learn more about the AUX Trigger capabilities.

**PNA Settings**

- **Number of points**: Same as PSG
- **Frequency span**: Does NOT have to be the same as PSG

**PNA Trigger Settings**

- Trigger Source:
  - PNA-L and E836xB models: External
  - PNA-X: Internal, Manual
- Trigger Scope: **Channel**
- Channel Trigger State: **Same as PSG Sweep Repeat setting** *(Continuous or Single)*
- Point Sweep: **Checked**

**External / Auxiliary Trigger Settings**

<table>
<thead>
<tr>
<th></th>
<th>PNA-L and E836xB</th>
<th>PNA-X</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Where settings are made:</td>
<td><strong>External Tab</strong></td>
<td><strong>Aux Trigger Tab</strong></td>
</tr>
<tr>
<td>Level / Edge:</td>
<td>Edge</td>
<td>Same as PSG (Hi or Low)</td>
</tr>
<tr>
<td>Accept Trigger Before Armed:</td>
<td><strong>Checked</strong></td>
<td>N/A</td>
</tr>
<tr>
<td>Handshake</td>
<td>N/A</td>
<td>Checked</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Where settings are made:</td>
<td><strong>I/O2 Trig Out Tab</strong></td>
<td><strong>Aux Trigger Tab</strong></td>
</tr>
<tr>
<td>Enable</td>
<td><strong>Checked</strong></td>
<td><strong>Checked</strong></td>
</tr>
<tr>
<td>Polarity:</td>
<td>Same as PSG</td>
<td>Same as PSG</td>
</tr>
<tr>
<td>Position:</td>
<td>After</td>
<td>After</td>
</tr>
<tr>
<td>Per Point</td>
<td><strong>Checked</strong></td>
<td><strong>Checked</strong></td>
</tr>
</tbody>
</table>

**PSG Settings**

- Number of points: **Same as PNA**
- Sweep: **Step** or **List**
- Sweep Trig: **Free Run**
- Sweep Repeat: **Same as PNA Channel Trigger State** *(Continuous or Single)*
- Sweep Direction: **UP**
- Point Trig: **Ext**
- Manual Mode: **OFF**
- Trigger In / Out Polarity: **Same as PNA**
**What is Happening?**

The following is a flow diagram showing the handshake / synchronization process between the PNA and an External Source.

<table>
<thead>
<tr>
<th>PNA</th>
<th>External Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. <strong>Go to 1st data point</strong></td>
<td>1. <strong>Go to 1st data point</strong></td>
</tr>
<tr>
<td>2. <strong>Wait</strong> (PNA-X only)</td>
<td><strong>Ready for Trigger (IN)</strong></td>
</tr>
<tr>
<td>3. <strong>Start</strong></td>
<td>3. <strong>Start</strong></td>
</tr>
<tr>
<td>4. <strong>Data Acq</strong></td>
<td>4. <strong>Data Acq</strong></td>
</tr>
<tr>
<td>5. <strong>End</strong></td>
<td>5. <strong>End</strong></td>
</tr>
<tr>
<td>6. <strong>Go to next data point</strong></td>
<td>6. <strong>Go to next data point</strong></td>
</tr>
</tbody>
</table>

### Text Description

1. After the measurement setup is complete, both instruments wait on the first data point of a measurement sweep. Both instruments are configured for Continuous or Single sweep.

2. *(see note below)* A trigger signal from the source starts the measurement. This is usually accomplished by a key press on the source front panel.

3. PNA data acquisition (measurement) starts, and then stops AFTER the first data point acquisition.

4. The PNA sends a trigger signal out to the source telling it to move to the next frequency data point. This signal can optionally be sent BEFORE data acquisition if required by your application.

5. The external source and PNA move to the next data point. The source usually takes longer than the PNA.

6. The source sends the Ready for Trigger signal to the PNA.

- **PNA-L and E836xB models** - If the source arrives first, the **Accept Trigger Before Armed** setting is used to accept the trigger signal even if the PNA is not yet ready to start acquisition.

- **PNA-X using AUX Triggering** - If enabled, the PNA waits indefinitely for a trigger signal from the source. Although AUX triggering does NOT have the **Accept Trigger Before Armed** setting, the Ready for Trigger signal is latched and has the same effect.

### Step 2 Note

**PNA-X (Aux TRIG IN)** The PNA looks for a level trigger at the start of each sweep, and an edge thereafter. This assumes that the external source ready line will remain in the ready state (high or low) until triggered (step 4) and will then transition to the NOT ready state while moving to the next frequency, and then transition again to the ready state.
How do you know when the PNA and PSG are in synch?

The measurement results are the ultimate test of whether the source and PNA are synchronized. However, it is possible to see the PSG and PNA sweeping at exactly the same time.

First, lower the PNA IFBandwidth or increase the sweep time so that the sweep is slow enough to watch the sweep indicator moving across the PNA screen. At the same time, watch the PSG “progress bar” as it moves through the entire sweep.

If the PNA is stopped in the middle of a sweep, then re-triggered, it returns to the first data point. The PSG continues from where it stopped. Therefore, to re-synch the two instruments, the PSG needs to return to the first data point. There are a number of ways to do this. One way is to press the PSG Manual button to ON, then OFF. Then trigger a new sweep.

To trigger a sweep

- **Single** Trigger mode: Both the PNA and PSG Single (trigger) buttons must be pressed (in any order) for each trigger.

- **Continuous** Trigger mode: First, reset the PSG to the first data point, then press the PNA Continuous (trigger) button.

Maintaining Synchronization

In general, the above setup should start the two instruments sweeping simultaneously. However, any interaction with the PNA could cause the PNA sweep to abort or delay, in which case the two instruments will be out of sync. To avoid this, you can use the PNA Interface Control feature to send an ABORT to the external device after each sweep.

When the PNA ends a sweep, it sends an ABORT to stop the source. A trigger signal is then sent, either Continuous (automatically) or Single (manual). In either case, both instruments start sweeping in synch.

This takes more time to sweep, but maintains synchronization.

For example, to use this feature with Agilent’s PSG source, you would add the following:

On the “After Sweep End” tab, type:

```
24 :ABORT
```

Where 24 is the GPIB address of the source.

---

Last Modified:

- 11-Feb-2008      Updated note
- 1-Jun-2007       Added 7.22 update
- 1-Jan-2007       MX Updated for PNA-X
Small Signal Gain and Flatness

Small signal gain is the gain in the amplifier's linear region of operation. This is typically measured at a constant input power over a swept frequency. Gain flatness is the measure of the variation of gain over a specified frequency range.

- **What Is Gain?**
- **What Is Flatness?**
- **Why Measure Gain and Flatness?**
- **Accuracy Considerations**
- **How to Measure Gain and Flatness**

**See other Amplifier Parameter topics**

**What Is Gain?**

RF amplifier gain is defined as the difference in power between the amplifier output signal and the input signal. It is assumed that both input and output impedances of the amplifier are the same as the characteristic impedance of the system.

- Gain is called $S_{21}$ using S-parameter terminology
- Gain is expressed in dB—a logarithmic ratio of the output power relative to the input power.
- Gain can be calculated by subtracting the input from the output levels when both are expressed in dBm, which is power relative to 1 milliwatt.
- Amplifier gain is most commonly specified as a minimum value over a specified frequency range. Some amplifiers specify both minimum and maximum gain, to ensure that subsequent stages in a system are not under or over driven.

**What Is Flatness?**

Flatness specifies how much the amplifier's gain can vary over the specified frequency range. Variations in the flatness of the amplifier's gain can cause distortion of signals passing through the amplifier.

**Why Measure Small-Signal Gain and Flatness?**

Deviations in gain over the bandwidth of interest will induce distortion in the transmitted signal because frequency components are not amplified equally. Small-signal gain allows you to quantify the amplifier's gain at a particular frequency in a 50-ohm system. Flatness allows you to view the deviations in the amplifier's gain over a specified frequency range in a 50-ohm system.
Accuracy Considerations

- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

- The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:
  - damage the analyzer receiver
  - exceed the input compression level of the analyzer receiver, resulting in inaccurate measurements.

  **Attenuation** of the amplifier's output power can be accomplished using:
  - attenuators
  - couplers

  The frequency-response effects and mismatches of the attenuators and couplers must be accounted for during **calibration** since they are part of the test system. Proper error-correction techniques can reduce these effects.

- The **frequency response** is the dominant error in a small-signal gain and flatness measurement setup. Performing a thru-response measurement calibration significantly reduces this error. For greater accuracy, perform a 2-port measurement calibration.
- Reducing IF bandwidth or using **averaging** improves measurement **dynamic range** and accuracy, at the expense of measurement speed.

How to Measure Gain and Flatness

1. Preset the analyzer.
2. Select an S21 measurement parameter.
3. Set the analyzer's source power to be in the linear region of the amplifier's output response (typically 10-dB below the 1-dB compression point).
4. Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port-2.
5. Connect the amplifier as shown in the following graphic, and provide the dc bias.

6. Select the analyzer settings for your amplifier under test.

7. Remove the amplifier and perform a measurement calibration. Be sure to include the attenuator and cables in the calibration setup if they will be used when measuring the amplifier.

8. Save the instrument-state to memory.

9. Reconnect the amplifier.

10. Scale the displayed measurement for optimum viewing and use a marker to measure the small signal gain at a desired frequency.

11. Measure the gain flatness over a frequency range by using markers to view the peak-to-peak ripple.

12. Print or save the data to a disk.

13. This type of measurement can be automated.
Gain Compression

Gain compression measures the level of input power applied to an amplifier that will cause a distorted output. The Gain Compression Application (Opt 086) makes fast and accurate compression measurements.

- **What Is Gain Compression?**
- **Why Measure Gain Compression?**
- **Accuracy Considerations**
- **How to Measure Gain Compression**

---

**What Is Gain Compression?**

Gain compression occurs when the input power of an amplifier is increased to a level that reduces the gain of the amplifier and causes a nonlinear increase in output power.

The analyzer has the ability to do power sweeps as well as frequency sweeps. Power sweeps help characterize the nonlinear performance of an amplifier. Refer to the graphic below (a plot of an amplifier's output power versus input power at a single frequency) for the following discussion.

- The amplifier has a linear region of operation where gain is constant and independent of power level. The gain in this region is commonly referred to as "small-signal gain."
- As the input power increases, the amplifier gain appears to decrease, and the amplifier goes into compression.
- The most common measurement of amplifier compression is the 1-dB compression point. This is defined as the input power (or sometimes the output power) which results in a 1-dB decrease in amplifier gain (relative to the amplifier's small-signal gain).

---

**Why Measure Gain Compression?**

When driven with a sinusoid, the output of an amplifier is no longer sinusoidal in the compression region. Some of the amplifier output appears in harmonics, rather than occurring only at the fundamental frequency of the input.
As input power is increased even more, the amplifier becomes saturated, and output power remains constant. At this point, further increases in amplifier input power result in no change in output power.

In some cases (such as with TWT amplifiers), output power actually decreases with further increases in input power after saturation, which means the amplifier has negative gain.

Since gain is desired in amplifier operation, it is important to know the limit of input signal that will result in gain compression.

**Accuracy Considerations**

The network analyzer must provide sufficient power to drive the amplifier into saturation. If you need a higher input-power level than the source of the analyzer can provide, use a preamplifier to boost the power level prior to the amplifier under test. (See High-Power Component Measurements.) If using a preamplifier, you can increase measurement accuracy in the following ways:

- Use a coupler on the output of the preamplifier so that a portion of the boosted input signal can be used for the analyzer's reference channel. This configuration removes the preamplifier's frequency response and drift errors from the measurement (by ratioing).
- Perform a thru-response calibration including the preamplifier, couplers, and attenuators in the test setup.

The output power of the amplifier should be sufficiently attenuated if necessary. Too much output power could:

- Damage the analyzer receiver
- Exceed the input compression level of the analyzer receiver

Attenuation of the amplifier's output power can be accomplished using:

- Attenuators
- Couplers

The frequency-response effects of the attenuators and couplers must be considered during calibration since they are part of the test system. Proper error-correction techniques can reduce these effects.

- The frequency response is the dominant error in a gain compression measurement setup. Performing a thru-response measurement calibration significantly reduces this error.
- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.
- Reducing IF bandwidth or using measurement averages improves accuracy, at the expense of measurement speed.

**How to Measure Gain Compression**

This procedure shows you how to make the following three measurements used to determine amplifier gain compression:
1. A **Swept-Frequency Gain Compression** measurement locates the lowest frequency at which the 1-dB gain compression first occurs.

2. A **Swept-Power Gain Compression** measurement shows the input power at which a 1-dB drop in gain occurs as a power ramp is applied to the amplifier at a particular frequency point (found in measurement 1).

3. An **Absolute Power** measurement shows the absolute power out (in dBm) at compression.

**Swept-Frequency Gain Compression Measurement**

A measurement of swept frequency gain compression locates the frequency point where 1-dB compression first occurs.

1. Preset the analyzer.

2. Select an $S_{21}$ measurement parameter.

3. Set the analyzer’s source power to be in the linear region of the amplifier’s output response (typically 10-dB below the 1-dB compression point).

4. Select an external attenuator (if needed) so the amplifier’s output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer’s port.

5. Connect the amplifier as shown in the following graphic, and provide the dc bias.

6. Select the analyzer settings for your amplifier under test. To reduce the effects of noise, you may want to specify a narrower IF bandwidth.

   ![Diagram of amplifier setup](image)

   * Direct Connection

7. Remove the amplifier and perform a thru-response calibration. Be sure to include the attenuator and cables in the calibration setup if they will be used when measuring the amplifier.

8. Save the instrument-state to memory.

9. Reconnect the amplifier.

10. Position a marker at approximately mid-span.

11. Adjust the analyzer’s scale to 1 dB per division.
12. Store the trace in memory and display Data/Mem.

13. Gradually increase the source power until a 1-dB decrease in gain is observed at the first frequency over some portion of the trace.

14. Use markers to locate the frequency where the 1-dB decrease in gain first occurs. Note this frequency for use in the following measurement.

15. Print the data or save it to a disk.

**Swept-Power Gain Compression Measurement**

A swept-power gain compression measurement shows the input power resulting in a 1-dB drop in gain as a power ramp at a particular frequency (found in step 13 of the previous measurement) is applied to the amplifier.

1. If not already done, perform the previous measurement of swept-frequency gain compression.

2. Setup an S21 measurement in the power-sweep mode. Include the following settings:
   - Set the CW frequency to the frequency noted in step 14 of the previous measurement of swept-frequency gain compression.
   - Enter the start and stop power levels for the sweep. The start power should be in the linear region of the amplifier's response (typically 10 dB below the 1-dB compression point). The stop power should be in the compression region of the amplifier's response.

3. Adjust the scale to 1-dB per division.

4. Use markers (including reference marker) to find the input power where the 1-dB decrease in gain occurs.

5. Print the data or save it to a disk.

**Absolute Output Power Measurement**

An absolute-power measurement shows the absolute power-out (in dBm) of the amplifier at compression.

1. Select an unratioed (absolute) power measurement. Choose the B input if using the test setup in the previous graphic.

2. Retain the CW frequency used in the previous measurement of swept-power gain compression.

3. Set a marker to the input power level where the 1-dB decrease in gain occurs (found in step 4 of the previous measurement).

4. Scale the displayed measurement for optimum viewing.

5. Read the marker value to find the absolute output power of the amplifier (in dBm) where the 1-dB decrease in gain occurs.

6. Print the data or save it to a disk.
**Note:** The measurement calibration does not apply to absolute power. Therefore, if there is any attenuation external to the analyzer, you will have to correct for it manually.
Group Delay

Group delay is a measure of phase distortion. Group delay is the actual transit time of a signal through a device under test as a function of frequency. When specifying group delay, it is important to specify the aperture used for the measurement.

- **What is Group Delay?**
- **Group Delay versus Deviation from Linear Phase**
- **What Is Aperture?**
- **Accuracy Considerations**
- **How to Measure Group Delay**

See also [Comparing the PNA Delay Functions](#).

### See other Amplifier Parameter topics

---

### What Is Group Delay?

Group delay is:

- A measure of device phase distortion.
- The transit time of a signal through a device, versus frequency.
- The derivative of the device's phase characteristic with respect to frequency.

Refer to the graphic below for the following discussion:

![Graph of phase characteristic](chart.png)

The phase characteristic of a device typically consists of both linear and higher order (deviations from linear) phase-shift components.
Linear phase-shift component:  Represents average signal transit time.
Attributed to electrical length of test device.

Higher-order phase-shift component:  Represents variations in transit time for different frequencies.
Source of signal distortion.

Refer to the graphic below for the following discussion:

In a group delay measurement:

- The linear phase shift component is converted to a constant value (representing the average delay).
- The higher order phase shift component is transformed into deviations from constant group delay (or group delay ripple).
- The deviations in group delay cause signal distortion, just as deviations from linear phase cause distortion.
- The measurement trace depicts the amount of time it takes for each frequency to travel through the device under test.

Refer to the following equation for this discussion on how the analyzer computes group delay:

\[
\text{Group Delay} = tg = \frac{-d\phi}{d\omega} = \frac{-1}{360^\circ} \cdot \frac{d\Theta}{df}
\]

- Phase data is used to find the phase change (-d \(\phi\)).
- A specified frequency aperture is used to find the frequency change (d \(\omega\)).
- Using the two values above, an approximation is calculated for the rate of change of phase with frequency.
- This approximation represents group delay in seconds (assuming linear phase change over the specified frequency aperture).

**Group Delay versus Deviation from Linear Phase**

Group delay is often a more accurate indication of phase distortion than Deviation from Linear Phase.

**Deviation from linear phase** results are shown in the upper region of the following graphic: Device 1 and device 2 have same value, despite different appearances.
**Group Delay** results are shown in the lower region:
Device 1 and device 2 have different values of group delay. This is because in determining group delay, the analyzer calculates slope of phase ripple, which is dependent on number of ripples which occur per unit of frequency.

**What Is Aperture?**
During a group delay measurement, the analyzer measures the phase at two closely spaced frequencies and then computes the phase slope. The frequency interval (frequency delta) between the two phase measurement points is called the *aperture*. Changing the aperture can result in different values of group delay. The computed slope (delta phase) varies as the aperture is increased. This is why when you are comparing group delay data, you must know the aperture that was used to make the measurements.

Refer to the graphic below for the following discussion:

Narrow aperture: Provides more fine detail in phase linearity.

Wide aperture: Provides less fine detail in phase linearity because some phase response averaged-out or not measured.

Makes measurement susceptible to noise (smaller signal-to-noise ratio) and analyzer phase detector resolution.

Makes measurement less susceptible to noise (larger signal-to-noise ratio).

The analyzer's default setting for group delay aperture is the *frequency span* divided by the number of points across the display. There are two ways to set the aperture to a different value.
1. Adjust the number of measurement points or the frequency span.

- Increasing the number of points or reducing the frequency span narrows the aperture.
- Decreasing the number of points and/or increasing the frequency span widens the aperture.

**Note:** if the aperture is too wide (more than 180° of phase shift between adjacent frequency points), errors in group delay data will occur.

2. Use the analyzer's smoothing function.

- Performs a single-sweep, moving average of adjacent data-points over a specified percentage of the frequency span.
- Results in an action similar to changing the frequency interval between points.
- Allows a wider aperture because greater than 180° of phase shift can occur over the smoothing aperture.

Group delay measurements can be made on the following sweep types:

- Linear frequency
- List frequency sweep segment

The group delay aperture varies depending on the frequency spacing and point density, therefore the aperture is not constant in segment sweep. In segment sweep, extra frequency points can be defined to ensure the desired aperture.

**Accuracy Considerations**

It is important to keep the phase difference between two adjacent measurement points less than 180° (see the following graphic). Otherwise, incorrect phase and delay information may result. Undersampling may occur when measuring devices with long electrical length. You can verify that the phase difference measured between two adjacent points is less than 180° by adjusting the following settings until the measurement trace no longer changes:

- Increase the number of points
- Narrow the frequency span

**Electrical delay** may also be used to compensate for this effect.
The **frequency response** is the dominant error in a group delay test setup. Performing a thru-response measurement **calibration** significantly reduces this error. For greater accuracy, perform a 2-port measurement calibration.

Particularly for an amplifier, the response may vary differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

**How to Measure Group Delay**

1. Preset the analyzer.

2. If your device under test is an amplifier, it may be necessary to adjust the analyzer's source power:
   
   - Set the analyzer's source power to be in the linear region of the amplifier's output response (typically 10-dB below the 1-dB compression point).
   
   - Select an external attenuator (if needed) so the amplifier's output power will be sufficiently attenuated to avoid causing receiver compression or damage to the analyzer's port 2.

3. Connect the device under test as shown in the following graphic.

4. Select an $S_{21}$ measurement.
5. Select the settings for your device under test, including the following:

   o number of measurement points: maximum
   o format: delay
   o scale: autoscale

6. Remove the device under test and perform a measurement calibration.

7. Reconnect the device under test.

8. Scale the displayed measurement for optimum viewing.

9. Use the analyzer's smoothing feature to increase the aperture, reducing noise on the trace while maintaining meaningful detail. To increase the aperture:

   o Switch on the analyzer's smoothing feature.
   o Vary the smoothing aperture (up to 25% of the span swept).

10. Use the markers to measure group delay (expressed in seconds) at a particular frequency of interest.

11. Print the data or save it to a disk.
High-Power Amplifier Measurements Using a PNA

This topic is now covered in detail in Application Note 1408-10, High-power measurements using the PNA (5989-1349EN) at Agilent.com.

See Also

High-Power Amplifier Measurements using a PNA-X
High Power Amplifier Measurements with the PNA-X

The following is a block diagram of the PNA-X Opt 423. The configuration displayed here is used to make high power amplifier measurements using a preamplifier at the rear panel. The preamplifier can then be switched (SW1) as needed using the RF Configurator.

Legend

<table>
<thead>
<tr>
<th>Color</th>
<th>Component</th>
<th>Damage Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Green</td>
<td>Bridges</td>
<td>+33 dBm</td>
</tr>
<tr>
<td>Blue</td>
<td>Couplers</td>
<td>+43 dBm</td>
</tr>
<tr>
<td>Orange</td>
<td>Bias-tees</td>
<td>+30 dBm</td>
</tr>
<tr>
<td>Purple</td>
<td>User-supplied pre-amp and high-power attenuator</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Notes

At J11 (rear-panel), max power is 4 dB to 11 dB higher than Source 1 Out at front panel jumper due to loss of the coupler thru arms, bias-tees, and cables.

At J10 max power +33 dBm, which is the damage level of the bridge. With +30 dBm into J10, there will be about +15 dBm at R1, assuming 15 dB coupling factor for the R1 bridge. +15 dBm is the damage level of that receiver. Therefore, it may be necessary to add attenuation in place of the R1 loop, not only to protect the receiver, but to bring it out of compression. The 0.1 dB compression level spec for the R1 receiver is between -3 and -18 dBm, depending on the frequency and option configuration.
At Test Port 2 (DUT output): With the bias-tees (orange), only +30 dBm is allowed into the test port. With Opt H85 (bias-tees removed), +43 dBm is allowed. Add appropriate attenuation to not damage other components.

See Also

- Front panel jumper specs.
- RF Path Configurator
- IF Path Configurator

Last Modified:

10-May-2007    MX New topic
Impedance Matching Model

Impedance matching is a procedure used in circuit design to match unequal source and load impedances, thereby optimizing the power delivered to the load from the source. Impedance matching is accomplished by inserting matching networks into a circuit between the source and the load.

- Introduction to the Model
- Impedance Matching Model
- Description of Exercises
- Smith Chart Circuit Elements Paths
- Forbidden Regions of the Smith Chart

Other Tutorials topics

Introduction

In this model, Smith Charts are used to visualize the interactive process of impedance matching to optimize transmitted power in simple circuits. Simple series/shunt, inductance/capacitance matching networks are used, and you can interactively adjust the values of corresponding L and C components. Adjusting the matching network components changes the reflectance of the overall circuit. The reflectance of each part of the circuit is indicated on the Smith Chart as a red or blue ball.

As you adjust the sliders and modify the component values, the model calculates new values for the circuit reflectance and moves the red and blue balls on the Smith Chart. The goal of each exercise is to move the reflectance point from the center of the Smith Chart, which represents either the load or source, into the appropriate red and blue rings which represent the desired matching condition. You can select three different impedance matching problems of increasing difficulty by clicking on one of the three labeled tabs.

Impedance Matching Model

Maximize this window for optimum viewing. Click if the Impedance model is not visible.

Description of Exercises

The first exercise lets you use the Smith Chart to perform basic impedance matching between a resistive source and a resistive load. A simple series-inductance shunt-capacitance network is used to match the 50 ohm source to the 300 ohm load. The source reflectance of the circuit looking from the load toward the source is represented by the red ball, while the 300 ohm load is indicated by the stationary red ring.
The objective of the exercise is to interactively match these two impedances by adjusting the L and C sliders. The model will provide graphical feedback by moving the red ball indicating circuit reflectance on the Smith Chart. Adjust the series L and shunt C sliders to move the reflectance point from the center of the Smith Chart to the matching impedance position inside the red ring. You can study the Smith Chart Circuit Element Paths below for hints on how different circuit elements change circuit reflectance on the Smith Chart.

The second exercise provides the impedance matching experience of optimizing the transducer power gain of a transistor amplifier. Matching the 50 ohm source to the input reflectance of the transistor, $s^{*11}$, and matching the 50 ohm load to the output reflectance of the transistor, $s^{*22}$, optimizes the power delivered from the source, through the transistor, to the load. You are required to match both the input red ball and output blue ball of the transistor separately. Adjust the component values to move both reflectance points to their proper positions within the red and blue rings.

The third part of the interactive impedance matching model is a collection of exercises involving a modular circuit. You begin by constructing a circuit with either one or two modular drag-and-drop matching network components. Once the matching networks have been added to the circuit, the sliders will become active and allow you to adjust the component values. Then you will engage in impedance matching for the circuit you have just created! There are 8 different circuits you can construct and there are 5 different value pairs for $s^{*11}$ and $s^{*22}$ on the Smith Chart, altogether 40 impedance matching exercises. You will find that not all matching networks will work! For some of the circuits you will be able to construct, you will not be able to position the red ball within the red ring or the blue ball within the blue ring. To determine in advance which matching networks will work, take a close look at the Forbidden Regions of the Smith Chart below. There are 5 different location pairs for $s^{*11}$ and $s^{*22}$ corresponding to different frequencies that can be matched. Use the frequency indicator to select an operating frequency, and then drag-and-drop appropriate matching networks into the circuit and adjust the component values to move both reflectance points to their proper positions within the red and blue rings.

**Smith Chart Circuit Elements Paths**

The graphs below demonstrate how the various shunt and series L and C components change the circuit reflectance on the Smith Chart. Assuming the given component is the last component in the matching network, the circuit reflectance will move as indicated along constant resistance or constant conductance circles.

You can think of impedance matching using the Smith Chart as driving a car to a specific destination in Smith Town - a city were none of the streets are straight! By adjusting circuit components in appropriate order, we can constrain the circuit reflectance to paths along constant resistance or constant conductance circles. Just like road signs can direct a car along the circular streets of Smith Town, so can we reach the matching impedance condition in a straightforward and deterministic way.
Forbidden Regions of the Smith Chart

For a given load reflectance, only certain L-C matching networks will be capable of transforming the source impedance to the load impedance. In fact, for any load reflectance, exactly two of the four possible L-C matching networks in the Transistor Amplifier-II model above will be able to do the matching job. But which two?

The charts below can be used to determine which matching networks will work in a given load situation. If the load reflectance lies within the forbidden region of the Smith Chart for the indicated matching network, then that network cannot perform the required matching operation. You cannot drive your car into the forbidden neighborhoods of Smith Town! They are unpaved!

Use these charts to determine which matching network should be used. First, visually locate the position of the load reflectance from the Transistor Amplifier-II model above on each of the four color Smith Charts below. Then, eliminate the two networks whose forbidden regions overlap the reflectance point, and use one of the remaining two networks to perform the impedance match.
Phase Measurements

Knowledge of both magnitude and phase characteristics is needed for successful higher-level component integration.

- What are Phase Measurements?
- Why Measure Phase?
- Using the Analyzer's Phase Format
- Types of Phase Measurements

See other Tutorials

What are Phase Measurements?
Phase measurements are made using S-parameters, just like amplitude measurements. A phase measurement is a relative (ratio) measurement and not an absolute measurement. Phase measurements compare the phase of the signal going into a device (the incident signal) to the phase of the device's response signal. The response signal can be either reflected or transmitted. Assuming an accurate calibration has been performed, the difference in phase between the two signals (known as phase shift) is a result of the electrical characteristics of the device under test.

The following graphic shows the phase shift (in time or degrees) between an incident signal and a transmitted signal (as might be seen on an oscilloscope display).

Why Measure Phase?
Measuring phase is a critical element of network analysis. The following graphic lists five reasons for measuring both magnitude and phase.
When used in communications systems to pass signals, components or circuits must not cause excessive signal distortion. This distortion can be:

- **Linear**, where flat magnitude and linear phase shift versus frequency is not maintained over the **bandwidth** of interest.
- **Nonlinear**, such as AM-to-PM conversion.

It is important to measure how reflective a component or circuit is, to ensure that it transmits or absorbs energy efficiently. Measuring the complex **impedance** of an antenna is a good example.

### Using the Analyzer’s Phase Format

The analyzer’s phase format displays a phase-versus-frequency or phase-versus-power measurement. The analyzer does not display more than ±180 degrees phase difference between the reference and test signals. As the phase value varies between +180 degrees and -180 degrees, the analyzer display creates the sawtooth pattern as shown in the following graphic.

The sawtooth pattern does not always reach +180 degrees and -180 degrees. This is because the measurement is made at discrete frequencies, and the data point at +180 degrees and -180 degrees may not be measured for the selected sweep.

### Types of Phase Measurements

**Complex impedance** data is information such as resistance, reactance, phase, and magnitude that can be
determined from an S11 or S22 measurement. Complex impedance data can be viewed using either the Smith Chart format or the Polar format.

**AM-to-PM conversion** is a measure of the amount of undesired phase deviation (PM) that is caused by amplitude variations (AM) of the system. AM-to-PM conversion is usually defined as the change in output phase for a 1-dB increment in the input power to an amplifier (i.e. at the 1 dB gain compression point). This is expressed in degrees-per-dB (°/dB).

**Deviation from linear phase** is a measure of phase distortion caused by a device. Ideally, the phase shift through a device is a linear function of frequency. The amount of variation from this theoretical phase shift is known as its deviation from linear phase (also called phase linearity).

**Group delay** is another way to look at phase distortion caused by a device. Group delay is a measure of transit time through a device at a particular frequency. The analyzer computes group delay from the derivative of the measured phase response.

**Deviation from Linear Phase Versus Group Delay**

Although deviation from linear phase and group delay are similar measurements, they each have their purpose.

The following are the advantages of deviation from linear phase measurements:

- Less noisy than group delay.
- Able to characterize devices that pass phase modulated signals, and show units of phase rather than units of seconds.

The following are the advantages of group delay measurements:

- More easily interpreted indication of phase distortion than deviation from linear phase.
- Able to most accurately characterize a device under test. This is because in determining group delay, the analyzer calculates the slope of the phase ripple, which is dependent on the number of ripples which occur per unit of frequency. Comparing two phase responses with equal peak-to-peak phase ripple, the response with the larger phase slope results in:
  - More group delay variation.
  - More signal distortion.

See also [Comparing the PNA Delay Functions](#).
Reverse Isolation

Reverse isolation is a measure of amplifier reverse transmission response-from output to input.

- What is Reverse Isolation
- Why Measure Reverse Isolation?
- Accuracy Considerations
- How to Measure Reverse Isolation

What is Reverse Isolation?
Reverse isolation is a measure of how well a signal applied to the device output is "isolated" from its input. The measurement of reverse isolation is similar to that of forward gain, except:

- The stimulus signal is applied to the amplifier's output port.
- The response is measured at the amplifier's input port.

The equivalent S-parameter is S12.

Why Measure Reverse Isolation?
An ideal amplifier would have infinite reverse isolation-no signal would be transmitted from the output back to the input. However, reflected signals can pass through the amplifier in the reverse direction. This unwanted reverse transmission can cause the reflected signals to interfere with the desired fundamental signal flowing in the forward direction. Therefore, reverse isolation is important to quantify.

Accuracy Considerations
Since amplifiers often exhibit high loss in the reverse direction, generally there is no need for any attenuation that may have been used to protect the port 2 receiver during forward transmission measurements. Removing the attenuation will:

- Increase the dynamic range, resulting in improved measurement accuracy.
- Require a new calibration for maximum accuracy.

The RF source power can be increased to provide more dynamic range and accuracy.

Note: With the attenuation removed and the RF source power increased, a forward sweep could damage the analyzer's port 2 receiver. Do not perform a forward sweep or use 2-port calibration unless the forward power is set low enough to avoid causing port 2 receiver compression or damage.
If the isolation of the amplifier under test is very large, the transmitted signal level may be near the noise floor or crosstalk level of the receiver. To lower the noise floor:

- Use or increase measurement averages.
- Reduce the IF bandwidth of the analyzer.

**Note:** Reducing IF bandwidth or using averaging improves measurement dynamic range and accuracy, at the expense of reduced measurement speed.

- When crosstalk levels affect the measurement accuracy, reduce the crosstalk error term by performing a response and isolation calibration. When performing the isolation part of the calibration it is important to use the same average factor and IF bandwidth during the calibration and measurement.
- The frequency response of the test setup is the dominant error in a reverse isolation measurement. Performing a thru-response measurement calibration significantly reduces this error. This calibration can be done as part of the response and isolation calibration.
- The amplifier may respond very differently at various temperatures. The tests should be done when the amplifier is at the desired operating temperature.

**How to Measure Reverse Isolation**

1. Connect the amplifier as shown in the following graphic.

2. Preset the analyzer.
3. Select an S12 measurement.
4. Select the settings for your amplifier under test.
5. Remove the amplifier and perform a thru-response calibration or a response and isolation calibration.
6. Scale the displayed measurement for optimum viewing and use a marker to measure the reverse isolation at a desired frequency.
7. Print or save the data to a disk.
Reflection Measurements

Reflection measurements are an important part of network analysis.

- **What are Reflection Measurements?**
- **Why Make Reflection Measurements?**
- **Expressing Reflected Waves**
  - **Return Loss**
  - **VSWR**
  - **Reflection Coefficient**
  - **Impedance**
- **Summary of Expressions**

---

**What are Reflection Measurements?**

To understand reflection measurements, it is helpful to think of traveling waves along a transmission line in terms of a lightwave analogy. We can imagine incident light striking some optical component like a clear lens. Some of the light is reflected off the surface of the lens, but most of the light continues on through the lens. If the lens had mirrored surfaces, then most of the light would be reflected and little or none would be transmitted.

![Reflection Diagram](image)

1. Incident  
2. Reflected  
3. Transmitted

With RF energy, reflections occur when the impedance of two mated devices are not the same. A reflection measurement is the ratio of the reflected signal to the incident signal. Network analyzers measure the incident wave with the R (for reference) channel and the reflected wave with the A channel. Therefore, reflection is often shown as the ratio of A over R (A/R). We can completely quantify the reflection characteristics of our device under test (DUT) with the amplitude and phase information available at both the A and R channel. In S-parameter terminology, S11 is a reflection measurement of port 1 of the device (the input port); S22 is a reflection measurement of the port 2 (the output port).
Why Make Reflection Measurements?

One reason we make reflection measurements to assure efficient transfer of RF power. We do this because:

1. RF energy is not cheap. When energy is reflected, that means less energy is transmitted to where it is intended to go.
2. If the reflected energy is large, it can damage components, like amplifiers.

For example, in the following graphic, the radio station on the left is not operating at peak efficiency. The amplifier impedance is not the same as the transmission line, and the transmission line impedance is not the same as the antenna. Both of these conditions cause high reflected power. This condition results in less transmitted power, and the high reflected power could damage the amplifier.

![Image of radio stations showing mismatched and matched transmission lines and antennas.]

The radio station on the right installed properly "matched" transmission line and antenna. Very little of the transmitted signal is reflected, resulting in increased broadcast power, more listeners, more advertising revenue, and more profit. The amplifier, transmission, and antenna all need to be measured to ensure that reflected power is minimized.

Expressing Reflected Waves

After making a reflection measurement, the reflection data can be expressed in a number of ways, depending on what you are trying to learn. The various expressions are all calculated by the analyzer from the same reflection measurement data. Each method of expressing reflection data can be graphically displayed in one or more formats. For more information, see display formats.

Return Loss

The easiest way to convey reflection data is return loss. Return loss is expressed in dB, and is a scalar (amplitude only) quantity. Return loss can be thought of as the absolute value or dB that the reflected signal is below the incident signal. Return loss varies between infinity for a perfect impedance match and 0 dB for an open or short circuit, or a lossless reactance. For example, using the log magnitude format on the analyzer, the measured reflection value on the screen may be -18dB. The minus sign is ignored when expressing return loss, so the component is said to have 18dB of return loss.

VSWR

Two waves traveling in opposite directions on the same transmission line cause a "standing wave". This condition can be measured in terms of the voltage standing wave ratio (VSWR or SWR for short). VSWR is defined as the maximum reflected voltage over the minimum reflected voltage at a given frequency. VSWR is a scalar (amplitude only) quantity. VSWR varies between one for a perfect match, and infinity for an open or short circuit or lossless reactance.

Reflection Coefficient

Another way of expressing reflection measurements is reflection coefficient gamma (\( \Gamma \)). Gamma includes both...
magnitude and phase.

The magnitude portion of gamma is called rho (ρ). Reflection coefficient is the ratio of the reflected signal voltage to the incident signal voltage. The range of possible values for ρ is between zero and one. A transmission line terminated in its characteristic impedance will have all energy transferred to the load; zero energy will be reflected and ρ = 0. When a transmission line terminated in a short or open circuit, all energy is reflected and ρ = 1. The value of rho is unitless.

Now for the phase information. At high frequencies, where the wavelength of the signal is smaller than the length of conductors, reflections are best thought of as waves moving in the opposite direction of the incident waves. The incident and reflected waves combine to produce a single "standing" wave with voltage that varies with position along the transmission line.

When a transmission line is terminated in its characteristic impedance (Zo) there is no reflected signal. All of the incident signal is transferred to the load, as shown in the following graphic. There is energy flowing in one direction along the transmission line.

![Standing Wave](image)

When a transmission line is terminated in a short circuit termination, all of the energy is reflected back to the source. The reflected wave is equal in magnitude to the incident wave (ρ = 1). The voltage across any short circuit is zero volts. Therefore, the voltage of the reflected wave will be 180 degrees out of phase with the incident wave, canceling the voltage at the load.

![Standing Wave](image)

When a transmission line is terminated in an open circuit termination, all of the energy is reflected back to the source. The reflected wave is equal in magnitude to the incident wave (ρ = 1). However, no current can flow in an open circuit. Therefore, the voltage of the reflected wave will be in phase with the voltage of the incident wave.

![Standing Wave](image)
When a transmission line is terminated in a 25 ohm resistor, some but not all of the incident energy will be absorbed, and some will be reflected back towards the source. The reflected wave will have an amplitude 1/3 that of the incident wave and the voltage of the two waves will be out of phase by 180 degrees at the load. The phase relationship will change as a function of distance along the transmission line from the load. The valleys of the standing wave pattern will no longer go to zero, and the peaks will be less than that of the open / short circuit.

For more information, see Phase Measurements.

**Impedance**

Impedance is another way of expressing reflection data. For more information on Impedance, see Smith Charts.

**Summary of the Expressions of Reflection Measurements:**

<table>
<thead>
<tr>
<th>No reflection</th>
<th>Full reflection</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Z_l = Z_0)</td>
<td>(Z_l = open, short)</td>
</tr>
</tbody>
</table>

- **Return loss** (RL) = -20 log(ρ), ρ = |Γ|
- **VSWR** = \( \frac{E_{max}}{E_{min}} = \frac{1 + ρ}{1 - ρ} \)
- **Reflection Coefficient** \( Γ = \frac{V_{reflected}}{V_{incident}} \)  
  \( ρ = \frac{Z_l - Z_0}{Z_l + Z_0} \)
Reflected Waves Along a Transmission line

When a sine wave from an RF signal generator is placed on a transmission line, the signal propagates toward the load. This signal, shown here in yellow, appears as a set of rotating vectors, one at each point on the transmission line.

Maximize this window for optimum viewing. Click if the applet is not visible.

In our example, the transmission line has a characteristic impedance of 50 ohms. If we choose a load of 50 ohms, then the amplitude of the signal will not vary with position along the line. Only the phase will vary along the line, as shown by the rotating vectors in yellow.

If the load impedance does not perfectly match the characteristic impedance of the line, there will be a reflected signal that propagates toward the source. At any point along the transmission line, that signal also appears to be a constant voltage whose phase is dependent upon physical position along the line.

The voltage seen at one particular point on the line will be the vector sum of the transmitted and reflected sinusoids. We can demonstrate this by looking at two examples.

Example 1: Perfect Match: 50 Ohms

Set the terminating resistor to 50 ohms by using the "down arrow" dialog box. Notice there is no reflection. We have a perfect match. Each rotating vector has a normalized amplitude of 1. If we were to observe the waveform at any point with a perfect measuring instrument, we would see equal sine wave amplitudes anywhere along the transmission line. The signal amplitudes are indicated by the green line.

Example 2: Mismatched Load: 200 Ohms

Now let's intentionally create a mismatched load. Set the terminating resistor to 200 ohms by using the down arrow. Hit the PLAY button and notice the change in the reflected waveform. If it were possible to measure just the reflected wave, we would see that its amplitude does not vary with position along the line. The only difference between the reflected (blue) signal, say at point z6 and point z4, is the phase.

But the amplitude of the resultant waveform, indicated by the standing wave (green), is not constant along the entire line because the transmitted and reflected signals (yellow and blue) combine. Since the phase between the transmitted and reflected signals varies with position along the line, the vector sums will be different, creating what's called a "standing wave".

With the load impedance at 200 ohms, a measuring device placed at point z6 would show a sine wave of constant amplitude. The sine wave at point z4 would also be of constant amplitude, but its amplitude would differ from that of the signal at point z6. And the two would be out of phase with each other. Again, the difference is shown by the green line, which indicates the amplitude at that point on the transmission line.

The impedance along the line also changes, as shown by the points labeled z1 through z7.
## Programming Guide

Two ways to find programming commands:

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1.</strong> From a simulated PNA User Interface</td>
<td><strong>2.</strong> Command Tree</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>New</strong></td>
<td><strong>GPIB / SCPI</strong></td>
<td><strong>COM</strong></td>
<td><strong>Legacy</strong></td>
<td><strong>File</strong></td>
<td><strong>View</strong></td>
</tr>
<tr>
<td><strong>See Also</strong></td>
<td><strong>Example Programs</strong></td>
<td><strong>Example Programs</strong></td>
<td><strong>Learning about GPIB</strong></td>
<td><strong>Learning about COM</strong></td>
<td></td>
</tr>
</tbody>
</table>

**What is New vs Legacy UI?**

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GPIB / SCPI</strong></td>
<td><strong>COM</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>2.</strong></td>
<td><strong>Command Tree</strong></td>
<td><strong>COM Object Model</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>See Also</strong></td>
<td><strong>Example Programs</strong></td>
<td><strong>Example Programs</strong></td>
<td><strong>Learning about GPIB</strong></td>
<td><strong>Learning about COM</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Important: Potential for programs to BREAK after upgrading to 6.0**

If you have a SCPI or COM program that does NOT work after you upgraded to 6.0, it could be for the following reason. With 6.0 we implemented a change that defaults to saving completed calibrations to Cal Registers instead of User Cal Sets. Learn how to revert to the old behavior.

- New Programming Commands
- New Remotely Specifying a Source Port
- VEE Examples with runtime installed.
- Using Macros
- Superseded / Replacement Commands
- Data Access Map
- See more PNA programming information and examples at: [http://na.tm.agilent.com/pna/programming/]
# PNA Object Model

<table>
<thead>
<tr>
<th>Application</th>
<th>Measurements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channels</td>
<td>Measurement</td>
</tr>
<tr>
<td>Channel</td>
<td>IMixer</td>
</tr>
<tr>
<td>GainCompression</td>
<td>EmbeddedLO</td>
</tr>
<tr>
<td>NoiseFigure</td>
<td>ELODiag</td>
</tr>
<tr>
<td>AuxiliaryTrigger</td>
<td>Balanced</td>
</tr>
<tr>
<td>BalancedTopology</td>
<td>Measurement</td>
</tr>
<tr>
<td>Calibrator</td>
<td>Equation</td>
</tr>
<tr>
<td>Fixturing</td>
<td>Gating</td>
</tr>
<tr>
<td>FOM</td>
<td>Marker</td>
</tr>
<tr>
<td>FOMRange</td>
<td>LimitTest</td>
</tr>
<tr>
<td>IFConfiguration</td>
<td>LimitSegment</td>
</tr>
<tr>
<td>PathConfiguration</td>
<td>Transform</td>
</tr>
<tr>
<td>PulseGenerator</td>
<td>NaWindows</td>
</tr>
<tr>
<td>Segments</td>
<td>NaWindow</td>
</tr>
<tr>
<td>Segment</td>
<td>Traces</td>
</tr>
<tr>
<td>PathElement</td>
<td>Trace</td>
</tr>
<tr>
<td>SignalProcessing</td>
<td>Capabilities</td>
</tr>
<tr>
<td>CalKit</td>
<td>E5091ATestSets</td>
</tr>
<tr>
<td>CalStandard</td>
<td>E5091ATestSet</td>
</tr>
<tr>
<td>CalManager</td>
<td>ENRFile</td>
</tr>
<tr>
<td>CalSets</td>
<td>ExternalTestSets</td>
</tr>
<tr>
<td>CalSet</td>
<td>TestSetControl</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>HWAUXIO</td>
</tr>
<tr>
<td>SMCTYPE</td>
<td>HWExternalTestSet</td>
</tr>
<tr>
<td>VMCTYPE</td>
<td>HWMaterialHandlerIO</td>
</tr>
<tr>
<td>NoiseCal</td>
<td></td>
</tr>
<tr>
<td>GainCompressionCal</td>
<td></td>
</tr>
<tr>
<td>SourcePowerCal</td>
<td>InterfaceControl</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------</td>
</tr>
<tr>
<td>PowerLossSegments</td>
<td>PathConfigurationMgr</td>
</tr>
<tr>
<td>PowerLossSegment</td>
<td>PortExtension</td>
</tr>
<tr>
<td>PowerMeterInterfaces</td>
<td>Preferences</td>
</tr>
<tr>
<td>PowerMeterInterface</td>
<td>SCPIStringParser</td>
</tr>
<tr>
<td>PowerSensors</td>
<td>TriggerSetup</td>
</tr>
<tr>
<td>PowerSensor</td>
<td></td>
</tr>
<tr>
<td>CalFactorSegs</td>
<td></td>
</tr>
<tr>
<td>PowerSensorCal</td>
<td></td>
</tr>
<tr>
<td>FactorSegment</td>
<td></td>
</tr>
</tbody>
</table>

**Legend:**
- **Object**
- **Collection**
- **Interface**

**Last Modified:**
13-Nov-2007  Replaced image with text
 Added NFA, ENRFile, and GCA
Application Object

Description
The Application object is the highest object in the PNA object model. This object presents methods and properties that affect the entire analyzer, rather than a specific channel or measurement. For example, the application object provides the GetIDString method. There’s only one ID string for the instrument, unrelated to the channel or parameter being measured. Likewise, the TriggerSignal Property is global to the instrument. You can elect to use an internally generated (free run) trigger or a manual trigger. Either way, that type of trigger generation will be used on all measurements, on all channels. Therefore, it is under the Application object.

Accessing the Application object
This object is unique in that you must create this object rather than just get a handle to it.

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Replace <analyzerName> with the full computer name of your PNA. For example, "My PNA". See Change Computer Name.

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- Getting a Handle to an Object
- Example Programs
- Superseded commands

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActivateWindow</td>
<td>IApplication</td>
<td>Makes a window object the Active Window.</td>
</tr>
<tr>
<td>AllowAllEvents</td>
<td>IApplication</td>
<td>Monitors all events</td>
</tr>
<tr>
<td>AllowEventCategory</td>
<td>IApplication</td>
<td>Monitors an event category</td>
</tr>
<tr>
<td>AllowEventMessage</td>
<td>IApplication</td>
<td>Monitors an event</td>
</tr>
<tr>
<td>AllowEventSeverity</td>
<td>IApplication</td>
<td>Monitors an event severity level</td>
</tr>
<tr>
<td>BuildHybridKit</td>
<td>IApplication</td>
<td>Defines the user kit as port1kit + port2kit.</td>
</tr>
<tr>
<td>Channel</td>
<td>IApplication</td>
<td>Stimulus values like frequency, power, IF bandwidth, and number of points.</td>
</tr>
<tr>
<td>Method Name</td>
<td>Interface</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Configure</td>
<td>IApplication9</td>
<td>Restarts as an &quot;N-port&quot; PNA using the specified multiport test set.</td>
</tr>
<tr>
<td>CreateCustomMeasurementEx</td>
<td>IApplication3</td>
<td>Creates a new custom measurement with initialization.</td>
</tr>
<tr>
<td>CreateCustomMeasurement</td>
<td>IApplication</td>
<td>Superseded with CreateCustomMeasurementEx Method</td>
</tr>
<tr>
<td>CreateMeasurement</td>
<td>IApplication</td>
<td>Creates a new measurement.</td>
</tr>
<tr>
<td>CreateSParameter</td>
<td>IApplication</td>
<td>Superseded with CreateSParameterEx Method</td>
</tr>
<tr>
<td>CreateSParameterEx</td>
<td>IApplication</td>
<td>Creates a new S-Parameter measurement with a 3-port load.</td>
</tr>
<tr>
<td>DeleteShortcut</td>
<td>IApplication</td>
<td>Removes a macro (shortcut) from the list of macros</td>
</tr>
<tr>
<td>DisallowAllEvents</td>
<td>IApplication</td>
<td>Monitors NO events</td>
</tr>
<tr>
<td>DoPrint</td>
<td>IApplication</td>
<td>Prints the screen to the active Printer.</td>
</tr>
<tr>
<td>ExecuteShortcut</td>
<td>IApplication</td>
<td>Executes a macro (shortcut) stored in the analyzer.</td>
</tr>
<tr>
<td>GetAuxIO</td>
<td>IApplication</td>
<td>Returns a handle to the AuxIO interface</td>
</tr>
<tr>
<td>GetCalManager</td>
<td>IApplication</td>
<td>Returns a handle to the CalManager interface</td>
</tr>
<tr>
<td>GetExternalTestSetIO</td>
<td>IApplication</td>
<td>Returns a handle to the ExternalTestSet IO interface</td>
</tr>
<tr>
<td>GetMaterialHandlerIO</td>
<td>IApplication</td>
<td>Returns a handle to the MaterialHandlerIO interface</td>
</tr>
<tr>
<td>GetShortcut</td>
<td>IApplication</td>
<td>Returns the title and path of the specified macro (shortcut).</td>
</tr>
<tr>
<td>LaunchCalWizard</td>
<td>IApplication</td>
<td>Launches the Cal Wizard</td>
</tr>
<tr>
<td>LaunchDialog</td>
<td>IApplication10</td>
<td>Launches the specified dialog box.</td>
</tr>
<tr>
<td>ManualTrigger</td>
<td>IApplication</td>
<td>Triggers the analyzer when TriggerSignal = naTriggerManual.</td>
</tr>
<tr>
<td>Preset</td>
<td>IApplication</td>
<td>Resets the analyzer to factory defined default settings.</td>
</tr>
<tr>
<td>PrintToFile</td>
<td>IApplication</td>
<td>Saves the screen data to bitmap (.bmp) file of the screen.</td>
</tr>
<tr>
<td>PutShortcut</td>
<td>IApplication</td>
<td>Puts a Macro (shortcut) file into the analyzer.</td>
</tr>
<tr>
<td>Quit</td>
<td>IApplication</td>
<td>Ends the Network Analyzer application.</td>
</tr>
<tr>
<td>Recall</td>
<td>IApplication</td>
<td>Recalls a measurement state, calibration state, or both from the hard drive into the analyzer.</td>
</tr>
<tr>
<td>RecallKits</td>
<td>IApplication</td>
<td>Recalls the calibration kits definitions that were stored with the SaveKits command.</td>
</tr>
<tr>
<td>Reset</td>
<td>IApplication</td>
<td>Removes all existing windows and measurements.</td>
</tr>
</tbody>
</table>
IApplication RestoreCalKitDefaults Restores the factory defaults for the specified kit.

IApplication RestoreCalKitDefaultsAll Restores the factory defaults for all kits.

IApplication Save Saves files to disk

IApplication5 SaveCitiDataData Saves UNFORMATTED trace data to .cti file.

IApplication5 SaveCitiFormattedData Saves FORMATTED trace data to .cti file.

IApplication SaveKits Saves all cal kits to disk.

IApplication SetFailOnOverRange Causes over range values to return an error code

IApplication ShowStatusBar Shows and Hides the Status Bar.

IApplication ShowStimulus Shows and Hides Stimulus information.

IApplication ShowTitleBars Shows and Hides the Title Bars.

IApplication ShowToolbar Shows and Hides the specified Toolbar.

IApplication7 UserPreset Performs a User Preset.

IApplication7 UserPresetLoadFile Loads an existing instrument state file (.sta or .cst) to be used for User Preset.

IApplication7 UserPresetSaveState Saves the current instrument settings as UserPreset.sta.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveCalKit</td>
<td>IApplication Returns a pointer to the kit identified by kitNumber.</td>
</tr>
<tr>
<td>ActiveChannel</td>
<td>IApplication Returns a handle to the Active Channel object.</td>
</tr>
<tr>
<td>ActiveMeasurement</td>
<td>IApplication Returns a handle to the Active Measurement object.</td>
</tr>
<tr>
<td>ActiveNAWindow</td>
<td>IApplication Returns a handle to the Active Window object.</td>
</tr>
<tr>
<td>ArrangeWindows</td>
<td>IApplication Sets or returns the arrangement of all the windows.</td>
</tr>
<tr>
<td>AuxiliaryTriggerCount</td>
<td>IApplication11 Returns the number of Aux trigger input / output connector pairs in the instrument.</td>
</tr>
<tr>
<td>CalKitType</td>
<td>IApplication Sets or returns the calibration kit type for to be used for calibration or for kit modification. Shared with the CalKit object.</td>
</tr>
<tr>
<td>Capabilities</td>
<td>IApplication4 Return capabilities of the remote PNA.</td>
</tr>
<tr>
<td><strong>Channels</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>CoupledMarkers</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>DisplayAutomationErrors</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>DisplayGlobalPassFail</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>E5091Testsets</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>ENRFile</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>ExternalALC</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>ExternalTestsets</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>GPIBAddress</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>GPIBMode</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>IDString</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>InterfaceControl</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>LocalLockoutState</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Measurement</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Measurements</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>MessageText</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>NaWindows</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>NoiseSourceState</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>NumberOfPorts</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Options</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>PathConfigurationManager</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Port Extensions</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Preferences</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
### ScpiStringParser
- **IAplication**
  - Provides the ability to send a SCPI command from within the COM command.

### SecurityLevel
- **IAplication4**
  - Turns ON or OFF the display of frequency information.

### SICL
- **IAplication5**
  - Allows control of the PNA via SICL

### SICLAddress
- **IAplication8**
  - Sets and returns the PNA SICL address

### SourcePowerCalibrator
- **IAplication2**
  - Allows capability for performing source power calibrations.

### SourcePowerState
- **IAplication**
  - Turns Source Power ON and OFF.

### SystemImpedanceZ0
- **IAplication**
  - Sets the analyzer impedance value.

### SystemName
- **IAplication**
  - Returns the full computer name of the PNA.

### Touchscreen
- **IAplication12**
  - Enables and disables touchscreen.

### TriggerDelay
- **IAplication**
  - Sets or returns the delay time for a trigger.

### TriggerSetup
- **IAplication4**
  - Controls triggering for the entire PNA application.

<table>
<thead>
<tr>
<th>Name</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TriggerSignal</td>
<td><strong>IAplication</strong></td>
<td>Superseded with <strong>Source Property</strong></td>
</tr>
<tr>
<td>TriggerType</td>
<td><strong>IAplication</strong></td>
<td>Superseded with <strong>Scope Property</strong></td>
</tr>
<tr>
<td>UserPresetEnable</td>
<td><strong>IAplication7</strong></td>
<td>'Checks' and 'clears' the enable box on the User Preset dialog box.</td>
</tr>
<tr>
<td>VelocityFactor</td>
<td><strong>IAplication</strong></td>
<td>Sets the velocity factor to be used with Electrical Delay, Port Extensions, and Time Domain marker distance calculations.</td>
</tr>
<tr>
<td>Visible</td>
<td><strong>IAplication</strong></td>
<td>Makes the Network Analyzer application visible or not visible.</td>
</tr>
<tr>
<td>WindowState</td>
<td><strong>IAplication</strong></td>
<td>Sets or returns the window setting of Maximized, Minimized, or Normal. Shared with the NAWindow Object</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Events</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OnCalEvent</strong></td>
<td><strong>IAplication</strong></td>
<td>Triggered by a calibration event.</td>
</tr>
<tr>
<td><strong>OnChannelEvent</strong></td>
<td><strong>IAplication</strong></td>
<td>Triggered by a channel event.</td>
</tr>
<tr>
<td><strong>OnDisplayEvent</strong></td>
<td><strong>IAplication</strong></td>
<td>Triggered by a display event.</td>
</tr>
<tr>
<td><strong>OnHardwareEvent</strong></td>
<td><strong>IAplication</strong></td>
<td>Triggered by a hardware event.</td>
</tr>
</tbody>
</table>
OnMeasurementEvent  IApplication  Triggered by a measurement event.
OnSCPIEvent  IApplication  Triggered by a SCPI event.
OnSystemEvent  IApplication  Triggered by a system event.
OnUserEvent  IApplication  For future use

IApplication History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IApplication</td>
<td>1.0</td>
</tr>
<tr>
<td>IApplication2</td>
<td>3.0</td>
</tr>
<tr>
<td>IApplication3</td>
<td>3.2</td>
</tr>
<tr>
<td>IApplication4</td>
<td>3.5</td>
</tr>
<tr>
<td>IApplication5</td>
<td>4.0</td>
</tr>
<tr>
<td>IApplication6</td>
<td>5.0</td>
</tr>
<tr>
<td>IApplication7</td>
<td>5.0</td>
</tr>
<tr>
<td>IApplication8</td>
<td>5.2</td>
</tr>
<tr>
<td>IApplication9</td>
<td>6.0</td>
</tr>
<tr>
<td>IApplication10</td>
<td>7.20</td>
</tr>
<tr>
<td>IApplication11</td>
<td>7.20</td>
</tr>
<tr>
<td>IApplication12</td>
<td>7.21</td>
</tr>
<tr>
<td>IApplication13</td>
<td>8.0</td>
</tr>
</tbody>
</table>

Last Modified:

17-Oct-2007  Updated IPathConfigMgr Prop
AuxiliaryTrigger Object

Description
These properties setup Auxiliary triggering on a channel.

Accessing the object
Use chan.AuxTrigger (n) to access the object.
where n= the connector pair to be used for Auxiliary Triggering.

- N5242A models: Use 1 or 2
- All other PNA models: Use 1

Use app.AuxiliaryTriggerCount to determine the number of auxiliary Trigger pairs on the back of a PNA.

```vba
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim AuxTrig as AuxTrigger
Set AuxTrig = chan.AuxTrigger(2)
```

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- Triggering in the PNA
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay</td>
<td>IAuxTrigger</td>
</tr>
<tr>
<td>Enable</td>
<td>IAuxTrigger</td>
</tr>
</tbody>
</table>
### IAuxTrigger

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HandshakeEnable</strong></td>
<td>Turns handshake ON / OFF.</td>
</tr>
<tr>
<td><strong>Number</strong></td>
<td>Reads the number of the Aux I/O pair being used.</td>
</tr>
<tr>
<td><strong>TriggerInPolarity</strong></td>
<td>Specifies the polarity of the trigger IN signal to which the PNA will respond.</td>
</tr>
<tr>
<td><strong>TriggerInType</strong></td>
<td>Specifies the type of Aux trigger input being supplied to the PNA.</td>
</tr>
<tr>
<td><strong>TriggerOutDuration</strong></td>
<td>Specifies the width of the pulse or the time that the Aux trigger output will be asserted</td>
</tr>
<tr>
<td><strong>TriggerOutInterval</strong></td>
<td>Specifies how often a trigger output signal is sent.</td>
</tr>
<tr>
<td><strong>TriggerOutPolarity</strong></td>
<td>Specifies the polarity of the trigger output signal being supplied by the PNA.</td>
</tr>
<tr>
<td><strong>TriggerOutPosition</strong></td>
<td>Specifies whether the Aux trigger out signal is sent Before or After the acquisition.</td>
</tr>
</tbody>
</table>

### IAuxTrigger History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAuxTrigger</td>
<td>7.2</td>
</tr>
</tbody>
</table>
BalancedMeasurement Object

Description
These properties set the measurement type that is used with balanced topologies.
Use the BalancedTopology Object to set the topology and port mappings for the DUT.

Accessing the BalancedMeasurement object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim balMeas As BalancedMeasurement
Set balMeas = app.ActiveMeasurement.BalancedMeasurement
```

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- About Balanced Measurements
- Example Programs

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BalancedMode</td>
<td>IBalancedMeasurement</td>
<td>Sets and returns whether the balanced transform is ON or OFF.</td>
</tr>
<tr>
<td>BalancedTopology</td>
<td>IBalancedMeasurement</td>
<td>Sets and returns the topology of a balanced DUT.</td>
</tr>
<tr>
<td>BBalMeasurement</td>
<td>IBalancedMeasurement</td>
<td>Sets and returns the measurement for the Balanced - Balanced topology.</td>
</tr>
</tbody>
</table>
IBalancedMeasurement History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBalancedMeasurement</td>
<td>5.0</td>
</tr>
</tbody>
</table>
BalancedTopology Object

Description
The DUTTopology property sets and returns the topology of a balanced DUT.
The following methods set the port mappings for the DUT.
The remaining properties return the port mappings for the DUT.
Use the BalancedMeasurement object to set the measurement type.

Accessing the BalancedTopology object

Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel

Dim balTopology as BalancedTopology
Set balTopology = chan.BalancedTopology

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- About Balanced Measurements
- Example Programs

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>See History</strong></td>
</tr>
<tr>
<td>SetBBPorts</td>
<td>IBalancedTopology</td>
<td>Sets the physical port mappings for the Balanced - Balanced DUT topology.</td>
</tr>
<tr>
<td>SetSBPorts</td>
<td>IBalancedTopology</td>
<td>Sets the physical port mappings for the Single-Ended - Balanced DUT topology.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB_BalPort1Negative</td>
<td>IBalancedTopology</td>
<td>Returns the PNA port number that is connected to the Negative side of the DUT's logical Port 1.</td>
</tr>
</tbody>
</table>
**BB_BalPort1Positive**  
*IBalancedTopology*  
Returns the first positive balanced port number in the Balanced - Balanced topology.

**BB_BalPort2Negative**  
*IBalancedTopology*  
Returns the second negative balanced port number in the Balanced - Balanced topology.

**BB_BalPort2Positive**  
*IBalancedTopology*  
Returns the second positive balanced port number in the Balanced - Balanced topology.

**DUTTopology**  
*IBalancedTopology*  
Sets and returns the device topology setting.

**SB_BalPortNegative**  
*IBalancedTopology*  
Returns the negative balanced port number in the Single-Ended - Balanced topology.

**SB_BalPortPositive**  
*IBalancedTopology*  
Returns the positive balanced port number in the Single-Ended - Balanced topology.

**SB_SEPort**  
*IBalancedTopology*  
Returns the single ended port number in the Single-Ended - Balanced topology.

**SSB_BalPortNegative**  
*IBalancedTopology*  
Returns the negative balanced port number in the Single-Ended - Single-Ended - Balanced topology.

**SSB_BalPortPositive**  
*IBalancedTopology*  
Returns the positive balanced port number in the Single-Ended - Single-Ended - Balanced topology.

**SSB_SEPort1**  
*IBalancedTopology*  

**SSB_SEPort2**  
*IBalancedTopology*  

---

**BalancedTopology History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBalancedTopology</td>
<td>5.0</td>
</tr>
</tbody>
</table>

---

712
**CalFactorSegments Collection**

**Description**
A collection object that provides a mechanism for iterating through the segments of a power sensor cal factor table.

**Accessing the CalFactorSegments collection**

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calFact As CalFactorSegments
```

**See Also:**
- [PowerSensorCalFactorSegment Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add</strong></td>
<td>Adds a PowerSensorCalFactorSegment object to the collection</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>Use to get a handle to a PowerSensorCalFactorSegment object in the collection.</td>
</tr>
<tr>
<td><strong>Remove</strong></td>
<td>Removes an object from the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Count</strong></td>
<td>Returns the number of objects in the collection.</td>
</tr>
<tr>
<td><strong>Parent</strong></td>
<td>Returns a handle to the Parent object (PowerSensor) of this collection.</td>
</tr>
</tbody>
</table>
Calibrator Object

See Also

- Example Programs
- Calibrator Methods and Properties
- ICalData Interface for putting and getting typed Calibration data.
- Superseded commands

Description

The Calibrator object, a child of the channel, is used to perform an Unguided calibration.

**Note:** You can NOT perform a full 3 or 4-port using the Calibrator object; you must use the GuidedCalibration object.

There must be a measurement present for the calibrator to use or you will receive a "no measurement found" error. Therefore, to perform a 2-port cal, you must have any S-parameter measurement on the channel. For a 1-port measurement, you must have the measurement (S11 or S22) on the channel. The same is true for a response measurement.

There are a number of approaches to calibration with the calibrator object:

- You can collect data yourself and download it to the ACQUISITION buffer. The acquisition buffer holds the actual measured data for each standard. See the PNA data map.

  1. Calibrator. SetCalInfo
  2. Connect a standard
  3. Trigger a sweep
  4. Retrieve the data for the standard
  5. Download the data - calibrator. putStandard
  6. Repeat for each standard
  7. Calibrator. CalculateErrorCoefficients

- You can tell the calibrator to acquire a standard. In this case, the calibrator collects the data and places it in the ACQUISITION buffer.

  1. Calibrator. SetCalInfo
  2. Connect a standard
  3. Calibrator. AcquireCalStandard2
  4. Repeat for each standard
  5. Calibrator. CalculateErrorCoefficients
You can put previously-retrieved error terms in the error correction buffer.

1. `PutErrorTerm`
2. Repeat for each term
3. `Measurement.Caltype` = pick one

You can also "piece together" a 2-port cal from two 1-port cals (S11 and S22) and four response (thru) cals. The system will detect that all the standards needed for a 2-port cal have been acquired even though they may not have gathered at the same time.

**Accessing the Calibrator object**

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim cal As ICalibrator
Set cal = app.ActiveChannel.Calibrator
```

**See Also:**

- **PNA Automation Interfaces**
- **The PNA Object Model**
- Learn about reading and writing Calibration data.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>AcquireCalConfidenceCheckECAL</code></td>
<td>ICalibrator</td>
<td>Superseded with <code>AcquireCalConfidenceCheckECALEx</code></td>
</tr>
<tr>
<td><code>AcquireCalConfidenceCheckECALEx</code></td>
<td>ICalibrator4</td>
<td>Transfers ECAL confidence data into analyzer memory</td>
</tr>
<tr>
<td><code>AcquireCalStandard</code></td>
<td>ICalibrator</td>
<td>Superseded with <code>AcquireCalStandard2</code></td>
</tr>
<tr>
<td><code>AcquireCalStandard2</code></td>
<td>ICalibrator</td>
<td>Causes the analyzer to measure a calibration standard. Also provides for sliding load.</td>
</tr>
<tr>
<td><code>CalculateErrorCoeffecients</code></td>
<td>ICalibrator</td>
<td>Generates Error Terms from standard and actual data in the error correction buffer.</td>
</tr>
<tr>
<td><code>DoECAL1Port</code></td>
<td>ICalibrator</td>
<td>Superseded with <code>DoECAL1PortEx</code></td>
</tr>
<tr>
<td><code>DoECAL1PortEx</code></td>
<td>ICalibrator4</td>
<td>Completes a 1 port ECAL</td>
</tr>
<tr>
<td><code>DoECAL2Port</code></td>
<td>ICalibrator</td>
<td>Superseded with <code>DoECAL2PortEx</code></td>
</tr>
<tr>
<td>Function</td>
<td>Interface</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DoECAL2PortEx</td>
<td>ICalibrator4</td>
<td>Completes a 2 port ECAL</td>
</tr>
<tr>
<td>DoneCalConfidenceCheckECAL</td>
<td>ICalibrator</td>
<td>Concludes an ECAL confidence check</td>
</tr>
<tr>
<td>DoReceiverPowerCal</td>
<td>ICalibrator5</td>
<td>Perform a receiver power cal.</td>
</tr>
<tr>
<td>GetECALModuleInfo</td>
<td>ICalibrator</td>
<td>Superseded with Get ECALModuleInfoEx</td>
</tr>
<tr>
<td>Get ECALModuleInfoEx</td>
<td>ICalibrator4</td>
<td>Returns information about the attached module</td>
</tr>
<tr>
<td>getErrorTerm</td>
<td>ICalibrator</td>
<td>Superseded with GetErrorTermByString</td>
</tr>
<tr>
<td>getStandard</td>
<td>ICalibrator</td>
<td>Superseded with GetStandardByString</td>
</tr>
<tr>
<td>putErrorTerm</td>
<td>ICalibrator</td>
<td>Superseded with PutErrorTermByString</td>
</tr>
<tr>
<td>putStandard</td>
<td>ICalibrator</td>
<td>Superseded with PutStandardByString</td>
</tr>
<tr>
<td>SaveCalSets</td>
<td>ICalibrator</td>
<td>Superseded with CalSet.Save</td>
</tr>
<tr>
<td>setCalInfo</td>
<td>ICalibrator</td>
<td>Specifies the type of calibration and prepares the internal state for the rest of the calibration.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcquisitionDirection</td>
<td>ICalibrator</td>
<td>Specifies the direction in a 2-Port cal using one set of standards.</td>
</tr>
<tr>
<td>ECALCharacterizationEx</td>
<td>ICalibrator2</td>
<td>Superseded with ECALCharacterizationEx</td>
</tr>
<tr>
<td>ECALCharacterizationEx</td>
<td>ICalibrator4</td>
<td>Specifies which set of characterization data within an ECal module will be used for ECal operations with that module.</td>
</tr>
<tr>
<td>ECALCharacterizationIndexList</td>
<td>ICalibrator6</td>
<td>Returns a list of characterizations stored in the specified ECal module.</td>
</tr>
<tr>
<td>ECAL Isolation</td>
<td>ICalibrator</td>
<td>Specifies whether the acquisition of the ECal calibration should include isolation or not.</td>
</tr>
<tr>
<td>ECALModuleNumberList</td>
<td>ICalibrator6</td>
<td>Returns a list of index numbers to be used for referring to the ECal modules that are currently attached to the PNA.</td>
</tr>
<tr>
<td>ECALPortMap</td>
<td>ICalibrator3</td>
<td>Superseded with ECALPortMapEx</td>
</tr>
<tr>
<td>ECALPortMapEx</td>
<td>ICalibrator4</td>
<td>Specifies which ports of the ECal module are connected to which ports of the PNA.</td>
</tr>
<tr>
<td>IsECALModuleFound</td>
<td>ICalibrator</td>
<td>Superseded with IsECALModuleFoundEx</td>
</tr>
</tbody>
</table>
IsECALModuleFoundEx ICalibrator4 Superseded with ECALCharacterizationIndexList and ECALModuleNumberList

IsolationAveragingIncrement ICalibrator7 Value to increase the channel's averaging factor.

OrientECALModule ICalibrator3 Specifies if the PNA should perform orientation of the ECal module during calibration.

Simultaneous2PortAcquisition ICalibrator Allows the use of 2 sets of standards at the same time.

ICalibrator History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalibrator</td>
<td>1.0</td>
</tr>
<tr>
<td>ICalibrator2</td>
<td>3.1</td>
</tr>
<tr>
<td>ICalibrator3</td>
<td>3.1</td>
</tr>
<tr>
<td>ICalibrator4</td>
<td>3.5</td>
</tr>
<tr>
<td>ICalibrator5</td>
<td>5.0</td>
</tr>
<tr>
<td>ICalibrator6</td>
<td>5.26</td>
</tr>
<tr>
<td>ICalibrator6</td>
<td>7.21</td>
</tr>
</tbody>
</table>

ICalData Interface

Description
Contains methods for putting Calibration data in and getting Calibration data out of the analyzer using typed data. This interface transfers data more efficiently than variant data. However, this interfaces is only usable from VB6, C, & C++. All other programming languages must use the ICalSet interface.

There is also an ICalData Interface on the CalSet Object
Learn about reading and writing Calibration data.
### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getErrorTermComplex</code></td>
<td>Retrieves error term data</td>
</tr>
<tr>
<td><code>getStandardComplex</code></td>
<td>Retrieves calibration data from the acquisition data buffer (before error-terms are applied).</td>
</tr>
<tr>
<td><code>putErrorTermComplex</code></td>
<td>Puts error term data</td>
</tr>
<tr>
<td><code>putStandardComplex</code></td>
<td>Puts calibration data into the acquisition data buffer (before error-terms are applied).</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

### ICalData History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalData</td>
<td>1.0</td>
</tr>
</tbody>
</table>
CalKit Object

Description
The calkit object provides the properties and methods to access and modify a calibration kit. The calkitType property can be set from two objects:

- Application object - app.calKitType
- CalKit object - calKit.calKitType

Both of these commands specify or read the calibration kit type. When specified, the cal kit also becomes the Active cal kit.

Accessing a CalKit object
To get a handle to a cal kit, use **app.ActiveCalKit**.

The calKit object behaves differently from other objects in the system in that you can only have a handle to one cal kit -- the active calkit. Therefore, when you change the calkitType from either the Application object or the CalKit object, you may also be changing the object to which you may have other references.

For example, the following example specifies two calKit type objects and in turn, assigns them to two different variables: ck1 and ck2.

```vba
Dim app As AgilentPNA835x.Application
Dim ck1 As calKit
Dim ck2 As calKit

Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
app.CalKitType = naCalKit_User1
Set ck1 = app.ActiveCalKit
ck1.Name = "My CalKit1"

app.CalKitType = naCalKit_User2
Set ck2 = app.ActiveCalKit
ck2.Name = "My CalKit2"

Print "ck1: " & ck1.Name
Print "ck2: " & ck2.Name
```

When the pointer to each of these kits is read (printed), they each have a pointer to the last kit to be assigned to the Active cal kit:

ck1: My CalKit2
ck2: My CalKit2

See Also:

- PNA Automation Interfaces
- The PNA Object Model
### Example Programs

**Bold** Methods or Properties provide access to a child object

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getCalStandard</code></td>
<td>Returns a handle to a calibration standard for modifying its definitions.</td>
</tr>
<tr>
<td><code>GetStandardsForClass</code></td>
<td>Returns the calibration standard numbers for a specified calibration class.</td>
</tr>
<tr>
<td><code>SetStandardsForClass</code></td>
<td>Sets the calibration standard numbers for a specified calibration class.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
</table>
| `CalKitType`             | Sets or returns the calibration kit type for to be used for calibration or for kit modification.  
|                          | Shared with the Application object.                                         |
| `Name`                   | Sets and returns the name of the cal kit                                     |
| `PortLabel`              | Labels the ports for the kit; only affects the cal wizard annotation.       |
| `StandardForClass`       | **Superseded with** Use `GetStandardForClass` and `SetStandardForClass`.    |
|                          | Maps a standard device to a cal class.                                      |

### ICalKit History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalKit</td>
<td>1.0</td>
</tr>
</tbody>
</table>
CalManager Object

Description
Use this interface to list, save, and delete Cal Sets.

Accessing the CalManager object
Get a handle to a the CalManager with the app.GetCalManager Method.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim mgr as ICalManager
Set mgr = app.GetCalManager
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs
- Superseded commands

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowChannelToSweepDuringCalAcquisition</td>
<td>ICalManager5</td>
<td>Specifies the channel to sweep during a Calibration.</td>
</tr>
<tr>
<td>CreateCalSet</td>
<td>ICalManager</td>
<td>Creates a new Cal Set</td>
</tr>
<tr>
<td>CreateCustomCal</td>
<td>ICalManager2</td>
<td>Creates an FCA cal object.</td>
</tr>
<tr>
<td>CreateCustomCalEx</td>
<td>ICalManager5</td>
<td>Creates a custom cal object.</td>
</tr>
<tr>
<td>DeleteCalSet</td>
<td>ICalManager</td>
<td>Deletes a Cal Set</td>
</tr>
<tr>
<td>DisplayNAWindowDuringCalAcquisition</td>
<td>ICalManager5</td>
<td>Set the 'show' state of the window to be displayed during</td>
</tr>
<tr>
<td>DisplayOnlyCalWindowDuringCalAcquisition</td>
<td>ICalManager5</td>
<td>Clears the flags for windows to be shown during calibrat</td>
</tr>
<tr>
<td>EnumerateCalSets</td>
<td>ICalManager4</td>
<td>Returns an array of Cal Set names being stored on the I</td>
</tr>
<tr>
<td>GetCalSetByGUID</td>
<td>ICalManager</td>
<td>Get a handle to a Cal Set</td>
</tr>
<tr>
<td>Method</td>
<td>Interface</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GetCalSetCatalog</td>
<td>ICalManager</td>
<td><strong>Superseded with</strong> EnumerateCalSets</td>
</tr>
<tr>
<td>GetCalSetUsageInfo</td>
<td>ICalManager</td>
<td>Returns the Cal Set ID and Error Term ID currently in use</td>
</tr>
<tr>
<td>GetCalTypes</td>
<td>ICalManager2</td>
<td>Query for a list of available calibration types.</td>
</tr>
<tr>
<td>GetRequiredEtermNames</td>
<td>ICalManager2</td>
<td>Returns an array of strings specifying the error terms required by the Cal Type correction algorithm.</td>
</tr>
<tr>
<td>SaveCalSets</td>
<td>ICalManager</td>
<td><strong>Superseded with</strong> CalSet.Save</td>
</tr>
<tr>
<td>SweepOnlyCalChannelDuringCal Acquisition</td>
<td>ICalManager5</td>
<td>Clears ALL flags for channels to sweep during calibration.</td>
</tr>
</tbody>
</table>

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cal Sets</td>
<td>ICalManager</td>
<td>Collection for iterating through all the Cal Sets in the analyzer.</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>ICalManager3</td>
<td>Used to perform a Guided Calibration.</td>
</tr>
</tbody>
</table>

**ICalManager History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalManager</td>
<td>2.0</td>
</tr>
<tr>
<td>CalManager2</td>
<td>3.1</td>
</tr>
<tr>
<td>CalManager3</td>
<td>3.5</td>
</tr>
<tr>
<td>CalManager4</td>
<td>5.0</td>
</tr>
<tr>
<td>ICalManager5</td>
<td>8.0</td>
</tr>
</tbody>
</table>
**CalSet Object**

See [ICalData Interface](#) for putting and getting typed Cal Set data.

**Description**

Use this interface to query and or change the contents of a Cal Set.

**Accessing the CalSet object**

Get a handle to a CalSet object by using the CalSets collection. This is done through the CalManager object with the app.GetCalManager Method.

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calst As ICalSet
Set calst = app.GetCalManager.CalSets.Item(1)
```

**See Also:**

- PNA Automation Interfaces
- The PNA Object Model
- Reading and Writing Calibration data
- Example Programs
- Superseded commands

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CloseCalSet</strong></td>
<td>ICalSet</td>
<td><strong>Obsolete</strong> - No longer necessary.</td>
</tr>
<tr>
<td><strong>ComputeErrorTerms</strong></td>
<td>ICalSet</td>
<td>Computes error terms for the CalType specified by a preceding OpenCal Set call.</td>
</tr>
<tr>
<td><strong>Copy</strong></td>
<td>ICalSet</td>
<td>Creates a new Cal Set and copies the current Cal Set data into it.</td>
</tr>
<tr>
<td><strong>getErrorTerm</strong></td>
<td>ICalSet</td>
<td><strong>Superseded with</strong> <code>getErrorTermByString</code></td>
</tr>
<tr>
<td><strong>getErrorTermByString</strong></td>
<td>ICalSet2</td>
<td>Returns variant error term data by specifying the string name of the error term.</td>
</tr>
<tr>
<td><strong>getErrorTermList</strong></td>
<td>ICalSet</td>
<td><strong>Superseded with</strong> <code>getErrorTermList2</code></td>
</tr>
<tr>
<td><strong>getErrorTermList2</strong></td>
<td>ICalSet2</td>
<td>Returns a list of error term names found in a calset.</td>
</tr>
<tr>
<td>Method</td>
<td>Class</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GetGUID</td>
<td>ICalSet</td>
<td>Returns the GUID identifying a Cal Set</td>
</tr>
<tr>
<td>getStandard</td>
<td>ICalSet</td>
<td>Superseded with getStandardByString</td>
</tr>
<tr>
<td>getStandardByString</td>
<td>ICalSet2</td>
<td>Returns variant standard acquisition data by specifying the string name of the standard.</td>
</tr>
<tr>
<td>getStandardsList</td>
<td>ICalSet</td>
<td>Superseded with getStandardList2</td>
</tr>
<tr>
<td>getStandardList2</td>
<td>ICalSet2</td>
<td>Returns a list of standard names found in a Cal Set.</td>
</tr>
<tr>
<td>HasCalType</td>
<td>ICalSet</td>
<td>Verifies that the Cal Set object contains the error terms required to apply the specified CalType to an appropriate measurement.</td>
</tr>
<tr>
<td>OpenCalSet</td>
<td>ICalSet</td>
<td>Obsolete - No longer necessary.</td>
</tr>
<tr>
<td>putErrorTerm</td>
<td>ICalSet</td>
<td>Superseded with putErrorTermByString</td>
</tr>
<tr>
<td>putErrorTermByString</td>
<td>ICalSet2</td>
<td>Writes variant error term data by specifying the string name of the error term.</td>
</tr>
<tr>
<td>putStandard</td>
<td>ICalSet</td>
<td>Superseded with putStandardByString</td>
</tr>
<tr>
<td>putStandardByString</td>
<td>ICalSet2</td>
<td>Writes variant standard acquisition data by specifying the string name of the standard.</td>
</tr>
<tr>
<td>Save</td>
<td>ICalSet</td>
<td>Saves the current Cal Set to disk.</td>
</tr>
<tr>
<td>StringToNACalClass</td>
<td>ICalSet</td>
<td>Converts string values from GetStandardsList into enumeration data</td>
</tr>
<tr>
<td>StringToNAErrorTerm2</td>
<td>ICalSet</td>
<td>Converts string values from GetErrorTermList into enumeration data</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlternateSweep</td>
<td>ICalSet3</td>
<td>Reads sweep either alternate or chopped.</td>
</tr>
<tr>
<td>Attenuator</td>
<td>ICalSet3</td>
<td>Returns the value of the attenuator control for the specified port number.</td>
</tr>
<tr>
<td>AttenuatorMode</td>
<td>ICalSet3</td>
<td>Returns the mode of operation (auto or manual) of the attenuator control for the specified port number.</td>
</tr>
<tr>
<td>CouplePorts</td>
<td>ICalSet3</td>
<td>Returns state of couple ports (ON or OFF)</td>
</tr>
<tr>
<td>CWFrequency</td>
<td>ICalSet3</td>
<td>Returns CW Frequency</td>
</tr>
<tr>
<td>Description</td>
<td>ICalSet</td>
<td>Set or return the descriptive string assigned to the Cal Set</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------</td>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td>DwellTime</td>
<td>ICalSet3</td>
<td>Returns the dwell time for the channel.</td>
</tr>
<tr>
<td>FrequencyOffsetCWOVERRIDE</td>
<td>ICalSet3</td>
<td>Reads state of CW Override (ON or OFF)</td>
</tr>
<tr>
<td>FrequencyOffsetDivisor</td>
<td>ICalSet3</td>
<td>Reads Frequency Offset Divisor value</td>
</tr>
<tr>
<td>FrequencyOffsetFrequency</td>
<td>ICalSet3</td>
<td>Reads Offset Frequency</td>
</tr>
<tr>
<td>FrequencyOffsetMultiplier</td>
<td>ICalSet3</td>
<td>Reads Frequency Offset Multiplier value</td>
</tr>
<tr>
<td>FrequencyOffsetState</td>
<td>ICalSet3</td>
<td>Reads Frequency Offset state (ON or OFF)</td>
</tr>
<tr>
<td>IFBandwidth</td>
<td>ICalSet3</td>
<td>Reads IF Bandwidth of the channel</td>
</tr>
<tr>
<td>LastModified</td>
<td>ICalSet3</td>
<td>Reads the time stamp of when the file was last modified</td>
</tr>
<tr>
<td>Name</td>
<td>ICalSet4</td>
<td>Sets and returns the Cal Set name.</td>
</tr>
<tr>
<td>NumberOfPoints</td>
<td>ICalSet3</td>
<td>Returns the Number of Points of the channel.</td>
</tr>
<tr>
<td>PowerSlope</td>
<td>ICalSet3</td>
<td>Returns the Power Slope value</td>
</tr>
<tr>
<td>ReceiverAttenuator</td>
<td>ICalSet3</td>
<td>Returns the value of the specified receiver attenuator control.</td>
</tr>
<tr>
<td>StartFrequency</td>
<td>ICalSet3</td>
<td>Returns the start frequency of the channel.</td>
</tr>
<tr>
<td>StartPower</td>
<td>ICalSet3</td>
<td>Returns the start power of the PNA when sweep type is set to Power Sweep.</td>
</tr>
<tr>
<td>StimulusValues</td>
<td>ICalSet3</td>
<td>Returns x-axis values for stimulus or response frequencies</td>
</tr>
<tr>
<td>StopFrequency</td>
<td>ICalSet3</td>
<td>Returns the stop frequency of the channel.</td>
</tr>
<tr>
<td>StopPower</td>
<td>ICalSet3</td>
<td>Returns the stop power of the PNA when sweep type is set to Power Sweep.</td>
</tr>
<tr>
<td>SweepGenerationMode</td>
<td>ICalSet3</td>
<td>Returns the method being used to generate a sweep: analog or stepped.</td>
</tr>
<tr>
<td>SweepTime</td>
<td>ICalSet3</td>
<td>Returns the sweep time of the analyzer.</td>
</tr>
<tr>
<td>SweepType</td>
<td>ICalSet3</td>
<td>Returns the type of X-axis sweep that is performed on a channel.</td>
</tr>
<tr>
<td>TestPortPower</td>
<td>ICalSet3</td>
<td>Returns the RF power level for the channel.</td>
</tr>
</tbody>
</table>

**ICalSet History**
# ICalData Interface

## Description

Use this interface as an alternative to the ICalSet Interface to avoid using variants when transmitting data to and from the Cal Set.

Learn about reading and writing Calibration data.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get ErrorTermComplex</td>
<td>ICalData2</td>
<td>Superseded with getErrorTermComplexByString</td>
</tr>
<tr>
<td>getErrorTermComplexByString</td>
<td>ICalData3</td>
<td>Returns typed error term data by specifying the string name of the error term.</td>
</tr>
<tr>
<td>getStandardComplex</td>
<td>ICalData2</td>
<td>Superseded with getStandardComplexByString</td>
</tr>
<tr>
<td>getStandardComplexByString</td>
<td>ICalData3</td>
<td>Returns typed standard acquisition data by specifying the string name of the standard.</td>
</tr>
<tr>
<td>put ErrorTermComplex</td>
<td>ICalData2</td>
<td>Superseded with putErrorTermComplexByString</td>
</tr>
<tr>
<td>putErrorTermComplexByString</td>
<td>ICalData3</td>
<td>Writes typed error term data by specifying the string name of the error term.</td>
</tr>
<tr>
<td>putStandardComplex</td>
<td>ICalData2</td>
<td>Superseded with putStandardComplexByString</td>
</tr>
<tr>
<td>putStandardComplexByString</td>
<td>ICalData3</td>
<td>Writes typed standard acquisition data by specifying the string name of the standard.</td>
</tr>
</tbody>
</table>

## Properties

| Properties | Description |
|------------|-------------|-------------|

---

Introduced with PNA Rev:

- ICalSet 2.0
- ICalSet2 3.0
- ICalSet3 3.2
- ICalSet4 6.0
History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalData2</td>
<td>2.0</td>
</tr>
<tr>
<td>ICalData3</td>
<td>3.1</td>
</tr>
</tbody>
</table>

The original ICalData Interface was introduced with PNA 1.0 on the Calibrator Object.

Last modified:

Nov. 1, 2006    New start and stop freq commands added
Cal Sets Collection

Description
A collection object that provides a mechanism for iterating through all the Cal Sets in the analyzer. There is no ordering to the items in the collection. Therefore make no assumptions about the formatting of the collection. For the Item and Remove methods, you can specify either the Cal Set string name, or the integer item of the Cal Set in the collection.

Accessing the CalSets collection
Get a handle to the CalSets collection through the CalManager object with the app.GetCalManager Method.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calsts As CalSets
Set calsts = app.GetCalManager.CalSets
```

See Also:
- Cal Set Object
- Collections in the Analyzer
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Returns a handle to a Cal Set object in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Deletes the Cal Set residing at position index in the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of Cal Sets in the collection.</td>
</tr>
</tbody>
</table>

Last Modified:
30-Oct-2007 added item and remove note.
CalStandard Object

Description
Contains all of the settings that are required to modify a calibration standard.

Accessing the CalStandard object
Get a handle to a standard with the calkit.GetCalStandard Method.

```vbs
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim std As ICalStandard
Set std = app.ActiveCalKit.GetCalStandard(1)
std.Delay = 0.00000003
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Reading and Writing Calibration data
- Example Programs

Methods
None

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0</td>
<td>ICalStandard</td>
<td>Sets and Returns the C0 (C-zero) value (the first capacitance value) for the calibration standard, when the Type is set to &quot;naOpen&quot;.</td>
</tr>
<tr>
<td>C1</td>
<td>ICalStandard</td>
<td>Sets and Returns the C1 value (the second capacitance value) for the calibration standard, when the Type is set to &quot;naOpen&quot;.</td>
</tr>
<tr>
<td>C2</td>
<td>ICalStandard</td>
<td>Sets and Returns the C2 value (the third capacitance value) for the calibration standard, when the Type is set to &quot;naOpen&quot;.</td>
</tr>
<tr>
<td>C3</td>
<td>ICalStandard</td>
<td>Sets and Returns the C3 value (the fourth capacitance value) for the calibration standard, when the Type is set to &quot;naOpen&quot;.</td>
</tr>
<tr>
<td>Delay</td>
<td>ICalStandard</td>
<td>Sets and Returns the electrical delay value for the calibration standard.</td>
</tr>
</tbody>
</table>
**L0** ICalStandard  
Sets and Returns the L0 (L-zero) value (the first inductance value) for the calibration standard, when the Type is set to "naShort".

**L1** ICalStandard  
Sets and Returns the L1 value (the second inductance value) for the calibration standard, when the Type is set to "naShort".

**L2** ICalStandard  
Sets and Returns the L2 value (the third inductance value) for the calibration standard, when the Type is set to "naShort".

**L3** ICalStandard  
Sets and Returns the L3 value (the third inductance value) for the calibration standard, when the Type is set to "naShort".

**Label** ICalStandard  
Sets and Returns the label for the calibration standard.

**loss** ICalStandard  
Sets and Returns the insertion loss for the calibration standard.

**Maximum Frequency** ICalStandard  
Sets and Returns the maximum frequency for the calibration standard.

**Medium** ICalStandard  
Sets and Returns the media type of the calibration standard.

**Minimum Frequency** ICalStandard  
Sets and Returns the minimum frequency for the calibration standard.

**Type** ICalStandard  
Sets and Returns the type of calibration standard. Selections are: naOpen, naShort, naLoad, naThru, naArbitraryImpedance and naSliding.

**TZReal** ICalStandard2  
Sets and Returns the TZReal value (the Real Terminal Impedance value) for the calibration standard, when the Type is set to "naArbitraryImpedance".

**TZImag** ICalStandard2  
Sets and Returns the TZImag value (the Imaginary Terminal Impedance value) for the calibration standard, when the Type is set to "naArbitraryImpedance".

**Z0** ICalStandard  
Sets and Returns the characteristic impedance for the calibration standard.

### ICalStandard History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalStandard</td>
<td>1.0</td>
</tr>
<tr>
<td>CalStandard2</td>
<td>3.0</td>
</tr>
</tbody>
</table>
Capabilities Object

Description
These properties return capabilities of the remote PNA.

Accessing the Capabilities object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim cap As Capabilities
Set cap = app.Capabilities

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- ICapabilities History
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetPortNumber Method</td>
<td>ICapabilities4</td>
<td>Returns the port number for the specified string port name.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FirmwareMajorRevision</td>
<td>ICapabilities</td>
</tr>
<tr>
<td>FirmwareMinorRevision</td>
<td>ICapabilities</td>
</tr>
<tr>
<td>FirmwareSeries</td>
<td>ICapabilities</td>
</tr>
<tr>
<td>GPIBPortCount</td>
<td>ICapabilities3</td>
</tr>
<tr>
<td>InternalTestsetPortCount</td>
<td>ICapabilities</td>
</tr>
<tr>
<td>IsFrequencyOffsetPresent</td>
<td>ICapabilities</td>
</tr>
<tr>
<td>IsReceiverStepAttenuatorPresent</td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>
**ICapabilities History**

<table>
<thead>
<tr>
<th>I Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICapabilities</td>
<td>3.5</td>
</tr>
<tr>
<td>ICapabilities2</td>
<td>5.23</td>
</tr>
<tr>
<td>ICapabilities3</td>
<td>6.0</td>
</tr>
<tr>
<td>ICapabilities4</td>
<td>7.20</td>
</tr>
</tbody>
</table>
Channel Object

See [SourcePowerCalData Interface](#) for putting and getting typed source power calibration data.

**Description**

The channel object is like the engine that produces data. Channel settings consist of stimulus values like frequency, power, IF bandwidth, and number of points.

**Accessing the Channel object**

You can get a handle to a channel in a number of ways. But first you have to make sure that the channel exists. When you first startup the analyzer, there is one S11 measurement on channel 1. Thus there is only one channel in existence. You can do the following:

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chan As IChannel
Set chan = app.ActiveChannel
```

or

```vba
Set chan = app.Channels(2)
```

The first method returns the channel object that is driving the active measurement. If there is no measurement, there may not be a channel. Once a channel is created, it does not go away. So if there once was a measurement (hence a channel), the channel will still be available.

If there is no channel you can create one in a couple ways. You can do the following:

```vba
Pna.CreateMeasurement( ch1, "S11", port1, window2)
```

or

```vba
Pna.Channels.Add(2)
```

The latter will have no visible effect on the analyzer. It will simply create channel 2 if it does not already exist.

**See Also:**

- PNA Automation Interfaces
- The PNA Object Model
- Reading and Writing Calibration data.
- Example Programs
- Superseded commands

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>IChannel</td>
<td>Aborts the current measurement sweep on the channel.</td>
</tr>
<tr>
<td>Method</td>
<td>Interface</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ApplySourcePowerCorrectionTo</td>
<td>IChannel11</td>
<td>Copies an existing Source Power Calibration to another channel.</td>
</tr>
<tr>
<td>AveragingRestart</td>
<td>IChannel</td>
<td>Clears and restarts averaging of the measurement data.</td>
</tr>
<tr>
<td>Continuous</td>
<td>IChannel</td>
<td>The channel continuously responds to trigger signals.</td>
</tr>
<tr>
<td>CopyToChannel</td>
<td>IChannel</td>
<td>Sets up another channel as a copy of this objects channel.</td>
</tr>
<tr>
<td>GetErrorCorrection</td>
<td>IChannel8</td>
<td>Returns the channel error correction state.</td>
</tr>
<tr>
<td>GetNumberOfGroups</td>
<td>IChannel3</td>
<td>Returns the <strong>number of groups</strong> a channel has yet to acquire.</td>
</tr>
<tr>
<td>getSourcePowerCalData</td>
<td>IChannel</td>
<td>Superseded with Get SourcePowerCalDataEx</td>
</tr>
<tr>
<td>getSourcePowerCalDataEx</td>
<td>IChannel4</td>
<td>Returns requested source power calibration data, if it exists.</td>
</tr>
<tr>
<td>GetSupportedALCModes</td>
<td>IChannel10</td>
<td>Returns a list of supported ALC modes</td>
</tr>
<tr>
<td>GetXAxisValues</td>
<td>IChannel</td>
<td>Returns the channel's X-axis values into a dimensioned Variant array.</td>
</tr>
<tr>
<td>GetXAxisValues2</td>
<td>IChannel</td>
<td>Returns the channel's X-axis values into a dimensioned NON-Variant array.</td>
</tr>
<tr>
<td>Hold</td>
<td>IChannel</td>
<td>Puts the Channel in Hold - not sweeping.</td>
</tr>
<tr>
<td>Next_IFBandwidth</td>
<td>IChannel</td>
<td>A function that returns the Next higher IF Bandwidth value.</td>
</tr>
<tr>
<td>NumberOfGroups</td>
<td>IChannel</td>
<td>Sets the Number of trigger signals the channel will receive.</td>
</tr>
<tr>
<td>Reset</td>
<td>IChannel</td>
<td>Resets the channel to factory defined settings.</td>
</tr>
<tr>
<td>PreviousIFBandwidth</td>
<td>IChannel</td>
<td>Returns the previous IF Bandwidth value.</td>
</tr>
<tr>
<td>putSourcePowerCalData</td>
<td>IChannel</td>
<td>Superseded with Put SourcePowerCalDataEx Method</td>
</tr>
<tr>
<td>putSourcePowerCalDataEx</td>
<td>IChannel4</td>
<td>Inputs source power calibration data to this channel for a specific source port.</td>
</tr>
<tr>
<td>SelectCalSet</td>
<td>IChannel</td>
<td>Specifies the Cal Set to use for the Channel</td>
</tr>
<tr>
<td>Single</td>
<td>IChannel</td>
<td>Channel responds to one trigger signal from any source (internal, external, or manual). Then channel switches to Hold.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALCLevelingMode</td>
<td>IChannel10</td>
<td>Set or return the ALC leveling mode.</td>
</tr>
<tr>
<td>AlternateSweep</td>
<td>IChannel</td>
<td>Sets sweeps to either alternate or chopped.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Attenuator</td>
<td>IChannel</td>
<td>Sets or returns the value of the attenuator control for the specified port number.</td>
</tr>
<tr>
<td>AttenuatorMode</td>
<td>IChannel</td>
<td>Sets or returns the mode of operation of the attenuator control for the specified port number.</td>
</tr>
<tr>
<td>AuxiliaryTrigger</td>
<td>IChannel10</td>
<td>Used to configure AuxiliaryTriggering</td>
</tr>
<tr>
<td>Property</td>
<td>Class</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FrequencyOffsetDivisor</td>
<td>IChannel2</td>
<td></td>
</tr>
<tr>
<td>FrequencyOffsetFrequency</td>
<td>IChannel2</td>
<td></td>
</tr>
<tr>
<td>FrequencyOffsetMultiplier</td>
<td>IChannel2</td>
<td>Superseded with FOM and FOMRange</td>
</tr>
<tr>
<td>FrequencyOffsetCWOVERRIDE</td>
<td>IChannel2</td>
<td></td>
</tr>
<tr>
<td>FrequencyOffsetState</td>
<td>IChannel2</td>
<td></td>
</tr>
<tr>
<td>FrequencySpan</td>
<td>IChannel</td>
<td>Sets or returns the frequency span of the channel.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Segment Object.</td>
</tr>
<tr>
<td>IFBandwidth</td>
<td>IChannel</td>
<td>Sets or returns the IF Bandwidth of the channel.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Segment Object.</td>
</tr>
<tr>
<td>IFConfiguration</td>
<td>IChannel4</td>
<td>Control the IF gain and source path settings for the H11 Option.</td>
</tr>
<tr>
<td>IsContinuous</td>
<td>IChannel3</td>
<td>Returns whether or not a channel is in continuous mode.</td>
</tr>
<tr>
<td>IsHold</td>
<td>IChannel3</td>
<td>Returns whether or not a channel is in hold mode.</td>
</tr>
<tr>
<td>NumberOfPoints</td>
<td>IChannel</td>
<td>Sets or returns the Number of Points of the channel.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Segment Object.</td>
</tr>
<tr>
<td>Parent</td>
<td>IChannel</td>
<td>Returns a handle to the parent object of the channel.</td>
</tr>
<tr>
<td>PowerSlope</td>
<td>IChannel</td>
<td>Sets or returns the Power Slope value.</td>
</tr>
<tr>
<td>R1InputPath</td>
<td>IChannel2</td>
<td>Throws internal reference switch (option 081).</td>
</tr>
<tr>
<td>ReceiverAttenuator</td>
<td>IChannel</td>
<td>Sets or returns the value of the specified receiver attenuator control.</td>
</tr>
<tr>
<td>ReduceIFBandwidth</td>
<td>IChannel5</td>
<td>Sets or returns the state of the Reduced IF Bandwidth at Low Frequencies setting.</td>
</tr>
<tr>
<td>Segments</td>
<td>IChannel</td>
<td>Collection for iterating through the sweep segments of a channel.</td>
</tr>
<tr>
<td>SourcePortMode</td>
<td>IChannel9</td>
<td>Sets the state of the PNA sources. (AUTO</td>
</tr>
<tr>
<td>SourcePowerCalPowerOffset</td>
<td>IChannel4</td>
<td>Sets or returns a power level offset from the PNA test port power.</td>
</tr>
<tr>
<td>SourcePowerCorrection</td>
<td>IChannel</td>
<td>Turns source power correction ON or OFF for a specific source port.</td>
</tr>
</tbody>
</table>
**StartFrequency**

IChannel

Sets or returns the start frequency of the channel.

Shared with the Segment Object

**StartPower**

IChannel

Sets the start power of the analyzer when sweep type is set to Power Sweep.

**StopFrequency**

IChannel

Sets or returns the stop frequency of the channel.

Shared with the Segment Object

**StopPower**

IChannel

Sets the Stop Power of the analyzer when sweep type is set to Power Sweep.

**SweepGenerationMode**

IChannel

Sets the method used to generate a sweep: continuous ramp (analog) or discrete steps (stepped).

**SweepTime**

IChannel

Sets the Sweep time of the analyzer.

**SweepType**

IChannel

Sets the type of X-axis sweep that is performed on a channel.

**TestPortPower**

IChannel

Sets or returns the RF power level for the channel.

Shared with the Segment Object

**TriggerMode**

IChannel

Determines the measurement that occurs when a trigger signal is sent to the channel.

**UserRangeMax**

IChannel

Sets the stimulus stop value for the specified User Range.

**UserRangeMin**

IChannel

Sets the stimulus start value for the specified User Range.

**XAxisPointSpacing**

IChannel

Sets X-Axis point spacing for the active channel.

### IChannel History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IChannel</td>
<td>1.0</td>
</tr>
<tr>
<td>IChannel2</td>
<td>3.0</td>
</tr>
<tr>
<td>IChannel3</td>
<td>4.0</td>
</tr>
<tr>
<td>IChannel4</td>
<td>4.0</td>
</tr>
<tr>
<td>IChannel5</td>
<td>4.2</td>
</tr>
<tr>
<td>IChannel6</td>
<td>5.0</td>
</tr>
<tr>
<td>IChannel7</td>
<td>5.2</td>
</tr>
</tbody>
</table>
**ISourcePowerCalData Interface**

**Description**
Contains methods for putting source power calibration data in and getting source power calibration data out of the analyzer using typed data. The methods in this interface transfer data more efficiently than methods that use variant data. However, this interface is only usable from VB6, C, & C++. All other programming languages must use the methods on the Channel Object.

*Note:* Interface ISourcePowerCalData is abbreviated as ISPCD in the following table.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getSourcePowerCalDataScalar</code></td>
<td>ISPCD</td>
<td>Superseded with - use <a href="https://example.com">PutSourcePowerCalDataScalarEx</a> Method</td>
</tr>
<tr>
<td><code>getSourcePowerCalDataScalarEx</code></td>
<td>ISPCD2</td>
<td>Returns requested source power calibration data, if it exists.</td>
</tr>
<tr>
<td><code>putSourcePowerCalDataScalar</code></td>
<td>ISPCD</td>
<td>Superseded with - use <a href="https://example.com">PutSourcePowerCalDataScalarEx</a> Method</td>
</tr>
<tr>
<td><code>putSourcePowerCalDataScalarEx</code></td>
<td>ISPCD2</td>
<td>Inputs source power calibration data to a channel, for a specific source port.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**ISourcePowerCalData History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISourcePowerCalData</td>
<td>2.0</td>
</tr>
<tr>
<td>ISourcePowerCalData2</td>
<td>4.0</td>
</tr>
</tbody>
</table>
Channels Collection

Description
A collection object that provides a mechanism for iterating through the channels.

Collections are, by definition, unordered lists of like objects. You cannot assume that Channels.Item(1) is always Channel 1.

Accessing the Channels collection
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chans As Channels
Set chans = app.Channels

See Also:
- Channel Object
- Collections in the Analyzer
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>IChannels</td>
<td>An alternate way to create a measurement.</td>
</tr>
<tr>
<td>Hold</td>
<td>IChannels</td>
<td>Places all channels in Hold trigger mode.</td>
</tr>
<tr>
<td>Item</td>
<td>IChannels2</td>
<td>Use to get a handle on a channel in the collection.</td>
</tr>
<tr>
<td>Resume</td>
<td>IChannels2</td>
<td>Resumes the trigger mode of all channels that was in effect before sending the channels.Hold method.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>IChannels</td>
<td>Returns the number of channels in the analyzer.</td>
</tr>
<tr>
<td>Parent</td>
<td>IChannels</td>
<td>Returns a handle to the current Application.</td>
</tr>
<tr>
<td>UnusedChannelNumbers</td>
<td>IChannels2</td>
<td>Returns an array of channel numbers that are NOT in use.</td>
</tr>
<tr>
<td>UsedChannelNumbers</td>
<td>IChannels2</td>
<td>Returns an array of channel numbers that are in use.</td>
</tr>
</tbody>
</table>
E5091Testsets Collection

Description

Two testsets can be connected and controlled by the PNA at any time.

The item number in the testsets collection is set by the DIP switches on the testset rear-panel. The valid item numbers are 1 and 2. If the testset DIP switches are set to 1, then item number in the collection is 1, and so forth. See your E5091A documentation for more information.

If the specified testset is not connected to USB or not ON, then setting Enabled = True will return an error. All other properties can be set when the testset is not connected.

Accessing the E5091Testsets collection

Child of the Application Object. Get a handle to one of the E5091Testset objects by specifying an item of the collection.

```vba
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Dim testsets As E5091Testsets
Set testsets = pna.E5091Testsets
Dim tset1 As E5091Testset
Set tset1 = testsets(1)
```

See Also:

- [E5091Testset Control COM Example](#)
- [E5091Testset Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Use to get a handle to a testset in the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of items in a collection of objects.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current naNetworkAnalyzer application.</td>
</tr>
</tbody>
</table>

E5091Testsets History

742
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE5091Testsets</td>
<td>5.2</td>
</tr>
</tbody>
</table>
E5091Testset Object

Description
There can be two test sets connected and controlled by the PNA at any time.

The item number in the testsets collection is set by the DIP switches on the test set rear-panel. The valid item numbers are 1 and 2. If the test set DIP switches are set to 1, then item number in the collection is 1, and so forth. See your E5091A documentation for more information.

If the specified test set is not connected to USB or not ON, then setting `Enabled` = True will return an error. All other properties can be set when the test set is not connected.

Accessing the E5091Testset object
Child of the Application Object. Get a handle to a E5091Testset object by specifying an item of the collection.

```vba
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Dim testsets As E5091Testsets
Set testsets = pna.E5091Testsets
Dim tset1 As E5091Testset
Set tset1 = testsets(1)
```

See Also:
- E5091Testset Control COM Example
- E5091 TestSet Control
- E5091Testsets Collection
- TestsetControl Object (for different test sets)
- The PNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ControlLines</td>
<td>Sets the control lines of the specified E5091A.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Enables and disables (ON/OFF) the port mapping and control line output of the specified testset.</td>
</tr>
<tr>
<td>ID</td>
<td>Returns the test set ID number.</td>
</tr>
<tr>
<td>NumberOfPorts</td>
<td>Reads the number of ports (7 or 9) that are on the specified E5091A test set.</td>
</tr>
</tbody>
</table>
**OutputPort**

Switches an input to one of the valid outputs on the specified E5091A.

**ShowProperties**

Turns ON and OFF the display of the test set control status bar.

---

**E5091Testset History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE5091Testset</td>
<td>5.2</td>
</tr>
</tbody>
</table>
EmbeddedLO Object

Description
Provides access to the properties that allow measurement of mixers that contain an embedded LO.

Accessing the EmbeddedLO Interface
Access the Interface through the IMixer Object.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset

' FCA Measurements can't share the channel with standard measurements
' Because preset creates a single measurement in channel 1, we first delete the
standard measurement
Dim standardMeas As IMeasurement
Set standardMeas = app.ActiveMeasurement
standardMeas.Delete

' Create a Measurement object, in this case using the IMeasurement interface
Dim meas As IMeasurement
Set meas = app.CreateCustomMeasurementEx(1, "SMC_Forward.SMC_ForwardMeas", "SC21")

' See if this measurement object supports IMixer
Dim mixer As IMixer
Dim embeddedLO
Set embeddedLO = mixer.EmbeddedLO
```

See an example program that shows how to create and calibrate a standard SMC or VMC measurement or a fixed output SMC measurement.

See Also:
PNA Automation Interfaces
The PNA Object Model
Making Embedded LO Measurements
<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResetLOFrequency</td>
<td>IEmbeddedLO</td>
<td>Reset LO Delta frequency.</td>
</tr>
<tr>
<td>ResetTuningParameters</td>
<td>IEmbeddedLO</td>
<td>Resets the tuning parameters to their defaults.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BroadbandTuningSpan</td>
<td>IEmbeddedLO Set broadband sweep span.</td>
</tr>
<tr>
<td>EmbeddedLODiagnostic</td>
<td>IEmbeddedLO Provides access to the status of tuning sweeps.</td>
</tr>
<tr>
<td>IsOn</td>
<td>IEmbeddedLO Set and return Embedded LO ON</td>
</tr>
<tr>
<td>LOFrequencyDelta</td>
<td>IEmbeddedLO Sets and returns LO delta frequency.</td>
</tr>
<tr>
<td>MaxPreciseTuningIterations</td>
<td>IEmbeddedLO Sets and returns precise tuning iterations.</td>
</tr>
<tr>
<td>NormalizePoint</td>
<td>IEmbeddedLO Sets and returns tuning point.</td>
</tr>
<tr>
<td>PreciseTuningTolerance</td>
<td>IEmbeddedLO Sets and returns precise tuning tolerance.</td>
</tr>
<tr>
<td>TuningIFBW</td>
<td>IEmbeddedLO Sets and returns the IF Bandwidth for tuning sweeps.</td>
</tr>
<tr>
<td>TuningMode</td>
<td>IEmbeddedLO Sets and returns the method used to determine the embedded LO Frequency.</td>
</tr>
<tr>
<td>TuningSweepInterval</td>
<td>IEmbeddedLO Set how often a tuning sweep is performed.</td>
</tr>
</tbody>
</table>

**IEmbeddedLO History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEmbeddedLO</td>
<td>7.21</td>
</tr>
</tbody>
</table>
EmbeddedLODiagnostic Object

Description
Allows access to the properties that provide information about the broadband and precise tuning of an embedded LO.

Accessing the EmbeddedLODiagnostic Interface
Access the Interface through the EmbeddedLO Object.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset

' FCA Measurements can't share the channel with standard measurements
' Because preset creates a single measurement in channel 1, we first delete the standard measurement
Dim standardMeas As IMeasurement
Set standardMeas = app.ActiveMeasurement
standardMeas.Delete

' Create a Measurement object, in this case using the IMeasurement interface
Dim meas As IMeasurement
Set meas = app.CreateCustomMeasurementEx(1, "SMC_Forward.SMC_ForwardMeas", "SC21")

' See if this measurement object supports IMixer
Dim mixer As IMixer
Dim embeddedLO
Set embeddedLO = mixer.EmbeddedLO
Dim embeddedLODiagnostic
Set embeddedLODiagnostic = embeddedLO.EmbeddedLODiagnostic
```

See an example program that shows how to create and calibrate a standard SMC or VMC measurement or a fixed output SMC measurement.

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Making Embedded LO Measurements
- EmbeddedLO Object
<table>
<thead>
<tr>
<th><strong>Methods</strong></th>
<th><strong>Interface</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear</td>
<td>IELODiag</td>
<td>Clear current diagnostic information.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Properties</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>IsMarkerOn</td>
<td>IELODiag</td>
</tr>
<tr>
<td>LODeltaFound</td>
<td>IELODiag</td>
</tr>
<tr>
<td>NumberOfSweeps</td>
<td>IELODiag</td>
</tr>
<tr>
<td>MarkerAnnotation</td>
<td>IELODiag</td>
</tr>
<tr>
<td>MarkerPosition</td>
<td>IELODiag</td>
</tr>
<tr>
<td>Parameter</td>
<td>IELODiag</td>
</tr>
<tr>
<td>StatusAsString</td>
<td>IELODiag</td>
</tr>
<tr>
<td>StepData</td>
<td>IELODiag</td>
</tr>
<tr>
<td>StepTitle</td>
<td>IELODiag</td>
</tr>
<tr>
<td>XAxisAnnotation</td>
<td>IELODiag</td>
</tr>
<tr>
<td>XAxisStart</td>
<td>IELODiag</td>
</tr>
<tr>
<td>XAxisStop</td>
<td>IELODiag</td>
</tr>
<tr>
<td>YAxisAnnotation</td>
<td>IELODiag</td>
</tr>
</tbody>
</table>

**History**

<table>
<thead>
<tr>
<th><strong>Interface</strong></th>
<th><strong>Introduced with PNA Rev:</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>IEmbeddedLODiagnostic</td>
<td>7.21</td>
</tr>
</tbody>
</table>
ENRFile Object

Description
Provide commands for creating or editing an ENR file. This is rarely necessary as ENR files, which contain factory calibrated data, are typically provided by the manufacturer of the noise source.

Learn more about Noise Figure Application

Accessing the ENRFile object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim enr As ENRFile
Set enr = app.ENRFile

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Program

Methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetENRData</td>
<td>IENRFile</td>
<td>Read the ENR calibration data from PNA memory.</td>
</tr>
<tr>
<td>PutENRData</td>
<td>IENRFile</td>
<td>Write the ENR calibration data to PNA memory.</td>
</tr>
<tr>
<td>LoadENRFile</td>
<td>IENRFile</td>
<td>Recalls an ENR file from disk into PNA Memory.</td>
</tr>
<tr>
<td>SaveENRFile</td>
<td>IENRFile</td>
<td>Saves an ENR file from PNA memory to disk.</td>
</tr>
</tbody>
</table>

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENRID</td>
<td>IENRFile</td>
<td>Sets and returns ID of ENR table.</td>
</tr>
<tr>
<td>ENRSN</td>
<td>IENRFile</td>
<td>Sets and returns the serial number of the noise source.</td>
</tr>
</tbody>
</table>

IENRFile History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IENRFile</td>
<td>8.0</td>
</tr>
</tbody>
</table>

Last Modified:

2-Aug-2007   MX New topic
Equation Object

Description
Provide commands for creating an equation.
Learn more about Equation Editor

Accessing the Equation object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim eq As Equation
Set eq = app.Equation
```

See Also:

- PNA Automation Interfaces
- The PNA Object Model

### Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>See History</strong></td>
<td></td>
</tr>
</tbody>
</table>

None

#### Properties

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>See History</strong></td>
<td></td>
</tr>
</tbody>
</table>

| Text | IEquation | Sets the Equation |
| State | IEquation | Sets the Equation enabled state |
| Valid | IEquation | Returns whether the equation is presently valid. |

Example Program using these commands:
Dim na
Dim meas
Set na = CreateObject("AgilentPNA835x.Application")
Set meas = na.ActiveMeasurement
' Define the measurement
meas.Equation.Text = "mysillyequ=sqrt(AR1_1)"
' Check to see if the equation is valid
valid_e = meas.Equation.Valid
MsgBox valid_e
' Turn on the Equation Editor
meas.Equation.State = True

IEquation History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEquation</td>
<td>6.03</td>
</tr>
</tbody>
</table>

Last Modified:

4-Dec-2007 Added example
ExternalTestsets Collection

Description
ExternalTestsets collection provides access to a TestsetControl object. Only one external testset can be controlled by the PNA at any time.

Accessing the ExternalTestsets collection
The ExternalTestsets collection is a property of the main Application Object. You can obtain a handle to a testset by specifying an item in the collection.

Visual Basic Example

```vbnet
Dim pna
Dim testsets As ExternalTestsets
Dim tset1 As TestsetControl
Set pna = CreateObject("AgilentPNA835x.Application")
Set testsets = pna.ExternalTestsets
Set tset1 = testsets(1)
' make COM calls on tset1 object
End Sub
```

See Also:
- ExternalTestset Control COM Example
- About External TestSet Control
- TestsetControl Object
- The PNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a testset to the collection and loads a test set configuration file.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a testset in the collection.</td>
</tr>
<tr>
<td>TestsetCatalog</td>
<td>Returns a list of supported test sets.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of items in a collection of objects.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current naNetworkAnalyzer application.</td>
</tr>
</tbody>
</table>

ExternalTestsets History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IExternalTestsets</td>
<td>6.0</td>
</tr>
<tr>
<td>IExternalTestsets</td>
<td>6.2</td>
</tr>
</tbody>
</table>
Fixturing Object

Description
Contains the properties for Embedding and De-embedding test fixtures.

Accessing the Fixturing object

```vba
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim fixt as Fixturing
Set fixt = chan.Fixturing
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- About Fixturing
- Example Programs

## Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoPortExtMeasure</td>
<td>IFixturing2</td>
<td>Measures either an OPEN or SHORT standard.</td>
</tr>
<tr>
<td>AutoPortExtReset</td>
<td>IFixturing2</td>
<td>Clears old port extension delay and loss data.</td>
</tr>
</tbody>
</table>

## Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoPortExtConfig</td>
<td>IFixturing2</td>
<td>Sets the frequency span that is used to calculate Automatic Port Extension.</td>
</tr>
<tr>
<td>AutoPortExtDCOffset</td>
<td>IFixturing2</td>
<td>Specifies whether or not to include DC Offset as part of Automatic port extension.</td>
</tr>
<tr>
<td>AutoPortExtLoss</td>
<td>IFixturing2</td>
<td>Specifies whether or not to include loss correction as part of Automatic Port Extension.</td>
</tr>
<tr>
<td>AutoPortExtSearchStart</td>
<td>IFixturing2</td>
<td>Set the start frequency for custom user span.</td>
</tr>
<tr>
<td>AutoPortExtSearchStop</td>
<td>IFixturing2</td>
<td>Set the stop frequency for custom user span.</td>
</tr>
</tbody>
</table>
AutoPortExtState IFixturing2 Enables and disables automatic port extensions on the specified port.

CmnModeZConvPortImag IFixturing2 Sets imaginary value for common port impedance conversion.

CmnModeZConvPortReal IFixturing2 Sets real value for common port impedance conversion.

CmnModeZConvPortState IFixturing2 Turns ON/OFF common port impedance conversion.

CmnModeZConvPortZ0 IFixturing2 Sets impedance value for common port impedance conversion.

DiffPortMatch_C IFixturing2 Sets Capacitance value of the differential matching circuit.

DiffPortMatch_G IFixturing2 Sets Conductance value of the differential matching circuit.

DiffPortMatch_L IFixturing2 Sets Inductance value of the differential matching circuit.

DiffPortMatch_R IFixturing2 Sets Resistance value of the differential matching circuit.

DiffPortMatchMode IFixturing2 Sets type of circuit to embed.

DiffPortMatchUserFilename IFixturing2 Specifies the 4-port touchstone file for user-defined differential matching circuit.

DiffPortMatchState IFixturing2 Turns ON/OFF differential matching circuit function.

DiffZConvPortImag IFixturing2 Sets imaginary value for differential port impedance conversion.

DiffZConvPortReal IFixturing2 Sets real value for differential port impedance conversion.

DiffZConvPortZ0 IFixturing2 Sets impedance value for differential port impedance conversion.

DiffZConvState IFixturing2 Turns ON/OFF differential port impedance conversion.

Embed4PortA IFixturing2 Returns PNA portA connections.

Embed4PortB IFixturing2 Returns PNA portB connections.

Embed4PortC IFixturing2 Returns PNA portC connections.
<table>
<thead>
<tr>
<th>Function</th>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embed4PortD</td>
<td>IFixturing2</td>
<td>Returns PNA portD connections.</td>
</tr>
<tr>
<td>Embed4PortList</td>
<td>IFixturing2</td>
<td>Specifies all PNA port connections.</td>
</tr>
<tr>
<td>Embed4PortNetworkFilename</td>
<td>IFixturing2</td>
<td>Specifies *.s4p filename.</td>
</tr>
<tr>
<td>Embed4PortNetworkMode</td>
<td>IFixturing2</td>
<td>Specify embed, de-embed, or none.</td>
</tr>
<tr>
<td>Embed4PortState</td>
<td>IFixturing2</td>
<td>Turns ON or OFF 4-port Network Embed/De-embed.</td>
</tr>
<tr>
<td>Embed4PortTopology</td>
<td>IFixturing2</td>
<td>Specifies the PNA / DUT topology.</td>
</tr>
<tr>
<td>FixturingState</td>
<td>IFixturing</td>
<td>Turns Fixturing ON and OFF on this channel.</td>
</tr>
<tr>
<td>Port2PdeembedCktModel</td>
<td>IFixturing</td>
<td>Sets and returns the 2 port De-embedding circuit model for the specified port number.</td>
</tr>
<tr>
<td>Port2PdeembedState</td>
<td>IFixturing</td>
<td>Turns 2 port de-embedding ON and OFF on this channel.</td>
</tr>
<tr>
<td>PortArbzImag</td>
<td>IFixturing3</td>
<td>Sets and returns the imaginary impedance value for the specified single-ended port number.</td>
</tr>
<tr>
<td>PortArbzReal</td>
<td>IFixturing3</td>
<td>Sets and returns the real impedance value for the specified single-ended port number.</td>
</tr>
<tr>
<td>PortArbzState</td>
<td>IFixturing</td>
<td>Turns single-ended port impedance ON and OFF on the specified channel.</td>
</tr>
<tr>
<td>PortArbzZ0</td>
<td>IFixturing3</td>
<td>Sets and returns the real and imaginary impedance value for the specified single-ended port number.</td>
</tr>
<tr>
<td>PortDelay</td>
<td>IFixturing</td>
<td>Sets and returns the Port Delay value for the specified port number.</td>
</tr>
<tr>
<td>PortExtState</td>
<td>IFixturing</td>
<td>Turns Port Extension ON and OFF on this channel.</td>
</tr>
<tr>
<td>PortExtUse1</td>
<td>IFixturing</td>
<td>Sets and returns the USE1 ON/OFF state for the Loss1 and Freq1 values for the specified port number.</td>
</tr>
<tr>
<td>PortExtUse2</td>
<td>IFixturing</td>
<td>Sets and returns the USE2 ON/OFF state for the Loss2 and Freq2 values for the specified port number.</td>
</tr>
<tr>
<td>PortFreq1</td>
<td>IFixturing</td>
<td>Sets and returns the 1st Port Frequency value for the specified port number.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>PortFreq2</td>
<td>Sets and returns the 2nd Port Frequency value for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>PortLoss1</td>
<td>Sets and returns the 1st Port Loss value for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>PortLoss2</td>
<td>Sets and returns the 2nd Port Loss value for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>PortLossDC</td>
<td>Sets and returns the Port Loss at DC value for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>PortMatching_C</td>
<td>Sets and returns the Capacitance, 'C' value for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>PortMatching_G</td>
<td>Sets and returns the Conductance, 'G' value for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>PortMatching_L</td>
<td>Sets and returns the Inductance, 'L' value for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>PortMatching_R</td>
<td>Sets and returns the Resistance, 'R' value for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>PortMatchingCktModel</td>
<td>Sets and returns the Port Matching circuit model for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>PortMatchingState</td>
<td>Turns Port Matching ON and OFF on this channel.</td>
<td></td>
</tr>
<tr>
<td>strPort2Pdeembed_S2PFile</td>
<td>Sets and returns the 2 port De-embedding 'S2P' file name for the specified port number.</td>
<td></td>
</tr>
<tr>
<td>strPortMatch_S2PFile</td>
<td>Sets and returns the Port Matching 'S2P' file name for the specified port number.</td>
<td></td>
</tr>
</tbody>
</table>

**IFixturing History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFixturing</td>
<td>5.0</td>
</tr>
<tr>
<td>IFixturing2</td>
<td>5.2</td>
</tr>
<tr>
<td>IFixturing3</td>
<td>5.25</td>
</tr>
</tbody>
</table>
FOM Collection

Description
The FOM collection provides access to the source and receiver range objects which are used for configuring frequency offset measurements.

The FOM range items are typically numbered as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)

Accessing the FOM Collection and FOMRange objects

```vba
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel

Dim ifom as FOM
Set ifom = chan.FOM

ifom.item(2).Coupled = false
```

See Also:

- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [About FOM](#)
- [Example Programs](#)
<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>IFOM</td>
<td>See History</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DisplayRange</td>
<td>IFOM</td>
<td>Sets the range to be displayed on the PNA x-axis.</td>
</tr>
<tr>
<td>FOMRange</td>
<td>IFOM</td>
<td>Object</td>
</tr>
<tr>
<td>RangeCount</td>
<td>IFOM</td>
<td>Returns the number of FOM ranges available on the PNA.</td>
</tr>
<tr>
<td>State</td>
<td>IFOM</td>
<td>Turns Frequency Offset ON and OFF.</td>
</tr>
</tbody>
</table>

**FOM History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFOM</td>
<td>7.10</td>
</tr>
</tbody>
</table>

Last Modified:

8-Mar-2007  Modified Access
**FOMRange Object**

**Description**
The FOM Range object provides access to the properties and methods for configuring a specific Range for frequency offset measurements.

**Accessing an FOMRange object**
Get a handle to a FOM Range by specifying an item in the FOM collection.
The FOM range items are typically numbered as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)

```vbnet
Dim app as AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim ranges as FOM
Set ranges = app.ActiveChannel.FOM
ranges.item(2).Coupled = False
```

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- About FOM
- Example Programs
<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>See History</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupled</td>
<td>IFOMRange</td>
<td>Sets and returns the state of coupling (ON or OFF) of this range to the primary range.</td>
</tr>
<tr>
<td>CWFrequency</td>
<td>IFOMRange</td>
<td>Set the Continuous Wave (CW) frequency.</td>
</tr>
<tr>
<td>Divisor</td>
<td>IFOMRange</td>
<td>Sets and returns the Divisor value to be used when coupling this range to the primary range.</td>
</tr>
<tr>
<td>Multiplier</td>
<td>IFOMRange</td>
<td>Sets and returns the Multiplier value to be used when coupling this range to the primary range.</td>
</tr>
<tr>
<td>Name</td>
<td>IFOMRange</td>
<td>Returns the name of this FOM range object.</td>
</tr>
<tr>
<td>Offset</td>
<td>IFOMRange</td>
<td>Sets and returns the offset value to be used when coupling this range to the primary range.</td>
</tr>
<tr>
<td>rangeNumber</td>
<td>IFOMRange</td>
<td>Returns the index number of the range within the FOM collection.</td>
</tr>
<tr>
<td>Segments</td>
<td>IFOMRange</td>
<td>Collection - Used to add segment sweep capability to a range.</td>
</tr>
<tr>
<td>StartFrequency</td>
<td>IFOMRange</td>
<td>Sets or returns the start frequency of this FOM Range.</td>
</tr>
<tr>
<td>StopFrequency</td>
<td>IFOMRange</td>
<td>Sets or returns the stop frequency of this FOM Range.</td>
</tr>
<tr>
<td>Sweep Type</td>
<td>IFOMRange</td>
<td>Sets the type of range sweep.</td>
</tr>
</tbody>
</table>

**Note**: Use the [Start Power](#) and [Stop Power](#) settings from the **channel object**.

### FOM History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFOM</td>
<td>7.10</td>
</tr>
</tbody>
</table>

Last Modified:
7-Jan-2008    Added Start/Stop power note

7-Mar-2007    Modified Receivers
Gain Compression Object

Description
Controls the Gain Compression Application settings.

Accessing the GainCompression object

```vba
Dim app as AgilentPNA835x.Application
app.CreateCustomMeasurementEx(1, "Gain Compression", "CompIn21", 1)
Dim GCA
Set GCA = app.ActiveChannel.CustomChannelConfiguration
```

See Also:

- Example Program Create and Cal a Gain Compression Measurement
- GainCompressionCal Object
- About Gain Compression Application
- PNA Automation Interfaces
- The PNA Object Model

**Note:** Set the Start/Stop Frequency and Start/Stop Power Settings using the Channel Object.

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetRaw2DData</td>
<td>IGainCompression</td>
<td>Reads Gain Compression data from specified location.</td>
</tr>
<tr>
<td>GetDataIm</td>
<td>IGainCompression</td>
<td>Reads Imaginary part of specified frequency or power points.</td>
</tr>
<tr>
<td>GetDataRe</td>
<td>IGainCompression</td>
<td>Reads REAL part of specified frequency or power points.</td>
</tr>
<tr>
<td>SetPortMap</td>
<td>IGainCompression</td>
<td>Maps the PNA ports to the DUT ports</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AcquisitionMode</strong></td>
<td>IGainCompression</td>
<td>Set and read the method by which gain compression data is acquired.</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>CompressionAlgorithm</strong></td>
<td>IGainCompression</td>
<td>Set and read the algorithm method used to compute gain compression.</td>
</tr>
<tr>
<td><strong>CompressionBackoff</strong></td>
<td>IGainCompression</td>
<td>Set and read value for the BackOff compression algorithm.</td>
</tr>
<tr>
<td><strong>CompressionDeltaX</strong></td>
<td>IGainCompression</td>
<td>Set and read the 'X' value in the delta X/Y compression algorithm.</td>
</tr>
<tr>
<td><strong>CompressionDeltaY</strong></td>
<td>IGainCompression</td>
<td>Set and read the 'Y' value in the delta X/Y compression algorithm.</td>
</tr>
<tr>
<td><strong>CompressionInterpolation</strong></td>
<td>IGainCompression</td>
<td>Sets whether or not to interpolate the final power level when the measured compression level deviates from the specified level.</td>
</tr>
<tr>
<td><strong>CompressionLevel</strong></td>
<td>IGainCompression</td>
<td>Set and read the decrease in gain which indicates that the amplifier is compressing.</td>
</tr>
<tr>
<td><strong>DeviceInputPort</strong></td>
<td>IGainCompression</td>
<td>Set and read the PNA port number which is connected to the DUT input.</td>
</tr>
<tr>
<td><strong>DeviceOutputPort</strong></td>
<td>IGainCompression</td>
<td>Set and read the PNA port number which is connected to the DUT Output.</td>
</tr>
<tr>
<td><strong>EndOfSweepOperation</strong></td>
<td>IGainCompression</td>
<td>Set and read the action which should be taken at the end of the last frequency or power sweep in the measurement.</td>
</tr>
<tr>
<td><strong>InputLinearPowerLevel</strong></td>
<td>IGainCompression</td>
<td>Set and read the input power level that should produce linear gain.</td>
</tr>
<tr>
<td><strong>MaximumNumberOfPoints</strong></td>
<td>IGainCompression</td>
<td>Returns the maximum possible number of data points.</td>
</tr>
<tr>
<td><strong>NumberOfFrequencyPoints</strong></td>
<td>IGainCompression</td>
<td>Set and read the number of data points in each frequency sweep.</td>
</tr>
<tr>
<td><strong>NumberOfPowerPoints</strong></td>
<td>IGainCompression</td>
<td>Set and read the number of data points in each power sweep.</td>
</tr>
<tr>
<td><strong>ReverseLinearPowerLevel</strong></td>
<td>IGainCompression</td>
<td>Set and read the reverse power level to the DUT.</td>
</tr>
</tbody>
</table>
SafeSweepCoarsePowerAdjustment IGainCompression Set and read the Safe Sweep COURSE power adjustment.

SafeSweepEnable IGainCompression Set and read the (ON | OFF) state of Safe Sweep mode.

SafeSweepFinePowerAdjustment IGainCompression Set and read the Safe Sweep FINE power adjustment.

SafeSweepFineThreshold IGainCompression Set and read the compression level in which Safe Sweep changes from the COARSE power adjustment to the FINE power adjustment.

SearchFailures IGainCompression Read number of points that did not achieve compression.

SmartSweepMaximumIterations IGainCompression Set and read the maximum number of iterations to be used to find the compression level in a SMART sweep.

SmartSweepSettlingTime IGainCompression Set and read SMART sweep settling time.

SmartSweepShowIterations IGainCompression Set and read whether to show results for each SMART sweep iteration.

SmartSweepTolerance IGainCompression Set and read the level of tolerance to be used to find the compression level in a SMART sweep.

TotalNumberOfPoints IGainCompression Set and read the total number of data points.(Freq x Power)

IGainCompression History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGainCompression</td>
<td>8.0</td>
</tr>
</tbody>
</table>

Last Modified:

11-Sep-2007   MX New topic
Gain Compression Cal Object

Description
Sets properties that are unique to a Gain Compression Cal (opt 086).
The remaining commands to perform a GCA Cal use the Guided Calibration commands.

Accessing the GainCompressionCal object

```vba
Dim app as AgilentPNA835x.Application
Set GCAcal = pna.GetCalmanager.CreateCustomCalEx(channelNum)
Set GCACalExtension = GCAcal.CustomCalConfiguration
GCACalExtension.PowerLevel = 5
```

See Also:
- Example Program [Create and Cal a Gain Compression Measurement](#)
- [GainCompression Object](#)
- [About Gain Compression Application](#)
- [The PNA Object Model](#)
- [PNA Automation Interfaces](#)

### Method

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>See History</td>
</tr>
</tbody>
</table>

None

### Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerLevel</td>
<td>IGainCompressionCal</td>
<td>Set and read the power level of the source power cal.</td>
</tr>
</tbody>
</table>

IGainCompressionCal History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGainCompressionCal</td>
<td>8.0</td>
</tr>
</tbody>
</table>
Gating Object

Description
Contains the methods and properties that control Time Domain Gating.

Accessing the Gating Object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim gate As Gating
Set gate = app.ActiveMeasurement.Gating
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Time Domain Topics
- Example Programs

Methods
None

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center</td>
<td>IGating</td>
<td>Sets or returns the Center time.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Transform Object</td>
</tr>
<tr>
<td>CoupledParameters</td>
<td>IGating2</td>
<td>Select Gating parameters to couple</td>
</tr>
<tr>
<td>Shape</td>
<td>IGating</td>
<td>Specifies the shape of the gate filter.</td>
</tr>
<tr>
<td>Span</td>
<td>IGating</td>
<td>Sets or returns the Span time.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Transform Object</td>
</tr>
<tr>
<td>Start</td>
<td>IGating</td>
<td>Sets or returns the Start time.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Transform Object</td>
</tr>
<tr>
<td>State</td>
<td>IGating</td>
<td>Turns an Object ON and OFF.</td>
</tr>
</tbody>
</table>

(see History)
Stop
IGating  Sets or returns the Stop time.
Shared with the Transform Object

Type
IGating  Specifies the type of gate filter used.

History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGating</td>
<td>1.0</td>
</tr>
<tr>
<td>IGating2</td>
<td>4.2</td>
</tr>
</tbody>
</table>
GuidedCalibration Object

Description
Contains the methods and properties used to perform a Guided Calibration.

A Guided Calibration must be performed on the Active Channel. To activate a channel, activate any measurement on that channel. Do this using `meas.Activate`, which requires you already have a handle to the measurement.

**Note:** ECal orientation is performed using the `OrientECALModule.Property` and `ECALPortMapEx.Property` on the Calibrator Object.

Accessing the GuidedCalibration object

```vba
Dim app as AgilentPNA835x.Application
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim guidedCal
Set guidedCal = CalMgr.GuidedCalibration
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AcquireStep</strong></td>
<td>IGuidedCalibration</td>
</tr>
<tr>
<td><strong>ApplyDeltaMatchFromCalSet Method</strong></td>
<td>IGuidedCalibration2</td>
</tr>
<tr>
<td><strong>GenerateErrorTerms</strong></td>
<td>IGuidedCalibration</td>
</tr>
<tr>
<td><strong>GenerateGlobalDeltaMatchSequence</strong></td>
<td>IGuidedCalibration2</td>
</tr>
<tr>
<td><strong>GenerateSteps</strong></td>
<td>IGuidedCalibration</td>
</tr>
<tr>
<td><strong>GetIsolationPaths</strong></td>
<td>IGuidedCalibration3</td>
</tr>
<tr>
<td><strong>GetStepDescription</strong></td>
<td>IGuidedCalibration</td>
</tr>
</tbody>
</table>
**Initialize**
IGuidedCalibration
Initial setup with channel context for the remote cal object.

**SetIsolationPaths**
IGuidedCalibration3
Sets the list of port pairings for which isolation standards will be measured during calibration.

**SetupMeasurementsForStep**
IGuidedCalibration4
Show the Cal Window, or custom Cal Window, before acquiring a Cal standard.

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalKitType</td>
<td>IGuidedCalibration</td>
<td>Sets the cal kit for the port.</td>
</tr>
<tr>
<td>CompatibleCalKits</td>
<td>IGuidedCalibration</td>
<td>Returns the list of cal kits for the port.</td>
</tr>
<tr>
<td>ConnectorType</td>
<td>IGuidedCalibration</td>
<td>Sets the connector type for the port.</td>
</tr>
<tr>
<td>IsolationAveragingIncrement</td>
<td>IGuidedCalibration3</td>
<td>Value by which to increment the channel's averaging factor during measurement of isolation standards.</td>
</tr>
<tr>
<td>PathCalMethod</td>
<td>IGuidedCalibration3</td>
<td>Specifies the calibration method for each port pair.</td>
</tr>
<tr>
<td>PathThruMethod</td>
<td>IGuidedCalibration3</td>
<td>Specifies the calibration THRU method for each port pair.</td>
</tr>
<tr>
<td>PortsNeedingDeltaMatch</td>
<td>IGuidedCalibration2</td>
<td>Returns port numbers that need delta match cal.</td>
</tr>
<tr>
<td>ThruCalMethod</td>
<td>IGuidedCalibration</td>
<td><strong>Superseded with</strong> PathCalMethod and PathThruMethod</td>
</tr>
<tr>
<td>ThruPortList</td>
<td>IGuidedCalibration</td>
<td>Sets the thru connection port pairs.</td>
</tr>
<tr>
<td>UseCalWindow</td>
<td>IGuidedCalibration</td>
<td>Turns Cal window ON or OFF</td>
</tr>
<tr>
<td>ValidConnectorTypes</td>
<td>IGuidedCalibration</td>
<td>Gets Valid Connector Types.</td>
</tr>
</tbody>
</table>

**IGuidedCalibration History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGuidedCalibration</td>
<td>5.0</td>
</tr>
<tr>
<td>IGuidedCalibration2</td>
<td>5.25</td>
</tr>
<tr>
<td>IGuidedCalibration3</td>
<td>7.11</td>
</tr>
<tr>
<td>IGuidedCalibration4</td>
<td>8.0</td>
</tr>
</tbody>
</table>

Last Modified:

9-Nov-2007  Added Setup command and Activate note
HWAuxIO Object

Description
Contains the methods and properties that control the rear panel Auxiliary Input / Output connector.

Note: PNA-X models do NOT have this connector. However, the get/put Input/Output voltage commands can be used on the PNA-X to control ADC voltages on the Power I/O connector. Sending other Control:AUX commands to a PNA-X may result in unusual behavior.

Accessing the HWAuxIO object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim AuxIO As HWAuxIO
Set AuxIO = app.GetAuxIO

See Also:
- Pinout of the Aux IO Connector
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_InputVoltage</td>
<td>IHWAuxIO</td>
<td>Superseded by get_InputVoltageEX</td>
</tr>
<tr>
<td>get_InputVoltageEX</td>
<td>IHWAuxIO5</td>
<td>Reads the ADC input voltage</td>
</tr>
<tr>
<td>get_OutputVoltage</td>
<td>IHWAuxIO</td>
<td>Reads ADC output voltages.</td>
</tr>
<tr>
<td>get_OutputVoltageMode</td>
<td>IHWAuxIO2</td>
<td>Reads mode setting for either DAC output.</td>
</tr>
<tr>
<td>get_PortCData</td>
<td>IHWAuxIO</td>
<td>Reads a 4-bit value from Port C</td>
</tr>
<tr>
<td>put_OutputVoltage</td>
<td>IHWAuxIO</td>
<td>Writes voltages to the DAC/Analog Output 1 and Output 2</td>
</tr>
<tr>
<td>put_OutputVoltageMode</td>
<td>IHWAuxIO2</td>
<td>Writes the mode setting for either DAC output.</td>
</tr>
<tr>
<td>put_PortCData</td>
<td>IHWAuxIO</td>
<td>Writes a 4-bit value to Port C</td>
</tr>
<tr>
<td>Properties</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>FootSwitch</td>
<td>IHWAuxIO Reads the Footswitch Input</td>
<td></td>
</tr>
<tr>
<td>FootswitchMode</td>
<td>IHWAuxIO3 Determines the action that occurs when the footswitch is pressed.</td>
<td></td>
</tr>
<tr>
<td>PassFailLogic</td>
<td>IHWAuxIO Sets and reads the logic of the PassFail line Shared with the HWMaterialHandler Object</td>
<td></td>
</tr>
<tr>
<td>PassFailMode</td>
<td>IHWAuxIO Sets and reads the mode of the PassFail line Shared with the HWMaterialHandler Object</td>
<td></td>
</tr>
<tr>
<td>PassFailPolicy</td>
<td>IHWAuxIO4 Sets the policy used to determine how global pass/fail is computed. Shared with the HWMaterialHandler Object</td>
<td></td>
</tr>
<tr>
<td>PassFailScope</td>
<td>IHWAuxIO Sets and reads the scope of the PassFail line Shared with the HWMaterialHandler Object</td>
<td></td>
</tr>
<tr>
<td>PassFailStatus</td>
<td>IHWAuxIO4 Returns the most recent pass/fail status value. Shared with the HWMaterialHandler Object</td>
<td></td>
</tr>
<tr>
<td>PortCLogic</td>
<td>HWAuxIO Sets and reads the logic mode of Port C</td>
<td></td>
</tr>
<tr>
<td>PortCMode</td>
<td>HWAuxIO Sets and reads the mode of Port C</td>
<td></td>
</tr>
<tr>
<td>SweepEndMode</td>
<td>HWAuxIO Sets and reads the event that causes the Sweep End line to go to a false state. Shared with the HWMaterialHandler Object</td>
<td></td>
</tr>
</tbody>
</table>

IHWAuxIO History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IHWAuxIO</td>
<td>2.0</td>
</tr>
<tr>
<td>IHWAuxIO2</td>
<td>3.0</td>
</tr>
<tr>
<td>IHWAuxIO3</td>
<td>3.0</td>
</tr>
<tr>
<td>IHWAuxIO4</td>
<td>5.0</td>
</tr>
<tr>
<td>IHWAuxIO5</td>
<td>7.5</td>
</tr>
</tbody>
</table>

Last Modified:

- 10-Jul-2007    Added new command
- 29-Jun-2007    Updated for PNA-X ADC commands
HWExternalTestSetIO Object

Description
Contains the methods and properties that control the rear panel External Test Set Input / Output connector

Accessing the HWExternalTestSetIO object

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim ExtTS As HWExternalTestSetIO
Set ExtTS = app.GetExternalTestSetIO
```

See Also:
- Pinout of the Aux IO Connector
- Pinout for the External Test Set Connector
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadData</td>
<td>Reads data and generates the appropriate timing signals</td>
</tr>
<tr>
<td>ReadRaw</td>
<td>Reads data, but does NOT generate appropriate timing signals</td>
</tr>
<tr>
<td>WriteData</td>
<td>Writes data and generates the appropriate timing signals</td>
</tr>
<tr>
<td>WriteRaw</td>
<td>Writes data, but does NOT generate the appropriate timing signals</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupt</td>
<td>Returns the state of the Interrupt line</td>
</tr>
<tr>
<td>SweepHoldOff</td>
<td>Returns the state of the Sweep Holdoff line</td>
</tr>
</tbody>
</table>

IHWExternalTestSetIO History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWExternalTestSetIO</td>
<td>2.0</td>
</tr>
</tbody>
</table>
HWMaterialHandlerIO Object

Description
Contains the methods and properties that control the rear panel Material Handler Input / Output connector.

Accessing the HWMaterialHandlerIO object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim MatHdlr As HWMaterialHandlerIO
Set MatHdlr = app.GetMaterialHandlerIO

See Also:
- Pinout for the Material HandlerIO Connector
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_Input1</td>
<td>HWMaterialHandlerIO</td>
<td>Reads a hardware latch that captures low to high transition on Input1</td>
</tr>
<tr>
<td>get_Output</td>
<td>HWMaterialHandlerIO</td>
<td>Returns the last value written to the selected output pin.</td>
</tr>
<tr>
<td>get_Port</td>
<td>HWMaterialHandlerIO</td>
<td>Returns the value from the specified &quot;readable&quot; port.</td>
</tr>
<tr>
<td>put_Output</td>
<td>HWMaterialHandlerIO</td>
<td>Writes a TTL HI or TTL Low to output pins 3 or 4.</td>
</tr>
<tr>
<td>put_Port</td>
<td>HWMaterialHandlerIO</td>
<td>Writes a value to the specified port.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IndexState</td>
<td>HWMaterialHandlerIO2</td>
</tr>
<tr>
<td>ReadyForTriggerState</td>
<td>HWMaterialHandlerIO2</td>
</tr>
</tbody>
</table>
**PassFailLogic**  
HWMaterialHandlerIO  
Sets and reads the logic of the PassFail line  
Shared with the HWAuxIO Object

**PassFailMode**  
HWMaterialHandlerIO  
Sets and reads the mode for the PassFail line  
Shared with the HWAuxIO Object

**PassFailPolicy**  
HWMaterialHandlerIO2  
Sets the policy used to determine how global pass/fail is computed.  
Shared with the HWAuxIO Object

**PassFailScope**  
HWMaterialHandlerIO  
Sets and reads the scope for the PassFail line  
Shared with the HWAuxIO Object

**PassFailStatus**  
HWMaterialHandlerIO2  
Returns the most recent pass/fail status value.  
Shared with the HWAuxIO Object

**PortLogic**  
HWMaterialHandlerIO  
Sets and returns the logic mode of data ports A-H

**PortMode**  
HWMaterialHandlerIO  
Sets and returns whether Port C or Port D is used for writing or reading data

**SweepEndMode**  
HWMaterialHandlerIO  
Sets and reads the event that cause the Sweep End line to go to a low state.  
Shared with the HWAuxIO Object

---

**HWMaterialHandlerIO History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWMaterialHandlerIO</td>
<td>2.0</td>
</tr>
<tr>
<td>HWMaterialHandlerIO2</td>
<td>5.0</td>
</tr>
</tbody>
</table>
IFConfiguration Object

Description
These properties control the IF gain and source path settings for the following:

- E836x Opt H11 - all IFConfiguration and IFConfiguration2 commands
- PNA-X - IFConfiguration3 commands ONLY

Accessing the IFConfiguration object

Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim cfg as IIFConfiguration
Set cfg = chan.IFConfiguration

See Also:

- SignalProcessingModuleFour Object (PNA-X ONLY)
- PulseGenerator Object (PNA-X ONLY)
- IF Path Configuration (PNA-X ONLY)
- IF Access User Interface Settings
- PNA Automation Interfaces
- The PNA Object Model
- Pulsed Application
- Pulsed Measurement Example
### Methods

None

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFFilterSampleCount</td>
<td>IFConfiguration2</td>
<td>Sets or returns the number of taps in the IF filter.</td>
</tr>
<tr>
<td>IFFilterSamplePeriod</td>
<td>IFConfiguration2</td>
<td>Sets or returns the IF filter sample period time.</td>
</tr>
<tr>
<td>IFFilterSamplePeriodList</td>
<td>IFConfiguration2</td>
<td>Returns the list of available IF filter sample periods for the instrument.</td>
</tr>
<tr>
<td>IFFilterSamplePeriodMode</td>
<td>IFConfiguration2</td>
<td>Sets or returns the IF filter sample period mode.(Auto or Manual).</td>
</tr>
<tr>
<td>IFFilterSource</td>
<td>IFConfiguration2</td>
<td>Sets or retrieves type of IF filter to be used.</td>
</tr>
<tr>
<td>IFFrequency</td>
<td>IFConfiguration3</td>
<td>Sets IF frequency in manual mode.</td>
</tr>
<tr>
<td>IFFrequencyMode</td>
<td>IFConfiguration3</td>
<td>Sets IF frequency mode to automatic or manual.</td>
</tr>
<tr>
<td>IFGainLevel</td>
<td>IFConfiguration</td>
<td>Sets the gain level for the specified receiver.</td>
</tr>
<tr>
<td>IFGainMode</td>
<td>IFConfiguration</td>
<td>Sets the gain state for ALL receivers.</td>
</tr>
<tr>
<td>IFFGateEnable</td>
<td>IFConfiguration2</td>
<td>Sets or retrieves the state of the IF Gate.</td>
</tr>
<tr>
<td>IFSOriginPath</td>
<td>IFConfiguration</td>
<td>Sets the source path of the specified receiver to Internal or External.</td>
</tr>
<tr>
<td>MaximumIFFilterSampleCount</td>
<td>IFConfiguration2</td>
<td>Returns the maximum allowed value for the IFFilterSampleCount.</td>
</tr>
<tr>
<td>MinimumIFFilterSampleCount</td>
<td>IFConfiguration2</td>
<td>Returns the minimum allowed value for the IFFilterSampleCount.</td>
</tr>
<tr>
<td>MaximumIFFrequency</td>
<td>IFConfiguration3</td>
<td>Returns the maximum IF frequency setting</td>
</tr>
<tr>
<td>MinimumIFFrequency</td>
<td>IFConfiguration3</td>
<td>Returns the minimum IF frequency setting</td>
</tr>
</tbody>
</table>
**IFConfiguration History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIFFCOnfiguration</td>
<td>4.0</td>
</tr>
<tr>
<td>IIFFCOnfiguration2</td>
<td>4.0</td>
</tr>
<tr>
<td>IIFFCOnfiguration3</td>
<td>7.2</td>
</tr>
</tbody>
</table>

785
**IMixer Interface (Option 083)**

**Description**
Contains the methods and properties to setup FCA Mixer measurements. For performing calibrations, use either the SMC Type Object or the VMC Type Object.

**Accessing the IMixer Interface**
Access the IMixer Interface through the Measurement Object. If the particular type of Measurement that was created supports IMixer, then the program determines this at run time and can access the functionality exposed by IMixer. Because the determination of IMixer support is not made until runtime, the program should handle the case where IMixer is not supported on the object.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", "analyzerName")
app.Preset
' FCA Measurements can't share the channel with standard measurements
' Because preset creates a single measurement in channel 1, we first delete the standard measurement
Dim standardMeas As IMeasurement
Set standardMeas = app.ActiveMeasurement
standardMeas.Delete

' Create a Measurement object, in this case using the IMeasurement interface
Dim meas As IMeasurement
Set meas = app.CreateCustomMeasurementEx(1, "SMC_Forward.SMC_ForwardMeas", "SC21")

' See if this measurement object supports IMixer
Dim mixer As IMixer

See an example program that shows how to create and calibrate a standard SMC or VMC measurement or a fixed output SMC measurement.

**See Also:**
PNA Automation Interfaces
The PNA Object Model

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apply</td>
<td>IMixer3</td>
<td>Applies mixer settings.</td>
</tr>
<tr>
<td>Calculate</td>
<td>IMixer</td>
<td>Automatically calculate Input and Output frequencies for mixer setup.</td>
</tr>
<tr>
<td>LoadFile</td>
<td>IMixer</td>
<td>Loads a previously-configured mixer attributes file (.mxr)</td>
</tr>
</tbody>
</table>
SaveFile IMixer Saves the settings for the mixer/ converter test setup to a mixer attributes file.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveXAxisRange IMixer3</td>
<td>Sets or returns the swept parameter to display on the X-axis.</td>
</tr>
<tr>
<td>AvoidSpurs IMixer</td>
<td>Sets and returns the state of the avoid spurs feature.</td>
</tr>
<tr>
<td>EmbeddedLO IMixer7</td>
<td>Provides measurements of mixers with an embedded LO.</td>
</tr>
<tr>
<td>IFDenominator IMixer</td>
<td>Sets or returns the denominator value of the IF Fractional Multiplier.</td>
</tr>
<tr>
<td>IFNumerator IMixer</td>
<td>Sets or returns the numerator value of the IF Fractional Multiplier.</td>
</tr>
<tr>
<td>IFSideband IMixer</td>
<td>Sets or returns the value of the IF sideband.</td>
</tr>
<tr>
<td>IFStartFrequency IMixer</td>
<td>Returns the start frequency of the mixer IF.</td>
</tr>
<tr>
<td>IStopFrequency IMixer</td>
<td>Returns the stop frequency of the mixer IF.</td>
</tr>
<tr>
<td>InputDenominator IMixer</td>
<td>Sets or returns the denominator value of the Input Fractional Multiplier.</td>
</tr>
<tr>
<td>InputFixedFrequency IMixer6</td>
<td>Sets or returns the mixer fixed Input frequency value.</td>
</tr>
<tr>
<td>InputNumerator IMixer</td>
<td>Sets or returns the numerator value of the Input Fractional Multiplier.</td>
</tr>
<tr>
<td>InputPower IMixer</td>
<td>Sets or returns the value of the Input Power.</td>
</tr>
<tr>
<td>InputRangeMode IMixer6</td>
<td>Sets or returns the Input sweep mode.</td>
</tr>
<tr>
<td>InputStartFrequency IMixer</td>
<td>Sets or returns the start frequency of the mixer input.</td>
</tr>
<tr>
<td>InputStopFrequency IMixer</td>
<td>Sets or returns the stop frequency of the mixer input.</td>
</tr>
<tr>
<td>IsInputGreaterThanLO IMixer2</td>
<td>Specifies whether to use the Input frequency that is greater than the LO or I</td>
</tr>
<tr>
<td>LODenominator IMixer</td>
<td>Sets or returns the denominator value of the LO Fractional Multiplier.</td>
</tr>
<tr>
<td>LOFixedFrequency IMixer</td>
<td>Sets or returns the fixed frequency of the specified LO.</td>
</tr>
<tr>
<td>LOName IMixer</td>
<td>Sets or returns the LO name.</td>
</tr>
<tr>
<td>LONumerator IMixer</td>
<td>Sets or returns the numerator value of the LO Fractional Multiplier.</td>
</tr>
<tr>
<td>LOPower IMixer</td>
<td>Sets or returns the value of the LO Power.</td>
</tr>
<tr>
<td>LORangeMode IMixer3</td>
<td>Sets or returns the LO sweep mode to fixed or swept.</td>
</tr>
<tr>
<td>LOStage IMixer</td>
<td>Returns the number of stages.</td>
</tr>
</tbody>
</table>
LOStartFrequency | IMixer3  | Sets or returns the start frequency of the specified LO.
LOStopFrequency  | IMixer3  | Sets or returns the start frequency of the specified LO.
NominalIncidentPowerState | IMixer4 | Toggles Nominal Incident Power ON and OFF.
OutputFixedFrequency | IMixer3 | Sets or returns the fixed frequency of the mixer output.
OutputRangeMode   | IMixer6  | Sets or returns the Output sweep mode.
OutputSideband    | IMixer  | Sets or returns the value of the output sideband.
OutputStartFrequency | IMixer  | Sets or returns the start frequency of the mixer output.
OutputStopFrequency | IMixer  | Sets or returns the stop frequency of the mixer output.

**IMixer History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMixer</td>
<td>1.0</td>
</tr>
<tr>
<td>IMixer2</td>
<td>3.5</td>
</tr>
<tr>
<td>IMixer3</td>
<td>4.0</td>
</tr>
<tr>
<td>IMixer4</td>
<td>4.8</td>
</tr>
<tr>
<td>IMixer5</td>
<td>6.04</td>
</tr>
<tr>
<td>IMixer6</td>
<td>6.20</td>
</tr>
<tr>
<td>IMixer7</td>
<td>7.21</td>
</tr>
</tbody>
</table>
**InterfaceControl Object**

**Description**
Contains the methods and properties that support Interface Control.

**Accessing the InterfaceControl object**

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim IntControl As InterfaceControl
Set IntControl = app.InterfaceControl
```

**See Also:**
- PNA Automation Interfaces
- The PNA Object Model
- Interface Control Feature
- Example Programs

### Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ConfigurationFile</code></td>
<td>InterfaceControl</td>
</tr>
<tr>
<td></td>
<td>Recalls an Interface Control file</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>State</code></td>
<td>InterfaceControl</td>
</tr>
<tr>
<td></td>
<td>Turns Interface Control ON and OFF</td>
</tr>
</tbody>
</table>

**InterfaceControl History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>InterfaceControl</td>
<td>5.2</td>
</tr>
</tbody>
</table>
Limit Test Collection

Description
Child of the Measurement Object. A collection that provides a mechanism for iterating through the Measurement's Limit Segment objects (Limit Lines). The collection has 100 limit lines by default.

Accessing the LimitTest collection
Get a handle to an individual limit segment by specifying an item of the LimitTest collection.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim limSegs As LimitTest
Set limSegs = app.ActiveMeasurement.LimitTest
limSegs.Item(1).BeginResponse = 1000000000#
```

See Also:
- LimitSegment Object
- Collections in the Analyzer
- The PNA Object Model
- Limit Line Testing Example

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetTestResult</td>
<td>Retrieves the Pass/Fail results of the Limit Test (State).</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle on a limit line in the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of limit lines used in the measurement.</td>
</tr>
<tr>
<td>LineDisplay</td>
<td>Displays the limit lines on the screen.</td>
</tr>
<tr>
<td>SoundOnFail</td>
<td>Enables a beep on Limit Test fails.</td>
</tr>
<tr>
<td>State</td>
<td>Turns ON and OFF limit testing.</td>
</tr>
</tbody>
</table>

LimitTest History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILimitTest</td>
<td>1.0</td>
</tr>
</tbody>
</table>
LimitSegment Object

Description
The LimitSegment object is an individual limit line.

Accessing the LimitSegment object
Get a handle to an individual limit line by using the LimitTest collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim limSegs As LimitTest
Set limSegs = app.ActiveMeasurement.LimitTest
limSegs(1).BeginResponse = 1000000000#
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

### Methods

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BeginResponse</td>
</tr>
<tr>
<td>BeginStimulus</td>
</tr>
<tr>
<td>EndResponse</td>
</tr>
<tr>
<td>EndStimulus</td>
</tr>
<tr>
<td>Type</td>
</tr>
</tbody>
</table>

### LimitSegment History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILimitSegment</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Marker Object

Description
Contains the methods and properties that control Markers. There are 10 markers available per measurement:

- 1 reference marker
- 9 markers for absolute data or data relative to the reference marker (delta markers).

There are two ways to control markers through COM.

1. The Measurement object has properties that apply to ALL of the markers for that measurement. For example, `meas.MarkerFormat = naLinMag` applies formatting to all markers.

2. Marker object properties override the Measurement object properties. For example, you can then override the format setting for an individual marker by specifying `mark.Format = naLogMag` on the marker object.

Note: `SearchFilterBandwidth` is available through the measurement object.

Accessing the Marker object
To turn ON a marker, get a handle to the marker through the measurement object. If not already activated, this command will turn ON marker 1

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

app.ActiveMeasurement.marker(1).Format = naLinMag
```

You can also set the marker object to an object variable:

```
Dim m1 As Marker
Set m1 = app.ActiveMeasurement.Marker(1)
m1.Format = naMarkerFormat_LinMag
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activate</td>
<td>IMarker</td>
<td>Makes an object the Active Object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Marker Object</td>
</tr>
<tr>
<td>SearchMax</td>
<td>IMarker</td>
<td>Searches the marker domain for the maximum value.</td>
</tr>
<tr>
<td><strong>SearchMin</strong></td>
<td>IMarker</td>
<td>Searches the marker domain for the minimum value.</td>
</tr>
<tr>
<td><strong>SearchNextPeak</strong></td>
<td>IMarker</td>
<td>Searches the marker's domain for the next largest peak value.</td>
</tr>
<tr>
<td><strong>SearchPeakLeft</strong></td>
<td>IMarker</td>
<td>Searches the marker's domain for the next VALID peak to the left of the marker.</td>
</tr>
<tr>
<td><strong>SearchPeakRight</strong></td>
<td>IMarker</td>
<td>Searches the marker's domain for the next VALID peak to the right of the marker.</td>
</tr>
<tr>
<td><strong>SearchTarget</strong></td>
<td>IMarker</td>
<td>Searches the marker's domain for the target value.</td>
</tr>
<tr>
<td><strong>SearchTargetLeft</strong></td>
<td>IMarker</td>
<td>Moving to the left of the marker position, searches the marker's domain for the target value.</td>
</tr>
<tr>
<td><strong>SearchTargetRight</strong></td>
<td>IMarker</td>
<td>Moving to the right of the marker position, searches the marker's domain for the target value.</td>
</tr>
<tr>
<td><strong>SetCenter</strong></td>
<td>IMarker</td>
<td>Changes the analyzer's center frequency to the X-axis position of the marker.</td>
</tr>
<tr>
<td><strong>SetCW</strong></td>
<td>IMarker</td>
<td>Changes the analyzer to sweep type CW mode and makes the CW frequency the marker's frequency.</td>
</tr>
<tr>
<td><strong>SetElectricalDelay</strong></td>
<td>IMarker</td>
<td>Changes the measurement's electrical delay to the marker's delay value.</td>
</tr>
<tr>
<td><strong>SetReferenceLevel</strong></td>
<td>IMarker</td>
<td>Changes the measurement's reference level to the marker's Y-axis value.</td>
</tr>
<tr>
<td><strong>SetStart</strong></td>
<td>IMarker</td>
<td>Changes the analyzer's start frequency to the X-axis position of the marker.</td>
</tr>
<tr>
<td><strong>SetStop</strong></td>
<td>IMarker</td>
<td>Changes the analyzer's stop frequency to the X-axis position of the marker.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th><strong>Properties</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bucket Number</strong></td>
<td>IMarker</td>
</tr>
<tr>
<td><strong>DeltaMarker</strong></td>
<td>IMarker</td>
</tr>
<tr>
<td><strong>Distance</strong></td>
<td>IMarker2</td>
</tr>
<tr>
<td><strong>Format</strong></td>
<td>IMarker</td>
</tr>
<tr>
<td><strong>Interpolated</strong></td>
<td>IMarker</td>
</tr>
<tr>
<td><strong>Number</strong></td>
<td>IMarker</td>
</tr>
<tr>
<td><strong>PeakExcursion</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
**PeakThreshold**  IMarker  Sets peak threshold for the specified marker.

**SearchFunction**  IMarker  Emulates the Tracking function in the marker search dialog box.

**Stimulus**  IMarker  Sets and reads the X-Axis value of the marker.

**Target Value**  IMarker  Sets the target value for the marker when doing Target Searches.

**Tracking**  IMarker  The tracking function finds the selected search function every sweep.

**Type**  IMarker  Sets and reads the marker type.

**UserRange**  IMarker  Assigns the marker to the specified User Range.

**UserRangeMax**  IMarker  Sets the stimulus stop value for the specified User Range.

**UserRangeMin**  IMarker  Sets the stimulus start value for the specified User Range.

**Value**  IMarker  Reads the Y-Axis value of the marker.

### Marker History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMarker</td>
<td>1.0</td>
</tr>
<tr>
<td>IMarker2</td>
<td>4.2</td>
</tr>
</tbody>
</table>
Measurement Object

See IArrayTransfer Interface for putting and getting typed data.
See IMixer Interface (used with Option 083)

Description

The Measurement object is probably the most used object in the PNA Object Model. A measurement object represents the chain of data processing algorithms that take raw data from the channel and make it ready for display, which then becomes the scope of the Trace object.

A Measurement object is defined by its parameter (S11, S22, A/R1, B and so forth). The measurement object is associated with a channel which drives the hardware that produces the data that feeds the measurement. The root of a measurement is the raw data. This buffer of complex paired data then flows through a number of processing blocks: error-correction, trace math, phase correction, time domain, gating, formatting. All of these are controlled through the measurement object.

The ACTIVE measurement is the measurement that will be acted upon if you make a setting from the front panel. It is the measurement whose "button" is pressed in the window with the red "active window" frame. If you create a new measurement, that measurement becomes the active measurement.

Therefore, all automation methods with the word "Active" in them refer to the object associated with the Active measurement, whether that object is a Channel, Window, Trace or Limit line.

Learn about the IMeasurement2 Interface for reading stimulus properties.

Accessing the Measurement object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim meas As IMeasurement
Set meas = app.ActiveMeasurement

or

Set meas = app.Measurements(n)
```

You can access four other objects through the Measurement object: markers, limit test, transform, and gating. For example, because each measurement has its own set of markers, you can set a marker by doing this:

```vba
Dim meas as measurement
Set meas = app.ActiveMeasurement
meas.marker(1).Stimulus = 900e6
```

IMeasurement2 Interface

Some of the properties and methods for the IMeasurement2 Interface return stimulus values that are set using the channel object. The following is the reason these properties and methods are duplicated.

Every measurement carries with it a snapshot of the stimulus properties of the channel that were in effect when the measurement last acquired data. Therefore, it is the measurement that provides the most accurate stimulus description of its data. Any change made to the channel after the measurement was acquired renders the IChannel interface unreliable in terms of describing the measurement.

See Also:
### PNA Automation Interfaces

- **The PNA Object Model**
- **Example Programs**
- **Superseded commands**

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Activate</strong></td>
<td>IMeasurement</td>
<td>Makes an object the Active Object. Shared with the Marker Object</td>
</tr>
<tr>
<td><strong>ActivateMarker</strong></td>
<td>IMeasurement</td>
<td>Makes a marker the Active Marker.</td>
</tr>
<tr>
<td><strong>ChangeParameter</strong></td>
<td>IMeasurement</td>
<td>Changes the parameter of the measurement.</td>
</tr>
<tr>
<td><strong>DataToDivisor</strong></td>
<td>IMeasurement</td>
<td><strong>Superseded with</strong> DoReceiverPowerCal Method</td>
</tr>
<tr>
<td><strong>DataToMemory</strong></td>
<td>IMeasurement</td>
<td>Stores the active measurement into memory.</td>
</tr>
<tr>
<td><strong>Delete</strong></td>
<td>IMeasurement</td>
<td>Deletes the measurement object.</td>
</tr>
<tr>
<td><strong>DeleteAllMarkers</strong></td>
<td>IMeasurement</td>
<td>Deletes all of the markers from the measurement.</td>
</tr>
<tr>
<td><strong>DeleteMarker</strong></td>
<td>IMeasurement</td>
<td>Deletes a marker from the active measurement.</td>
</tr>
<tr>
<td><strong>getData</strong></td>
<td>IMeasurement</td>
<td>Retrieves Complex data from analyzer memory</td>
</tr>
<tr>
<td><strong>getDataByUrl</strong></td>
<td>IMeasurement</td>
<td>Retrieves variant data from the specified location in your choice of formats.</td>
</tr>
<tr>
<td><strong>GetFilterStatistics</strong></td>
<td>IMeasurement</td>
<td>Returns all four Filter Statistics</td>
</tr>
<tr>
<td><strong>GetReferenceMarker</strong></td>
<td>IMeasurement</td>
<td>Returns a handle to the reference marker.</td>
</tr>
<tr>
<td><strong>GetSnpData</strong></td>
<td>IMeasurement3</td>
<td>Returns SnP data.</td>
</tr>
<tr>
<td><strong>GetSnpDataWithSpecifiedPorts</strong></td>
<td>IMeasurement7</td>
<td>Returns sNp data for the specified ports.</td>
</tr>
<tr>
<td><strong>GetTraceStatistics</strong></td>
<td>IMeasurement</td>
<td>Returns the Trace Statistics.</td>
</tr>
<tr>
<td><strong>GetXAxisValues</strong></td>
<td>IMeasurement2</td>
<td>Returns the stimulus values for the measurement.</td>
</tr>
<tr>
<td><strong>InterpolateMarkers</strong></td>
<td>IMeasurement</td>
<td>Turns All Marker Interpolation ON and OFF for the measurement.</td>
</tr>
<tr>
<td><strong>putDataComplex</strong></td>
<td>IMeasurement</td>
<td>Puts complex data into one of five data buffers.</td>
</tr>
<tr>
<td><strong>putDataScalar</strong></td>
<td>IMeasurement</td>
<td>Puts formatted variant data into the measurement results buffer.</td>
</tr>
</tbody>
</table>
### SearchFilterBandwidth

*IMeasurement*

Searches the domain with the current BW target.

### WriteSnpFileWithSpecifiedPorts

*IMeasurement7*

Write sNp data for specified ports to a file.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ActiveMarker</strong></td>
<td><em>IMeasurement</em></td>
<td>Returns a handle to the Active Marker object.</td>
</tr>
<tr>
<td><strong>BalancedMeasurement</strong></td>
<td><em>IMeasurement</em></td>
<td>Sets the measurement type that is used with balanced topologies.</td>
</tr>
<tr>
<td><strong>BandwidthTarget</strong></td>
<td><em>IMeasurement</em></td>
<td>The insertion loss value at which the bandwidth of a filter is measured.</td>
</tr>
<tr>
<td><strong>BandwidthTracking</strong></td>
<td><em>IMeasurement</em></td>
<td>Turns Bandwidth Tracking function ON and OFF.</td>
</tr>
<tr>
<td><strong>CalibrationName</strong></td>
<td><em>IMeasurement2</em></td>
<td>Returns the name of the cal type.</td>
</tr>
<tr>
<td><strong>CalibrationType</strong></td>
<td><em>IMeasurement</em></td>
<td><strong>Superseded with</strong> <em>CalibrationTypeID</em> property</td>
</tr>
<tr>
<td><strong>CalibrationTypeID</strong></td>
<td><em>IMeasurement2</em></td>
<td>Sets or returns the cal type for the current measurement.</td>
</tr>
<tr>
<td><strong>Center</strong></td>
<td><em>IMeasurement2</em></td>
<td>Returns the stimulus value of the center point for the measurement.</td>
</tr>
<tr>
<td><strong>channelNumber</strong></td>
<td><em>IMeasurement</em></td>
<td>Returns the channel number. Shared with the Channel Object</td>
</tr>
<tr>
<td><strong>Domain</strong></td>
<td><em>IMeasurement2</em></td>
<td>Returns the domain (frequency, time, power) for the measurement.</td>
</tr>
<tr>
<td><strong>ElectricalDelay</strong></td>
<td><em>IMeasurement</em></td>
<td>Sets electrical delay.</td>
</tr>
<tr>
<td><strong>ElecDelayMedium</strong></td>
<td><em>IMeasurement2</em></td>
<td>Sets or returns the characteristic of the electrical delay medium.</td>
</tr>
<tr>
<td><strong>Equation</strong></td>
<td><em>IMeasurement6</em></td>
<td>Access Equation Editor</td>
</tr>
<tr>
<td><strong>ErrorCorrection</strong></td>
<td><em>IMeasurement</em></td>
<td>Set or get the state of error correction for the measurement.</td>
</tr>
<tr>
<td><strong>FilterBW</strong></td>
<td><em>IMeasurement</em></td>
<td>Returns the results of the SearchBandwidth method.</td>
</tr>
<tr>
<td><strong>FilterCF</strong></td>
<td><em>IMeasurement</em></td>
<td>Returns the Center Frequency result of the SearchBandwidth method.</td>
</tr>
<tr>
<td><strong>FilterLoss</strong></td>
<td><em>IMeasurement</em></td>
<td>Returns the Loss value of the SearchBandwidth method.</td>
</tr>
<tr>
<td><strong>FilterQ</strong></td>
<td><em>IMeasurement</em></td>
<td>Returns the Q (quality factor) result of the SearchBandwidth method.</td>
</tr>
<tr>
<td><strong>Format</strong></td>
<td>IMeasurement</td>
<td>Sets display format.</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><strong>Gating</strong></td>
<td>IMeasurement</td>
<td>Controls Time Domain Gating.</td>
</tr>
<tr>
<td><strong>InterpolateCorrection</strong></td>
<td>IMeasurement</td>
<td>Turns ON and OFF the calculation of new error terms when stimulus values change.</td>
</tr>
<tr>
<td><strong>InterpolateNormalization</strong></td>
<td>IMeasurement</td>
<td>Superseded with DoReceiverPowerCal Method</td>
</tr>
<tr>
<td><strong>IsSparameter</strong></td>
<td>IMeasurement2</td>
<td>Returns true if measurement represents an S-Parameter.</td>
</tr>
<tr>
<td><strong>LimitTest</strong></td>
<td>IMeasurement</td>
<td>Collection for iterating through the Limit Segment objects (Limit Lines).</td>
</tr>
<tr>
<td><strong>LimitTestFailed</strong></td>
<td>IMeasurement</td>
<td>Returns the results of limit testing</td>
</tr>
<tr>
<td><strong>LoadPort</strong></td>
<td>IMeasurement</td>
<td>Returns the load port number associated with an S-parameter reflection measurement.</td>
</tr>
<tr>
<td><strong>LogMagnitudeOffset</strong></td>
<td>IMeasurement</td>
<td>Superseded with DoReceiverPowerCal Method</td>
</tr>
<tr>
<td><strong>MagnitudeOffset</strong></td>
<td>IMeasurement4</td>
<td>Offsets the magnitude of the entire data trace to a specified value.</td>
</tr>
<tr>
<td><strong>MagnitudeSlopeOffset</strong></td>
<td>IMeasurement4</td>
<td>Offsets the magnitude of the data trace to a value that changes linearly with frequency.</td>
</tr>
<tr>
<td><strong>Marker</strong></td>
<td>IMeasurement</td>
<td>Contains the methods and properties that control Markers.</td>
</tr>
<tr>
<td><strong>MarkerFormat</strong></td>
<td>IMeasurement</td>
<td>Sets or returns the format of all the markers in the measurement.</td>
</tr>
<tr>
<td><strong>Marker State</strong></td>
<td>IMeasurement3</td>
<td>Sets or returns the ON / OFF state of a marker.</td>
</tr>
<tr>
<td><strong>Mean</strong></td>
<td>IMeasurement</td>
<td>Returns the mean value of the measurement.</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td>IMeasurement</td>
<td>Sets or returns the name of the measurement.</td>
</tr>
<tr>
<td><strong>NAWindow</strong></td>
<td>IMeasurement</td>
<td>Controls the part of the display that contains the graticule, or what is written on the display.</td>
</tr>
<tr>
<td><strong>Normalization</strong></td>
<td>IMeasurement</td>
<td>Superseded with DoReceiverPowerCal Method</td>
</tr>
<tr>
<td><strong>Number</strong></td>
<td>IMeasurement</td>
<td>Returns the number of the measurement.</td>
</tr>
<tr>
<td><strong>NumberOfPoints</strong></td>
<td>IMeasurement2</td>
<td>Returns the Number of Points of the measurement.</td>
</tr>
<tr>
<td><strong>Parameter</strong></td>
<td>IMeasurement</td>
<td>Returns the measurement Parameter.</td>
</tr>
<tr>
<td>Method</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PeakToPeak</td>
<td>IMeasurement</td>
<td>Returns the Peak to Peak value of the measurement.</td>
</tr>
<tr>
<td>PhaseOffset</td>
<td>IMeasurement</td>
<td>Sets the Phase Offset for the active channel.</td>
</tr>
<tr>
<td>ReceivePort</td>
<td>IMeasurement2</td>
<td>Returns the receiver port of the measurement.</td>
</tr>
<tr>
<td>ReferenceMarkerState</td>
<td>IMeasurement</td>
<td>Turns the reference marker ON or OFF</td>
</tr>
<tr>
<td>ShowStatistics</td>
<td>IMeasurement</td>
<td>Displays and hides the measurement statistics (peak-to-peak, mean, standard deviation) on the screen.</td>
</tr>
<tr>
<td>Smoothing</td>
<td>IMeasurement</td>
<td>Turns ON and OFF data smoothing.</td>
</tr>
<tr>
<td>SmoothingAperture</td>
<td>IMeasurement</td>
<td>Specifies or returns the amount of smoothing as a ratio of the number of data points in the measurement trace.</td>
</tr>
<tr>
<td>SourcePort</td>
<td>IMeasurement2</td>
<td>Returns the source port of the measurement.</td>
</tr>
<tr>
<td>Span</td>
<td>IMeasurement2</td>
<td>Returns the stimulus span (stop - start) for the measurement.</td>
</tr>
<tr>
<td>StandardDeviation</td>
<td>IMeasurement</td>
<td>Returns the standard deviation of the measurement.</td>
</tr>
<tr>
<td>Start</td>
<td>IMeasurement2</td>
<td>Returns the stimulus value of the first point for the measurement.</td>
</tr>
<tr>
<td>StatisticsRange</td>
<td>IMeasurement</td>
<td>Sets the User Range number for calculating measurement statistics.</td>
</tr>
<tr>
<td>Stop</td>
<td>IMeasurement2</td>
<td>Returns the stimulus value of the last point for the measurement.</td>
</tr>
<tr>
<td>Trace</td>
<td>IMeasurement</td>
<td>Controls scale, reference position, and reference line.</td>
</tr>
<tr>
<td>TraceMath</td>
<td>IMeasurement</td>
<td>Performs math operations on the measurement object and the trace stored in memory.</td>
</tr>
<tr>
<td>TraceTitle</td>
<td>IMeasurement8</td>
<td>Writes and reads a trace title.</td>
</tr>
<tr>
<td>TraceTitleState</td>
<td>IMeasurement8</td>
<td>Turns trace title ON and OFF</td>
</tr>
<tr>
<td>Transform</td>
<td>IMeasurement</td>
<td>Controls Time Domain transforms.</td>
</tr>
<tr>
<td>View</td>
<td>IMeasurement</td>
<td>Sets (or returns) the type of trace displayed on the screen</td>
</tr>
<tr>
<td>WGCutoffFreq</td>
<td>IMeasurement2</td>
<td>Sets or returns the value of the waveguide cut off frequency</td>
</tr>
</tbody>
</table>

**IMeasurement History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMeasurement</td>
<td>1.0</td>
</tr>
<tr>
<td>IMeasurement2</td>
<td>3.0</td>
</tr>
<tr>
<td>IMeasurement3</td>
<td>4.0</td>
</tr>
<tr>
<td>IMeasurement4</td>
<td>4.2</td>
</tr>
<tr>
<td>IMeasurement5</td>
<td>5.0</td>
</tr>
<tr>
<td>IMeasurement7</td>
<td>6.2</td>
</tr>
</tbody>
</table>

**IArrayTransfer Interface**

**Description**
Contains methods for putting data in and getting data out of the analyzer using typed data. This interface transfers data more efficiently than the IMeasurement Interface. However, this interface is only usable from VB6, C, & C++.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getComplex</td>
<td>Retrieves real and imaginary data from the specified buffer.</td>
</tr>
<tr>
<td>getNAComplex</td>
<td>Retrieves typed <strong>NAComplex</strong> data from the specified buffer.</td>
</tr>
<tr>
<td>getPairedData</td>
<td>Retrieves magnitude and phase data pairs from the specified buffer.</td>
</tr>
<tr>
<td>getScalar</td>
<td>Retrieves scalar data from the specified buffer.</td>
</tr>
<tr>
<td>putComplex</td>
<td>Puts real and imaginary data into the specified buffer.</td>
</tr>
<tr>
<td>putNAComplex</td>
<td>Puts typed <strong>NAComplex</strong> data into the specified buffer.</td>
</tr>
<tr>
<td>putScalar</td>
<td>Puts scalar data into the measurement result buffer.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

**IArrayTransfer History**
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IArrayTransfer</td>
<td>1.0</td>
</tr>
</tbody>
</table>
# Measurement Collection

## Description
A collection object that provides a mechanism for iterating through the Application measurements.

### Accessing the Measurements collection

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim measments As Measurements
Set measments = app.Measurements
```

## See Also:
- [Measurement Object](#)
- [Collections in the Analyzer](#)
- [The PNA Object Model](#)
- [Example Programs](#)

## Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a Measurement to the collection.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle on a measurement in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes a measurement from the measurements collection.</td>
</tr>
</tbody>
</table>

## Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of measurements in the analyzer.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current Application.</td>
</tr>
</tbody>
</table>
NAWindow Object

Description
The NAWindow object controls the part of the display that contains the graticule, or what is written on the display.

Accessing the NaWindow object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim window As NAWindow
Set window = app.NAWindows(1)
window.AutoScale
```
or

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", "analyzerName")

app.NAWindows(1).AutoScale
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoscale</td>
<td>Autoscales all measurements in the window. Practical with the Trace Object</td>
</tr>
<tr>
<td>ShowMarkerReadout</td>
<td>Shows and Hides the Marker readout for the active marker in the upper-right corner of the window object.</td>
</tr>
<tr>
<td>ShowTable</td>
<td>Shows or Hides the specified table for the active measurement in the lower part of the window object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveTrace</td>
<td>Sets a trace to the Active Trace.</td>
</tr>
<tr>
<td>MarkerReadout</td>
<td>Sets and reads the state of the Marker readout for the active marker in the upper-right corner of the window object.</td>
</tr>
</tbody>
</table>
**MarkerReadoutSize**  
Specifies the size of font used when displaying Marker readout in the selected window.

**OneMarkerReadoutPerTrace**  
Either show marker readout of only the active trace or all of the traces simultaneously.

**Title**  
Writes or reads a custom title for the window.

**TitleState**  
Turns ON and OFF the window title.

**Traces**  
Collection for getting a handle to a trace or iterating through the traces in a window.

**WindowNumber**  
Reads the number of the active window.

**WindowState**  
Maximizes or minimizes a window.

Shared with the Application Object

**INaWindow History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>INaWindow</td>
<td>1.0</td>
</tr>
</tbody>
</table>

806
NAWindows Collection

Description
A collection object that provides a mechanism for iterating through the Application windows.

Accessing the NaWindows collection

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim windows As NAWindows
Set windows = app.NAWindows
```

See Also:
- **NAWindow Object**
- **Collections in the Analyzer**
- **The PNA Object Model**
- **Example Programs**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a window to the NAWindows collection.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a window in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes a window from the NAWindows collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of windows on the analyzer.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the current Application.</td>
</tr>
</tbody>
</table>
NoiseCal Object

Description
Controls the noise figure calibration settings. These commands are extensions which supplement the standard calibration commands on the GuidedCalibration Object.

Accessing the NoiseCal object

```vba
Dim app as AgilentPNA835x.Application
Set noisecal = pna.GetCalmanager.CreateCustomCalEx(channelNum)
Set noiseCalExtension = noisecal.CustomCalConfiguration
noiseCalExtension.NoiseSourceCold = 300
```

See Also:

- Example Create and Cal a Noise Figure Measurement
- NoiseFigure Object
- About Noise Figure Measurements
- PNA Automation Interfaces
- The PNA Object Model

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>See History</td>
</tr>
</tbody>
</table>

None

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalMethod</td>
<td>INoiseCal</td>
<td>Sets and returns the method for performing calibration on a noise channel.</td>
</tr>
<tr>
<td>ENRFile</td>
<td>INoiseCal</td>
<td>Sets and returns the name of the ENR file associated with the noise source.</td>
</tr>
<tr>
<td>NoiseSourceCalKitType</td>
<td>INoiseCal</td>
<td>Sets and reads the Cal Kit type used to perform a cal at the adapter which is used to connect the noise source (if required.)</td>
</tr>
</tbody>
</table>
**NoiseSourceCold**

INoiseCal

Sets and returns the current temperature at the noise source.

**NoiseSourceConnectorType**

INoiseCal

Sets and reads the connector type of the noise source used during the cal.

---

**NoiseConfiguration History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>INoiseCal</td>
<td>8.0</td>
</tr>
</tbody>
</table>

Last Modified:

30-May-2007  MX New topic
NoiseFigure Object

Description
Controls the Noise Figure application settings.

Accessing the NoiseFigure object

Dim app as AgilentPNA835x.Application
app.CreateCustomMeasurementEx(1, "NoiseFigure", "NF", 1)
Dim NoiseFig
Set NoiseFig = app.ActiveChannel.CustomChannelConfiguration

See Also:

- Example program Create and Cal a NoiseFigure Measurement
- About Noise Figure Measurements
- Noise Figure Calibration Object
- app.NoiseSourceState (ON and OFF)
- ENRFile Object
- PNA Automation Interfaces
- The PNA Object Model

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>See History</strong></td>
</tr>
</tbody>
</table>

None

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AmbientTemperature</td>
<td>INoiseFigure</td>
<td>Sets the air temperature at which the measurement is being performed.</td>
</tr>
<tr>
<td>ImpedanceStates</td>
<td>INoiseFigure</td>
<td>Sets the number of impedance states to use during calibrated measurements.</td>
</tr>
<tr>
<td>NoiseAverageFactor</td>
<td>INoiseFigure</td>
<td>Set averaging of noise receiver.</td>
</tr>
</tbody>
</table>
**NoiseAverageState**  INoiseFigure  Turn noise averaging ON and OFF

**NoiseBandwidth**  INoiseFigure  Set bandwidth of noise receiver.

**NoiseGain**  INoiseFigure  Set gain state of noise receiver.

**NoiseTuner**  INoiseFigure  Sets and returns the noise tuner identifier,

**NoiseTunerIn**  INoiseFigure  Sets and returns the port identifier of the ECal noise tuner Input

**NoiseTunerOut**  INoiseFigure  Sets and returns the port identifier of the ECal noise tuner Output

### NoiseFigure History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>INoiseFigure</td>
<td>8.0</td>
</tr>
</tbody>
</table>

Last Modified:

29-May-2007  MN New topic
**PathConfiguration Object**

**Description**
Provides access to the path configuration currently active on the channel object.
To load, store, or delete a configuration, see `ConfigurationManager` Object.

**Accessing the PathConfiguration object**

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim chan as Channel
Set chan = app.ActiveChannel

Dim pathConfig As PathConfiguration
Set pathConfig = chan.PathConfiguration
```

**Note:**
To learn how to make configuration (element) settings, see this [Path Configuration Example](#).
Also see this [list of configurable elements and settings](#).

**See Also:**
- `PathConfigurationManager Object`
- `PathElement Object`
- `Path Configurator UI`
- `PNA Automation Interfaces`
- `The PNA Object Model`

### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>See History</strong></td>
<td></td>
</tr>
</tbody>
</table>

#### Elements

IPathConfiguration
Elements are the objects that can be configured (switches and so forth). See the [list of elements](#) and settings.

#### Store

IPathConfiguration
Saves the current configuration to the specified name.

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>See History</strong></td>
<td></td>
</tr>
</tbody>
</table>

#### DescriptiveText
IPathConfiguration
Write and read descriptive text associated with the configuration.
Element  IPathConfiguration  Returns a handle to the IPathElement object.

Name  IPathConfiguration  Returns the name of the current configuration.

Parent  IPathConfiguration  Returns a pointer to the parent COM object (Channel).

IPathConfiguration History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPathConfiguration</td>
<td>7.2</td>
</tr>
</tbody>
</table>
PathConfigurationManager Object

Description
These commands allow configurations to be stored, loaded, or deleted on the PNA.
To make path configuration settings, see PathConfiguration Object and the PathElement Object.

Accessing the PathConfigurationManager object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim pathConfig As PathConfigurationManager
Set pathConfig = app.PathConfigurationManager

Note:
To learn how to make configuration (element) settings, see this Path Configuration Example.
Also see this list of configurable elements and settings.

See Also:
- Path Configuration Example
- Path Configurator
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteConfiguration</td>
<td>IPathConfigurationManager</td>
<td>Deletes the specified configuration from the PNA.</td>
</tr>
<tr>
<td>LoadConfiguration</td>
<td>IPathConfigurationManager</td>
<td>Loads the named configuration.</td>
</tr>
<tr>
<td>StoreConfiguration</td>
<td>IPathConfigurationManager</td>
<td>Saves the path configuration</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configurations</td>
<td>IPathConfigurationManager</td>
<td>Returns a list of configuration names stored in the PNA.</td>
</tr>
</tbody>
</table>
IPathConfigurationManager Returns a handle to the Application object.

**IPathConfigurationManager History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPathConfigurationManager</td>
<td>7.2</td>
</tr>
</tbody>
</table>
**PathElement Object**

**Description**
Provides access to the settings for the PathElement object.

**Accessing the PathElement object**

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim chan as Channel
Set chan = app.ActiveChannel

Dim pathConfig As PathConfiguration
Set pathConfig = chan.PathConfiguration

Dim element as PathElement
Set element = pathConfig.PathElement("Src1")
```

**Note:**
To learn how to make configuration (element) settings, see this [Path Configuration Example](#).
Also see this [list of configurable elements and settings](#).

**See Also:**
- [Path Configurator](#)
- [PathConfigurationManager Object](#)
- [PathConfiguration Object](#)
- [PNA Automation Interfaces](#)
- [The PNA Object Model](#)
- [Example Programs](#)
### Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>See History</td>
<td>None</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>See History</td>
<td>None</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>IPathElement</th>
<th>Returns the name of the element.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent</td>
<td>IPathElement</td>
<td>Returns a pointer to the Parent Object (PathConfiguration)</td>
</tr>
<tr>
<td>Value</td>
<td>IPathElement</td>
<td>Read / Write get the current setting for the element.</td>
</tr>
<tr>
<td>Values</td>
<td>IPathElement</td>
<td>Returns all valid settings for the element.</td>
</tr>
</tbody>
</table>

### IPathElement History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPathElement</td>
<td>7.2</td>
</tr>
</tbody>
</table>
PortExtension Object  Superseded

ALL methods and properties on the PortExtension Object are Superseded with the Fixturing Object.

Description
Contains the methods and properties that control Port Extensions.

Accessing a PortExtension object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PortExt As PortExtension
Set PortExt = app.PortExtension

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- Example Programs
- Superseded commands

Methods

None

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input A</td>
<td>Sets the Input A extension value.</td>
</tr>
<tr>
<td>Input B</td>
<td>Sets the Input B extension value.</td>
</tr>
<tr>
<td>Input C</td>
<td>Sets the Input C extension value.</td>
</tr>
<tr>
<td>Port 1</td>
<td>Sets the Port 1 extension value.</td>
</tr>
<tr>
<td>Port 2</td>
<td>Sets the Port 2 extension value.</td>
</tr>
<tr>
<td>Port 3</td>
<td>Sets the Port 3 extension value.</td>
</tr>
<tr>
<td>State</td>
<td>Turns Port Extensions ON and OFF.</td>
</tr>
</tbody>
</table>

IPort Extension History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPort Extension</td>
<td>1.0</td>
</tr>
</tbody>
</table>
PowerLossSegment Object

Description
Contains the properties describing a segment of the power loss table used in source power calibration.

You can get a handle to one of these segments through the segments.Item Method of the PowerLossSegments collection.

Accessing the PowerLossSegment object
You can get a handle to one of these segments through PowerLossSegments.Item(n)

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PwrLossSeg As PowerLossSegment
Set PwrLossSeg = app.SourcePowerCalibrator.PowerLossSegments(1)
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

Methods
None

Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>The frequency (Hz) associated with this segment.</td>
</tr>
<tr>
<td></td>
<td>Shared with the PowerSensorCalFactorSegment Object</td>
</tr>
<tr>
<td>Loss</td>
<td>The loss value (dB) associated with this segment.</td>
</tr>
<tr>
<td>SegmentNumber</td>
<td>Returns the number of this segment</td>
</tr>
<tr>
<td></td>
<td>Shared with the PowerSensorCalFactorSegment Object</td>
</tr>
</tbody>
</table>

IPowerLossSegment History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerLossSegment</td>
<td>2.0</td>
</tr>
</tbody>
</table>
PowerLossSegments Collection

Description
A collection object that provides a mechanism for iterating through the segments of the power loss table used in source power calibration.

Accessing the PowerLossSegments collection

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PwrLossSegs As PowerLossSegments
Set PwrLossSegs = app.SourcePowerCalibrator.PowerLossSegments
```

See Also:
- PowerLossSegment Object
- Collections in the Analyzer
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a PowerLossSegment object to the collection.</td>
</tr>
<tr>
<td>Item</td>
<td>Use to get a handle to a PowerLossSegment object in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes an object from the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of objects in the collection.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the Parent object (SourcePowerCalibrator) of this collection.</td>
</tr>
</tbody>
</table>
**PowerMeterInterface Object**

**Description**
Contains the properties used to select a power meter and sensor to be used for a source power calibration.

**Note:** This object replaces the `PowerMeterGPIBAAddress Property`.

**Accessing the PowerMeterInterface object**
Get a handle to a power meter object using the `PowerMeterInterfaces` collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim pwrMtrInterfaces As PowerMeterInterfaces
Set pwrMtrInterfaces = app.SourcePowerCalibrator.PowerMeterInterfaces
If pwrMtrInterfaces.Count > 0 Then
    Dim pwrMtrInterface As PowerMeterInterface
    Set pwrMtrInterface = pwrMtrInterfaces(1)
    pwrMtrInterface.Path = naUSB
    pwrMtrInterface.Locator = "Agilent Technologies,U2000A,MY12345678"
End If
```

**See Also:**
- Source Power Calibration
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

**Methods**
None

**Properties**

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Specifies the interface to use: GPIB, USB, LAN</td>
</tr>
<tr>
<td>Locator</td>
<td>Specifies the location (address) of the power meter/sensor.</td>
</tr>
</tbody>
</table>
### IPowerMeterInterface History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerMeterInterface</td>
<td>7.50</td>
</tr>
</tbody>
</table>

**Last Modified:**

- 5-Jul-2007  MX New topic
PowerMeterInterfaces Collection

Description
A collection object that provides a mechanism for accessing the PowerMeterInterface objects.

The collection size is limited to one PowerMeterInterface object. By default, that PowerMeterInterface object refers to GPIB, and to the GPIB address that is currently set for the power meter on that PNA.

The power meter is specified by using the Interface property.

Accessing the PowerMeterInterfaces collection

Get a handle to a power meter object using the PowerMeterInterfaces collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim pwrMtrInterfaces As PowerMeterInterfaces
Set pwrMtrInterfaces = app.SourcePowerCalibrator.PowerMeterInterfaces
If pwrMtrInterfaces.Count > 0 Then
    Dim pwrMtrInterface As PowerMeterInterface
    Set pwrMtrInterface = pwrMtrInterfaces(1)
    pwrMtrInterface.Path = naUSB
    pwrMtrInterface.Locator = "Agilent Technologies,U2000A,MY12345678"
End If
```

See Also:
- Source Power Calibration
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

Methods

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Use to get a handle to a PowerMeterInterface object in the collection.</td>
</tr>
</tbody>
</table>

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of objects in the collection.</td>
</tr>
</tbody>
</table>

IPowerMeterInterfaces History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerMeterInterfaces</td>
<td>7.50</td>
</tr>
</tbody>
</table>

Last Modified:
9-Jul-2007   MX New topic
PowerSensor Object

Description

Each power sensor connected to the power meter associated with Source Power Calibration will have a PowerSensor object created to represent it. These PowerSensor objects reside in the PowerSensors collection within the SourcePowerCalibrator object. You cannot directly create PowerSensor objects, but can only retrieve existing ones from the PowerSensors collection.

The PowerSensorCalFactorSegment object is also accessed through the PowerSensor object. These are accessed through the CalFactorSegments collection in the PowerSensor object.

Accessing a PowerSensor object

Dim pna As AgilentPNA835x.Application
Set pna = CreateObject("AgilentPNA835x.Application", analyzerName)

Dim powerCalibrator as SourcePowerCalibrator
Dim powerSensor as PowerSensor
Dim calFactorSegment as PowerSensorCalFactorSegment

Set powerCalibrator = pna.SourcePowerCalibrator

' Specify GPIB address of the power meter.
powerCalibrator.PowerMeterGPIBAddress = 13

' Each time the PowerSensors collection is accessed, the power meter is queried to determine which channels have sensors attached. The collection is updated accordingly.

If powerCalibrator.PowerSensors.Count > 0 Then
  ' If channel B of the meter has a sensor attached but channel A does not, then element 1 of the collection is sensor B. Whenever channel A has a sensor, sensor A will be element 1.
  Set powerSensor = powerCalibrator.PowerSensors(1)
  ' Insert one new PowerSensorCalFactorSegment at the beginning of the collection (index 1).
  powerSensor.CalFactorSegments.Add(1)
  ' Assign our variable to refer to that object.
  Set calFactorSegment = powerSensor.CalFactorSegments(1)

  ' Set property values for that object.
  calFactorSegment.Frequency = 300000
  ' frequency in Hz
  calFactorSegment.CalFactor = 98
  ' cal factor in percent

End If

See Also:
### Methods

| None |

### Properties

<table>
<thead>
<tr>
<th><strong>CalFactorSegments</strong></th>
<th>Collection for iterating through the segments of a power sensor cal factor table.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MinimumFrequency</strong></td>
<td>Minimum usable frequency (Hz) specified for this power sensor.</td>
</tr>
<tr>
<td><strong>MaximumFrequency</strong></td>
<td>Maximum usable frequency (Hz) specified for this power sensor.</td>
</tr>
<tr>
<td><strong>PowerMeterChannel</strong></td>
<td>Identifies which power sensor this object corresponds to (or which channel of the power meter the sensor is connected to).</td>
</tr>
<tr>
<td><strong>ReferenceCalFactor</strong></td>
<td>Reference cal factor (%) associated with this power sensor.</td>
</tr>
</tbody>
</table>

### IPowerSensor History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerSensor</td>
<td>2.0</td>
</tr>
</tbody>
</table>
PowerSensorCalFactorSegment Object

Description
Contains the properties describing a segment of a power sensor cal factor table.

Accessing the PowerSensorCalFactorSegment object
You can get a handle to one of these segments through CalFactorSegments.Item(n)

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim calFactSeg As CalFactorSegments
Set calFactSeg = app.SourcePowerCalibrator.PowerSensors(1).CalFactorSegments(1)
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

Methods
None

Properties

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CalFactor</td>
</tr>
<tr>
<td>SegmentNumber</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

IPowerSensorCalFactorSegment History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPowerSensorCalFactorSegment</td>
<td>2.0</td>
</tr>
</tbody>
</table>
PowerSensors Collection

Description
A collection object that provides a mechanism for iterating through the PowerSensor objects which are connected to the power meter. Each time this collection object is accessed, the power meter is queried to determine how many sensors are connected to it. The collection size and order of objects is then adjusted accordingly before the requested method or property operation is performed. The power meter is specified by using the PowerMeterGPIBAddress property of the SourcePowerCalibrator object.

Accessing the PowerSensors Collection

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim PwrSensors As PowerSensors
Set PwrSensors = app.SourcePowerCalibrator.PowerSensors
```

See Also:
- **PowerSensor Object**
- **Collections in the Analyzer**
- **The PNA Object Model**
- **Example Programs**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Use to get a handle to a PowerSensor object in the collection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Returns the number of objects in the collection.</td>
</tr>
<tr>
<td>Parent</td>
<td>Returns a handle to the Parent object (SourcePowerCalibrator) of this collection.</td>
</tr>
</tbody>
</table>
Preferences Object

Description
Sets the preferences for the behavior of several properties.

Accessing the Preferences object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim pref As Preferences
Set pref = app.Preferences
```

See Also:
- Citifile Define Data Saves
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

Methods
None

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>auxtrigger scopels global</td>
<td>IPreferences5</td>
<td>Sets the External Trigger OUT behavior to have either Global or Channel scope.</td>
</tr>
<tr>
<td>citicontents</td>
<td>IPreferences</td>
<td>Specifies the contents of subsequent citifile saves.</td>
</tr>
<tr>
<td>citiformat</td>
<td>IPreferences</td>
<td>Specifies the format of subsequent citifile saves.</td>
</tr>
<tr>
<td>enablesource unlevelled events</td>
<td>IPreferences6</td>
<td>Specifies whether or not to report Source Unleveled errors as system events.</td>
</tr>
<tr>
<td>offset receiver attenuator</td>
<td>IPreferences6</td>
<td>Mathematically offset the test port receiver.</td>
</tr>
<tr>
<td>offset source attenuator</td>
<td>IPreferences6</td>
<td>Mathematically offset the reference receiver.</td>
</tr>
<tr>
<td>port1 noise tuner switch presets to external</td>
<td>IPreferences8</td>
<td>Sets default setting for Noise Figure switch.</td>
</tr>
</tbody>
</table>
PowerOnDuringRetraceMode IPreferences4 Specify whether to turn RF power ON or OFF during a retrace for single-band frequency or segment sweeps ONLY.

PowerSweepRetracePowerMode IPreferences3 At the end of a power sweep, specifies whether to maintain source power at the start or stop power level.

PreferInternalTriggerOnChannelSingle IPreferences2 Sets the preference for chan.Single behavior.

PreferInternalTriggerOnUnguidedCal IPreferences2 Set the preference for the trigger behavior when performing an Unguided calibration.

RemoteCalStoragePreference IPreferences7 Specifies the default manner in which calibrations performed via SCPI or COM are to be stored.

SnPFormat IPreferences Specifies the format of subsequent .S1P, .S2P, .S3P file saves.

IPreferences History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPreferences</td>
<td>4.0</td>
</tr>
<tr>
<td>IPreferences2</td>
<td>6.0</td>
</tr>
<tr>
<td>IPreferences3</td>
<td>7.2</td>
</tr>
<tr>
<td>IPreferences4</td>
<td>6.04</td>
</tr>
<tr>
<td>IPreferences5</td>
<td>7.10</td>
</tr>
<tr>
<td>IPreferences6</td>
<td>7.20</td>
</tr>
<tr>
<td>IPreferences7</td>
<td>7.21</td>
</tr>
<tr>
<td>IPreferences8</td>
<td>8.0</td>
</tr>
</tbody>
</table>
PulseGenerator Object

Description
Contains the properties for configuring the five internal pulse generators in the PNA-X.

Accessing the PulseGenerator object

Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim pulse as PulseGenerator
Set pulse = chan.PulseGenerator

Each pulse generator is specified in the Pulse Generator properties. See below.

Pulse Definitions

- D = Delay; the time before each pulse begins
- W = Width; the time the pulse is ON
- P = Period; one complete pulse cycle
- W/P = Duty Cycle; the ratio of pulse ON/OFF

Important: If D + W is greater than P, then undefined PNA behavior results. There is NO error message or warning.

See Also:
- IF Path Block Diagram.
- PNA Automation Interfaces
- The PNA Object Model
- About PNA-X Pulse Capabilities
- Example Programs
## Methods

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

## Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay</td>
<td>IPulsedGenerator</td>
<td>Sets the pulse delay.</td>
</tr>
<tr>
<td>DelayIncrement</td>
<td>IPulsedGenerator</td>
<td>Sets the pulse delay increment.</td>
</tr>
<tr>
<td>Period</td>
<td>IPulsedGenerator</td>
<td>Sets the pulse-period (1/PRF) for ALL PNA-X internal pulse generators.</td>
</tr>
<tr>
<td>State</td>
<td>IPulsedGenerator</td>
<td>Turns the specified pulse generator ON and OFF.</td>
</tr>
<tr>
<td>Width</td>
<td>IPulsedGenerator</td>
<td>Sets the pulse width for the specified pulse generator.</td>
</tr>
</tbody>
</table>

### IPulseGenerator History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPulseGenerator</td>
<td>7.2</td>
</tr>
</tbody>
</table>

Last Modified:

1-Jan-2007  MX New topic
SCPIStringParser Object

Description
Provides the ability to send a SCPI command from within the COM command. The two commands differ in the following ways:

**Execute** - will not return an error unless the Execute command itself fails, which is unlikely. Otherwise, you are required to read the SCPI error queue for errors that were caused by the SCPI command. The Execute command operates with minimal interference between you, the programmer, and the SCPI parser. It does not presume how you want to handle errors: handle by ignore, handle by reading the status byte, etc. This command was defined because automation engines like VB throw runtime errors when a COM method returns a failed HRESULT.

**Parse** - parses the input command, and then reads the SCPI error queue until the queue is empty. If the queue contains errors, Parse returns a failed HRESULT (E_NA_BAD_SCPI_EXECUTE). It then creates an IErrorInfo object and bundles the error numbers and descriptions into the error object. This object is available so that you can detect the failed HRESULT and interrogate the errorInfo object for more details.

See an example of how to return error information when using the Parse method.

Accessing the ScpiStringParser object

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim SCPI As IScpiStringParser
Set SCPI = app.ScpiStringParser
```

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parse</td>
<td>ISCIStringParser</td>
<td>Provides the ability to send a SCPI command from within the COM command.</td>
</tr>
<tr>
<td>Execute</td>
<td>ISCIStringParser2</td>
<td>Does not convert scpi errors. Use :SYST:ERR?</td>
</tr>
</tbody>
</table>

Properties

None

History

835
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISCPIStringParser</td>
<td>1.0</td>
</tr>
<tr>
<td>ISCPIStringParser2</td>
<td>3.0</td>
</tr>
</tbody>
</table>
**Segment Object**

**Description**
Contains the methods and properties that affect a sweep segment.

**Note:** All of these properties are shared with at least one of the following objects: Channel, Cal Set, PowerSensorCalFactorSegment, or PowerLossSegment.

**Accessing a Segment object**
You can get a handle to a sweep segment through the segments collection.

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim segs As ISegments
Set segs = app.ActiveChannel.Segments

segs(2).NumberOfPoints = 30
```

**See Also:**
- PNA Automation Interfaces
- The PNA Object Model
- Segment Sweep
- Example Programs

**Methods**
None

**Properties**

<table>
<thead>
<tr>
<th>Properties</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>centerFrequency</td>
<td>ISegment</td>
<td>Sets or returns the center frequency of the segment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Channel Object</td>
</tr>
<tr>
<td>DwellTime</td>
<td>ISegment</td>
<td>Dwell time value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Channel Object</td>
</tr>
<tr>
<td>FrequencySpan</td>
<td>ISegment</td>
<td>Sets or returns the frequency span of the segment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Channel Object</td>
</tr>
<tr>
<td>IFBandwidth</td>
<td>ISegment</td>
<td>Sets or returns the IF Bandwidth of the segment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared with the Channel Object and Cal Set object.</td>
</tr>
</tbody>
</table>
**NumberOfPoints**  ISegment  Sets or returns the Number of Points of the segment.  
Shared with the Channel Object

**SegmentNumber**  ISegment  Returns the number of the current segment.

**StartFrequency**  ISegment  Sets or returns the start frequency of the segment.  
Shared with the Channel Object

**State**  ISegment  Turns On or OFF a segment.

**StopFrequency**  ISegment  Sets or returns the stop frequency of the segment.  
Shared with the Channel Object

**SweepTime**  ISegment2  Sets or returns the sweep time of the segment.  
Shared with the Channel Object

**TestPortPower**  ISegment  Sets or returns the RF power level of the segment.  
Shared with the Channel Object

**ISegment History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISegment</td>
<td>1.0</td>
</tr>
<tr>
<td>ISegment2</td>
<td>7.1</td>
</tr>
</tbody>
</table>

Last modified:

9/29/06  MQQ Added Sweep time
Segments Collection

Description
A collection object that provides a mechanism for iterating through the sweep segments of a channel. Sweep segments are a potentially faster method of sweeping the analyzer through only the frequencies of interest. Learn more about Segment Sweep.

Accessing the Segments collection
There are two paths to the Segments Collection:

1. From the Channel object
2. From the FOMRange object

Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim segs As ISegments
Set segs = app.ActiveChannel.Segments

or

Set segs = app.ActiveChannel.FOM.FOMRange(1).Segments

See Also:
- Segment Object
- Collections in the Analyzer
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>ISegments</td>
<td>Adds an item to either the Segments collection.</td>
</tr>
<tr>
<td>Item</td>
<td>ISegments</td>
<td>Use to get a handle to a segment in the collection.</td>
</tr>
<tr>
<td>Remove</td>
<td>ISegments</td>
<td>Removes an item from a collection of objects.</td>
</tr>
<tr>
<td>SetAllSegments</td>
<td>ISegments2</td>
<td>Uploads a segment table to the PNA.</td>
</tr>
</tbody>
</table>

Methods

See History

Description
**AllowArbitrarySegments**  
**ISegments3** Enables the setup of arbitrary segment sweep

**Count**  
**ISegments** Returns the number of items in a collection of objects.

**IF Bandwidth Option**  
**ISegments** Enables the IFBandwidth to be set on individual sweep segments.

**Parent**  
**ISegments** Returns a handle to the current naNetworkAnalyzer application.

**Source Power Option**  
**ISegments** Enables setting the Source Power for a segment.

**SweepTimeOption**  
**ISegments4** Enables the **Sweep time** or **Dwell time** to be set independently on sweep segments.

### ISegments History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISegments</td>
<td>1.0</td>
</tr>
<tr>
<td>ISegments2</td>
<td>3.5</td>
</tr>
<tr>
<td>ISegments3</td>
<td>4.2</td>
</tr>
<tr>
<td>ISegments4</td>
<td>7.1</td>
</tr>
</tbody>
</table>

---

Last modified:

- 8-Mar-2007   Modified access via fom
- 9/29/06     Added ISegments4
SignalProcessingModuleFour Object

Description
Contains the properties for configuring the DSP (digital filters) in the PNA-X.

[Image: IF Path Block diagram]

See the entire IF Path Block diagram.

Accessing the SignalProcessingModuleFour object

```vba
Dim app as AgilentPNA835x.Application
Dim chan as Channel
Set chan = app.ActiveChannel
Dim digFilter as SignalProcessingModuleFour
Set digFilter = chan.SignalProcessingModuleFour
```

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- About PNA-X Pulse Capabilities
- Example Programs

Methods

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADCCaptureMode</td>
<td>ISPM4</td>
<td>Sets ADC capture mode: auto or manual</td>
</tr>
<tr>
<td>FilterErrors</td>
<td>ISPM4</td>
<td>Returns errors with manual digital filter settings</td>
</tr>
<tr>
<td>FilterMode</td>
<td>ISPM4</td>
<td>Sets digital filter mode: auto or manual</td>
</tr>
<tr>
<td>Stage1Coefficients</td>
<td>ISPM4</td>
<td>Sets Stage1Coefficients</td>
</tr>
</tbody>
</table>
**Stage1Frequency**
- ISPM4
- Sets Stage1 NCO frequency

**Stage1MaximumCoefficient**
- ISPM4
- Returns the maximum value of any single stage1 coefficient.

**Stage1MaximumCoefficientCount**
- ISPM4
- Returns the maximum number of Stage1 coefficients.

**Stage1MaximumCoefficientSum**
- ISPM4
- Returns the maximum sum of all Stage1 coefficients.

**Stage1MinimumCoefficientCount**
- ISPM4
- Returns the minimum number of Stage1 coefficients.

**Stage2Coefficients**
- ISPM4
- Sets Stage2 Coefficients

**Stage2MaximumCoefficient**
- ISPM4
- Returns the maximum value of any single stage2 coefficient.

**Stage2MaximumCoefficientCount**
- ISPM4
- Returns the maximum number of Stage2 coefficients.

**Stage2MaximumCoefficientSum**
- ISPM4
- Returns the maximum sum of all Stage2 coefficients.

**Stage2MinimumCoefficientCount**
- ISPM4
- Returns the minimum number of Stage2 coefficients.

**Stage3FilterType**
- ISPM4
- Sets and returns stage3 filter type

**Stage3FilterTypes**
- ISPM4
- Returns the names of supported types of Stage3 filters.

**Stage3Parameter**
- ISPM4
- Sets and returns the parameter value of the current filter type.

**Stage3ParameterMaximum**
- ISPM4
- Returns maximum parameter value for the current filter type.

**Stage3ParameterMinimum**
- ISPM4
- Returns minimum parameter value for the current filter type.

**Stage3Parameters**
- ISPM4
- Returns the names of parameters for the current filter type.
### ISignalProcessingModuleFour History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISignalProcessingModuleFour</td>
<td>7.2</td>
</tr>
</tbody>
</table>

Last Modified:

- 5-Jan-2007  MX New topic
SMCType Object

Description
Contains the methods and properties to perform an Scalar Measurement Calibration for the Frequency Converter Application (option 083).

Accessing the SMCType object
See an example which creates and calibrates an SMC measurement.

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcquireStep</td>
<td>ISMCType</td>
<td>Acquire the measurement data for the specified step in the calibration process.</td>
</tr>
<tr>
<td>GenerateErrorTerms</td>
<td>ISMCType</td>
<td>Generates the error terms for the calibration.</td>
</tr>
<tr>
<td>GenerateSteps</td>
<td>ISMCType</td>
<td>Returns the number of steps required to complete the calibration.</td>
</tr>
<tr>
<td>GetStepDescription</td>
<td>ISMCType</td>
<td>Returns the description of the specified step calibration process.</td>
</tr>
<tr>
<td>Initialize</td>
<td>ISMCType</td>
<td>Begins a calibration.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoOrient</td>
<td>ISMCType Sets ECAL module automatic orientation ON or OFF.</td>
</tr>
<tr>
<td>CalibrationPort</td>
<td>ISMCType Sets or returns the calibration source port for the calibration.</td>
</tr>
<tr>
<td>CalKitType</td>
<td>ISMCType Sets and returns a calibration kit type for calibration.</td>
</tr>
<tr>
<td>CompatibleCalKits</td>
<td>ISMCType Returns a list of cal kits that are compatible with the connector type for the specified port.</td>
</tr>
</tbody>
</table>
**ConnectorType** ISMCType Sets or queries the connector type for the specified port.

**Do2PortEcal** ISMCType Specify ECAL or Mechanical calibration.

**EcalCharacterization** ISMCType Specifies the characterization data within an ECal module to be used for the calibration.

**EcalOrientation** ISMCType Specifies which port of the ECal module is connected to which port of the PNA when the AutoOrient property = False.

**NetworkFilename** ISMCType2 Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement.

**NetworkMode** ISMCType2 Embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement.

**OmitIsolation** ISMCType Sets and returns whether Isolation portion of the calibration will be performed or not.

**ThruCalMethod** ISMCType Sets and returns the method for performing the thru portion of the calibration.

**ValidConnectorTypes** ISMCType Returns a list of connector types for which there are calibration kits.

### ISMCType History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISMCType</td>
<td>3.5</td>
</tr>
<tr>
<td>ISMCType2</td>
<td>6.0</td>
</tr>
</tbody>
</table>
**SourcePowerCalibrator Object**

**Description**
This object is a child of the Application object and is a vehicle for performing source power calibrations.

**Accessing the SourcePowerCalibrator Object**

```vbnet
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim ispc As ISourcePowerCalibrator
Set ispc = app.SourcePowerCalibrator
```

**See Also:**
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

**Note:** Interface ISourcePowerCalibrator is abbreviated as ISPC in the following table.

(Bold Methods or Properties provide access to a child object)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbortPowerAcquisition</td>
<td>ISPC</td>
<td>Aborts a source power cal acquisition sweep that is currently in progress.</td>
</tr>
<tr>
<td>AcquirePowerReadings</td>
<td>ISPC</td>
<td>Superseded with AcquirePowerReadingsEx</td>
</tr>
<tr>
<td>AcquirePowerReadingsEx</td>
<td>ISPC4</td>
<td>Initiates a source power cal acquisition.</td>
</tr>
<tr>
<td>ApplyPowerCorrectionValues</td>
<td>ISPC</td>
<td>Superseded with ApplyPowerCorrectionValuesEx</td>
</tr>
<tr>
<td>ApplyPowerCorrectionValuesEx</td>
<td>ISPC5</td>
<td>Applies correction values after completing a source power cal acquisition sweep. Optionally perform a calibration of the reference receiver used in the source power cal.</td>
</tr>
<tr>
<td>CheckPower</td>
<td>ISPC2</td>
<td>Measures power at a specific frequency. Used to test power level before and/or after applying a source power calibration.</td>
</tr>
<tr>
<td>LaunchPowerMeterSettingsDialog</td>
<td>ISPC2</td>
<td>Launches the Power Meter Settings dialog on the PNA.</td>
</tr>
<tr>
<td>SetCalInfo</td>
<td>ISPC</td>
<td>Superseded with SetCalInfoEx Method</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalPower</td>
<td>ISPC</td>
<td>Specifies the power level that is expected at the desired reference plane.</td>
</tr>
<tr>
<td>IterationsTolerance</td>
<td>ISPC3</td>
<td>Sets the maximum desired deviation from the sum of the test port power and the offset value.</td>
</tr>
<tr>
<td>MaximumIterationsPerPoint</td>
<td>ISPC3</td>
<td>Specifies maximum number of readings to take at each data point for iterating the source power.</td>
</tr>
<tr>
<td>PowerAcquisitionDevice</td>
<td>ISPC2</td>
<td>Specifies the power sensor channel (A or B) that is currently selected for use at a specific frequency.</td>
</tr>
<tr>
<td>PowerLossSegments (collection)</td>
<td>ISPC2</td>
<td>Collection for iterating through the segments of the power loss table used in source power calibration.</td>
</tr>
<tr>
<td>PowerMeterGPIBAddress</td>
<td>ISPC</td>
<td>Specifies the GPIB address of the power meter.</td>
</tr>
<tr>
<td>PowerMeterInterfaces</td>
<td>ISPC6</td>
<td>Collection for getting a handle to the available power meters.</td>
</tr>
<tr>
<td>PowerSensors (collection)</td>
<td>ISPC2</td>
<td>Collection for iterating through the PowerSensor objects which are connected to the power meter for a source power cal.</td>
</tr>
<tr>
<td>ReadingsPerPoint</td>
<td>ISPC</td>
<td>Specifies the maximum power readings for power meter settling.</td>
</tr>
<tr>
<td>ReadingsTolerance</td>
<td>ISPC3</td>
<td>Power meter settling tolerance value.</td>
</tr>
<tr>
<td>USBPowerMeterCatalog</td>
<td>ISPC6</td>
<td>Returns a list of USB power meters that are connected to the PNA.</td>
</tr>
<tr>
<td>UsePowerLossSegments</td>
<td>ISPC</td>
<td>Specifies if subsequent calls to the AcquirePowerReadings method will make use of the loss table (PowerLossSegments).</td>
</tr>
</tbody>
</table>
UsePowerSensorFrequencyLimits ISPC Specifies if subsequent calls to the AcquirePowerReadings method will make use of power sensor frequency checking capability.

**ISourcePowerCalibrator History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISourcePowerCalibrator</td>
<td>2.0</td>
</tr>
<tr>
<td>ISourcePowerCalibrator2</td>
<td>3.5</td>
</tr>
<tr>
<td>ISourcePowerCalibrator3</td>
<td>4.0</td>
</tr>
<tr>
<td>ISourcePowerCalibrator4</td>
<td>6.2</td>
</tr>
<tr>
<td>ISourcePowerCalibrator5</td>
<td>7.2</td>
</tr>
<tr>
<td>ISourcePowerCalibrator6</td>
<td>7.5</td>
</tr>
</tbody>
</table>
TestsetControl Object

Description
A TestsetControl object is used to control one of the supported test sets. Only one external test set can be controlled by the PNA at any time. The Testset Control object appears as an item in the ExternalTestsets collection, which in turn is a property of the main application object.

If the specified test set is not connected to the PNA or is not ON, then setting Enabled = True will return an error. All other properties can be set even if the test set is not connected.

Note: The ONLY way to load a test set configuration file is by sending the testsets.Add method. There is no method to query the test set type. See an example program.

Accessing a TestsetControl object
The ExternalTestsets collection is a property of the main Application Object. You can obtain a handle to a testset object by specifying an item in the collection.

Visual Basic Example

```vba
Dim pna
Dim testsets As ExternalTestsets
Dim tset1 As TestsetControl
Set pna = CreateObject("AgilentPNA835x.Application")
Set testsets = pna.ExternalTestsets
Set tset1 = testsets(1)
' make COM calls on tset1 object
End Sub
```

See Also:

- E5091A Testset Object
- About External Testset Control
- ExternalTestset Control Example
- ExternalTestsets Collection
- The PNA Object Model
## Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(See history)</td>
<td></td>
</tr>
</tbody>
</table>

None

## Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ControlLines</td>
<td>IExternalTestset</td>
<td>Sets the control lines of the specified Test set.</td>
</tr>
<tr>
<td>Enabled</td>
<td>IExternalTestset</td>
<td>Enables and disables (ON/OFF) the port mapping and control line output of the specified test set.</td>
</tr>
<tr>
<td>ID</td>
<td>IExternalTestset</td>
<td>Returns the test set ID number.</td>
</tr>
<tr>
<td>Label</td>
<td>IExternalTestset</td>
<td>Returns the label on a given channel for the specified test set.</td>
</tr>
<tr>
<td>NumberOfPorts</td>
<td>IExternalTestset</td>
<td>Reads the number of ports that are on the specified test set.</td>
</tr>
<tr>
<td>OutputPorts</td>
<td>IExternalTestset</td>
<td>Sets or returns the port mappings for ALL ports.</td>
</tr>
<tr>
<td>PortCatalog</td>
<td>IExternalTestset</td>
<td>Returns the selections available for a given logical port.</td>
</tr>
<tr>
<td>SelectPort</td>
<td>IExternalTestset</td>
<td>Sets and returns the logical port value.</td>
</tr>
<tr>
<td>ShowProperties</td>
<td>IExternalTestset</td>
<td>Turns status bar display of test set properties on or off.</td>
</tr>
<tr>
<td>Type</td>
<td>IExternalTestset</td>
<td>Returns the test set model.</td>
</tr>
</tbody>
</table>

### ExternalTestset History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IExternalTestset</td>
<td>6.0</td>
</tr>
<tr>
<td>IExternalTestset</td>
<td>6.0</td>
</tr>
</tbody>
</table>
Trace Object

Description
The Trace object controls how the measurement data is displayed. You can control scale, reference position, and value from the Trace Object.

Accessing a Trace object
There are several ways to get a handle to a trace.

```
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim trace As Trace

Then you can do any of the following:

Set trace = app.NAWindows(1).traces(1)
set trace = app.NAWindows.item(1).ActiveTrace
set trace = app.ActiveNAWindow.traces.item(1)
set trace = app.ActiveNAWindow.ActiveTrace
Set trace = app.Measurements(1).trace
Set trace = app.ActiveMeasurement.trace
```

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- Traces, Channels, and Windows on the PNA
- Example Programs
### Methods

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoscale</td>
</tr>
<tr>
<td>Autoscales the trace or all of the traces in the selected window.</td>
</tr>
<tr>
<td>Shared with the NAWindow Object</td>
</tr>
</tbody>
</table>

### Property

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Sets or returns the trace name</td>
</tr>
</tbody>
</table>

| ReferencePosition |
| Sets or returns the Reference Position of the active trace. |

| ReferenceValue |
| Sets or returns the value of the Y-axis Reference Level of the active trace. |

| YScale |
| Sets or returns the Y-axis Per-Division value of the active trace. |

### ITrace History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITrace</td>
<td>1.0</td>
</tr>
</tbody>
</table>
**Traces Collection**

**Description**
Child of the Application Object. A collection that provides a mechanism for getting a handle to a trace or iterating through the traces in a window.

**Accessing the Traces collection**
Get a handle to the traces collection through the NaWindows collection. The following example sets the variable `trcs` to the collection of traces in window 1 of the NaWindows collection.

```vbscript
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)
Dim trcs As traces
Set trcs = app.NAWindows(1).traces
```

**See Also:**
- Trace Object
- Collections in the Analyzer
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Item</strong></td>
<td>Use to get a handle to a trace</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Count</strong></td>
<td>Returns the number of traces in the collection.</td>
</tr>
<tr>
<td><strong>Parent</strong></td>
<td>Returns a handle to the current Application.</td>
</tr>
</tbody>
</table>
### Transform Object

**Description**

Contains the methods and properties that control Time Domain transforms.

**Accessing the Transform Object**

```vba
Dim app As AgilentPNA835x.Application
Set app = CreateObject("AgilentPNA835x.Application", <analyzerName>)

Dim trans As Transform
Set trans = app.ActiveMeasurement.Transform
```

**See Also:**

- PNA Automation Interfaces
- The PNA Object Model
- Time Domain Topics
- Example Programs

**Note:** Sweep Type must be set to Linear before setting Time Domain Transform (state) ON.

<table>
<thead>
<tr>
<th>Method</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetFrequencyLowPass</td>
<td>ITransform</td>
<td>Sets low frequencies for low pass.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center</td>
<td>Sets or returns the Center time. Shared with the Gating Object</td>
</tr>
<tr>
<td>CoupledParameters</td>
<td>Select Transform parameters to couple</td>
</tr>
<tr>
<td>DistanceMarkerMode</td>
<td>Sets the measurement type in order to determine the correct marker distance.</td>
</tr>
<tr>
<td>DistanceMarkerUnit</td>
<td>Sets the unit of measure for the display of marker distance values.</td>
</tr>
<tr>
<td>ImpulseWidth</td>
<td>Sets or returns the Impulse Width of Time Domain transform windows.</td>
</tr>
<tr>
<td>KaiserBeta</td>
<td>Sets or returns the Kaiser Beta of Time Domain transform windows.</td>
</tr>
<tr>
<td>Mode</td>
<td>Sets the type of transform.</td>
</tr>
</tbody>
</table>
Span ITransform Sets or returns the Span time.
Shared with the Gating Object

Start ITransform Sets or returns the Start time.
Shared with the Gating Object

State ITransform Turns an Object ON and OFF.

StepRiseTime ITransform Sets or returns the Rise time of the stimulus in Low Pass Step Mode.

Stop ITransform Sets or returns the Stop time.
Shared with the Gating Object

**ITransform History**

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITransform</td>
<td>1.0</td>
</tr>
<tr>
<td>ITransform2</td>
<td>4.2</td>
</tr>
</tbody>
</table>
TriggerSetup Object

Description
These properties setup Global triggering that effects the entire PNA application.

Accessing the TriggerSetup object

```vbscript
Dim app as AgilentPNA835x.Application
Dim trigSetup as ITriggerSetup
Set trigSetup = app.TriggerSetup
```

See Also:

- PNA Automation Interfaces
- The PNA Object Model
- Triggering in the PNA
- Example Programs

### Methods

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>See History (below)</strong></td>
<td></td>
</tr>
</tbody>
</table>

None

### Properties

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AcceptTriggerBeforeArmed</strong> ITriggerSetup2 Allows a trigger signal to be remembered and then used when the PNA becomes armed (ready to be triggered).</td>
</tr>
<tr>
<td><strong>ExternalTriggerConnectionBehavior</strong> ITriggerSetup Configures the external triggering signal for the PNA</td>
</tr>
<tr>
<td><strong>Scope</strong> ITriggerSetup Determines whether a trigger signal affects a single channel or all channels in the PNA.</td>
</tr>
<tr>
<td><strong>Source</strong> ITriggerSetup Sets or returns the source of triggering in the PNA.</td>
</tr>
<tr>
<td><strong>TriggerOutputEnabled</strong> ITriggerSetup2 Enables the PNA to send trigger signals out the rear-panel TRIGGER OUT connector.</td>
</tr>
</tbody>
</table>
### ITriggerSetup History

<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITriggerSetup</td>
<td>4.0</td>
</tr>
<tr>
<td>ITriggerSetup2</td>
<td>4.2</td>
</tr>
</tbody>
</table>
VMC Type Object

Description
Contains the methods and properties to perform a Vector Measurement Calibration for the Frequency Converter Application (option 083).

Accessing the VMCTYPE object
See an example which creates and calibrates a VMC measurement.

See Also:
- PNA Automation Interfaces
- The PNA Object Model
- Example Programs

<table>
<thead>
<tr>
<th>Methods</th>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcquireStep</td>
<td>IVMCTYPE</td>
<td>Acquire the measurement data for the specified step in the calibration process.</td>
</tr>
<tr>
<td>GenerateErrorTerms</td>
<td>IVMCTYPE</td>
<td>Generates the error terms for the calibration.</td>
</tr>
<tr>
<td>GenerateSteps</td>
<td>IVMCTYPE</td>
<td>Returns the number of steps required to complete the calibration.</td>
</tr>
<tr>
<td>GetStepDescription</td>
<td>IVMCTYPE</td>
<td>Returns the description of the specified step in the calibration process.</td>
</tr>
<tr>
<td>Initialize</td>
<td>IVMCTYPE</td>
<td>Begins a calibration.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoOrient</td>
<td>IVMCTYPE Sets ECAL module automatic orientation ON or OFF.</td>
</tr>
<tr>
<td>CalKitType</td>
<td>IVMCTYPE Sets and returns a calibration kit type for calibration.</td>
</tr>
<tr>
<td>CharacterizeMixerOnly</td>
<td>IVMCTYPE Sets and returns whether to perform a mixer characterization ONLY or full 2-port calibration.</td>
</tr>
<tr>
<td>CharFileName</td>
<td>IVMCTYPE Specifies the .S2P mixer characterization file name.</td>
</tr>
</tbody>
</table>
CharMixerReverse  IVMCType2  Specifies the direction in which to characterize the calibration mixer.

CompatibleCalKits  IVMCType  Returns a list of cal kits that are compatible with the connector type for the specified port.

ConnectorType  IVMCType  Sets or queries the connector type for the specified port.

Do1PortEcal  IVMCType  Specify ECAL or Mechanical calibration for the mixer characterization portion of a VMC calibration.

Do2PortEcal  IVMCType  Specify ECAL or Mechanical calibration for the 2-port calibration portion of a VMC calibration.

EcalCharacterization  IVMCType  Specifies the characterization data within an ECal module to be used for the calibration.

EcalOrientation1Port  IVMCType  For Mixer Characterization ONLY - Specifies which port of the ECal module is connected to which port of the PNA.

EcalOrientation2Port  IVMCType  For full 2-port VMC cal - Specifies which port of the ECal module is connected to which port of the PNA.

LoadCharFromFile  IVMCType  Specifies and loads a mixer characterization (S2P) file.

NetworkFilename  IVMCType3  Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement.

NetworkMode  IVMCType3  Embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement.

OmitIsolation  IVMCType  Sets and returns whether Isolation portion of the calibration will be performed or not.

ThruCalMethod  IVMCType  Sets and returns the method for performing the thru portion of the calibration.

ValidConnectorTypes  IVMCType  Returns a list of connector types for which there are calibration kits.

IVMCTYPE History
<table>
<thead>
<tr>
<th>Interface</th>
<th>Introduced with PNA Rev:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IVMCTYPE</td>
<td>3.5</td>
</tr>
<tr>
<td>IVMCTYPE2</td>
<td>3.53</td>
</tr>
<tr>
<td>IVMCTYPE3</td>
<td>6.0</td>
</tr>
</tbody>
</table>
AcceptTriggerBeforeArmed Property

Description
Determined what happens to an EDGE trigger signal if it occurs before the PNA is ready to be triggered. (LEVEL trigger signals are always ignored.) For more information, see External triggering.

VB Syntax
trigsetup.AcceptTriggerBeforeArmed = boolean

Variable (Type) - Description
trigsetup A TriggerSetup2 (object)
boolean Choose from:
False - A trigger signal is ignored if it occurs before the PNA is ready to be triggered.
True - A trigger signal is remembered and then used when the PNA becomes armed (ready to be triggered). The PNA remembers only one trigger signal.

Return Type
Boolean

Default
False

Examples
trigsetup.AcceptTriggerBeforeArmed = True 'Write
atba = trigsetup.AcceptTriggerBeforeArmed 'Read

C++ Syntax
HRESULT get_AcceptTriggerBeforeArmed( BOOL *pVal);
HRESULT put_AcceptTriggerBeforeArmed( BOOL newVal);

Interface
ITriggerSetup2
AcquisitionDirection Property

Description  Specifies the direction of each part of a 2-port calibration.

VB Syntax  
```
cal.AcquisitionDirection = value
```

Variable  
**cal**  A Calibrator (object)

**value**  (enum NADirection) - Choose from:
- 0 - naForward - measures the forward direction
- 1 - naReverse - measures the reverse direction

Return Type  Long Integer

Default  naForward

Examples  
```
cal.AcquisitionDirection = naForward
```

C++ Syntax  
```
HRESULT AcquisitionDirection(tagNADirection dir);
```

Interface  ICalibrator
### About Compression Mode

**AcquisitionMode Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the method by which gain compression data is acquired.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>gca.AcquisitionMode = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>gca</code></td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(NAGCAAcquisitionMode) Choose from:</td>
</tr>
<tr>
<td></td>
<td>• <code>naSmartSweep (0)</code> Iterate quickly to find compression point</td>
</tr>
<tr>
<td></td>
<td>• <code>naSweepPowerAtEachFreq2D (1)</code> Sweep power at each frequency</td>
</tr>
<tr>
<td></td>
<td>• <code>naSweepFreqAtEachPower2D (2)</code> Sweep frequency at each power level</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><code>naSmartSweep</code></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>gca.AcquisitionMode = naSmartSweep 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>acqMode = gca.AcquisitionMode 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_AcquisitionMode(tagNAGCAAcquisitionMode* mode)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_AcquisitionMode(tagNAGCAAcquisitionMode mode)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>

Last Modified:

11-Sep-2007   MX New topic
### ActiveCalKit Property

**Description**  Returns a handle to the Active CalKit object. You can either (1) use the handle directly to access CalKit properties and methods, or (2) set a variable to the CalKit object. The variable retains a handle to the original object if another CalKit becomes active.

**VB Syntax**

1) `app.ActiveCalKit.<setting>`  
   or  
2) `Set cKit = app.ActiveCalKit`

**Variable (Type) - Description**

- **app**: An `Application` (object)
- **<setting>**: A CalKit property (or method) and arguments
- **cKit**: (object) - A CalKit object

**Return Type**  CalKit object

**Default**  None

**Examples**

```
Public cKit as calKit
Set cKit = app.ActiveCalKit 'read
```

**C++ Syntax**  `HRESULT get_ActiveCalKit (ICalkit * kit)`

**Interface**  `IApplication`
### ActiveChannel Property

**Description**
Returns a handle to the Active Channel object. You can either (1) use the handle directly to access channel properties and methods, or (2) set a variable to the channel object. The variable retains a handle to the original channel if another channel becomes active.

**VB Syntax**
1. `app.ActiveChannel.<setting>`
   or
2. `Set chan = app.ActiveChannel`

**Variable**
- **Type**
  - `chan` A Channel (object)
  - `app` An Application (object)

- `<setting>` A channel property (or method) and arguments

**Return Type**
Channel object

**Default**
Not applicable

**Examples**
1. `app.ActiveChannel.Averaging = 1`
2. `Public chan as Channel
   Set chan = app.ActiveChannel`

**C++ Syntax**
`HRESULT get_ActiveChannel(IChannel* *pVal)`

**Interface**
IApplcation
ActiveMarker Property

**Description**
Returns a handle to the Active Marker object. You can either (1) use the handle directly to access Marker properties and methods, or (2) set a variable to the Marker object. The variable retains a handle to the original object if another Marker becomes active.

**VB Syntax**
1) `meas.ActiveMarker.<setting>
   or
2) Set `mark = meas.ActiveMarker`

**Variable**
- `meas` (object) - An Measurement object
- `<setting>` A marker property (or method) and arguments
- `mark` (object) - A marker object

**Return Type**
marker object

**Default**
None

**Examples**
```vbnet
Public mark as marker
Set mark = meas.ActiveMarker
```

**C++ Syntax**
```cpp
HRESULT get_ActiveMarker(IMarker** marker)
```

**Interface**
IMeasurement
**ActiveMeasurement Property**

**Description**
Returns a handle to the Active Measurement object. You can either (1) use the handle directly to access measurement properties and methods, or (2) set a variable to the measurement object. The variable retains a handle to the original measurement.

**VB Syntax**
1) `app.ActiveMeasurement.<setting>`
   or
2) Set `meas = app.ActiveMeasurement`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>&lt;setting&gt;</td>
<td>A measurement property (or method) and arguments</td>
</tr>
</tbody>
</table>

**Return Type**
Measurement object

**Default**
None

**Examples**
1) `app.ActiveMeasurement.Averaging = 1`
2) `Public meas as Measurement
   Set meas = app.ActiveMeasurement`

**C++ Syntax**
`HRESULT get_ActiveMeasurement(IMeasurement **ppMeas)`

**Interface**
IApplication
ActiveNAWindow Property

**Description**
Returns a handle to the Active Window object. You can either (1) use the handle directly to access window properties and methods, or (2) set a variable to the window object. The variable retains a handle to the original window if another window becomes active.

**VB Syntax**
1) `app.ActiveNAWindow.<setting>`
   or
2) Set `win = app.ActiveNAWindow`

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>win</code></td>
<td>A NAWindow (object)</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
</tbody>
</table>

`<setting>` A NAWindow property (or method) and arguments

**Return Type**
A NAWindow object

**Default**
Not applicable

**Examples**
Public `win as NAWindow`
Set `win = app.ActiveWindow`

**C++ Syntax**
`HRESULT get_ActiveNAWindow(INAWindow **ppWindow)`

**Interface**
IApplication
ActiveXAxisRange Property

**Description**
Sets or returns the swept parameter to display on the X-axis for the selected FCA measurement.

**VB Syntax**

```vbnet
mixer.ActiveXAxisRange = value
```

**Variable (Type) - Description**

- **mixer**
  A Mixer (object)

- **value**
  (Enum as MixerStimulusRange) - Parameter to display on the X-axis. Choose from:
  
  - 0 - mixINPUT - Input frequency span
  - 1 - mixOUTPUT - Output frequency span
  - 2 - mixLO_1 - First LO frequency span
  - 3 - mixLO_2 - Second LO frequency span

**Return Type**
Enum

**Default**
OUTPUT

**Examples**

```vbnet
mixer.ActiveXAxisRange = 1 'Write
variable = mixer.ActiveXAxisRange 'Read
```

**C++ Syntax**

```cpp
HRESULT get_ActiveXAxisRange(tagMixerStimulusRange *Val)
HRESULT put_ActiveXAxisRange(tagMixerStimulusRange newVal)
```

**Interface**
IMixer3
ADCCaptureMode Property

Description
Sets and returns the ADC capture mode modeled as a 2-pole switch in the diagram on the SignalProcessingModuleFour page. The switch either bypasses or routes the IF through the 3-stage digital filter.

VB Syntax

```
spm4.ADCCaptureMode = value
```

Variable (Type) - Description

Value: 

- **spm4** - A SignalProcessingModuleFour (object)
- **value** - (Enum as NAStates) Capture mode.
  - **naOFF (0)** - The digital filters are used to process IF information. The filters can be configured automatically or manually using FilterMode Property.
  - **naON (1)** - The digital filters are bypassed and the raw ADC readings are taken directly. A maximum of 4096 data points per sweep can be acquired.

Return Type
Enum

Default
OFF

Examples

```
spm4.ADCCaptureMode = 0 'Write
mode = spm4.ADCCaptureMode 'Read
```

C++ Syntax

```
HRESULT get_ADCCaptureMode(tagNAStates* pCaptureMode);
HRESULT put_ADCCaptureMode(tagNAStates pCaptureMode);
```

Interface
ISignalProcessingModuleFour

Last Modified:

18-Jun-2007   MX New topic
ALCLevelingMode Property

**Description**
Sets and returns the ALC mode for the specified channel and port. Use GetSupportedALCModes to return a list of valid ALC modes for the PNA.

Learn more about ALC mode.

**VB Syntax**

```vbnet
chan.ALCLevelingMode (sourcePort) = value
```

**Variable**

**Type** - Description

- **chan** *(object)* - A Channel object
- **sourcePort** *(long integer)* - The source port for which to make this setting. If ports are remapped, specify the logical port number.
  Use GetPortNumber to return the port number of a source that only has a string name, such as an External Source.
- **value** *(enum as naALCLevelingMode)* - Choose from:
  - 0 naALCInternal
  - 1 naALCEXternal (E835x Only)
  - 2 naALCOpenLoop (PNA-X only)
  - 3 naALCIF (For future use)

**Return Type**
Enum

**Default**
naALCInternal

**Examples**

```vbnet
chan.ALCLevelingMode(1) = 'Write
state = chan.ALCLevelingMode(4) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_ALCLevelingMode(long port, tagNAALCLevelingMode* pVal);
HRESULT put_ALCLevelingMode(long port,tagNAALCLevelingMode newVal);
```

**Interface**
IChannel9

---

Last modified:

30-Apr-2007 Edited for src strings

10/18/06 MX New topic
### ActiveTrace Property

**Description**
Returns a handle to the Active Trace object. You can either (1) use the handle directly to access trace properties and methods, or (2) set a variable to the trace object. The variable retains a handle to the original trace if another trace becomes active.

**VB Syntax**
1) `win.ActiveTrace.<setting>`  
2) `Set trce = win.ActiveTrace`

**Variable**
- **(Type)** - **Description**
  - `trce` A Trace *(object)*
  - `win` An NAWindow *(object)*
  - `<setting>` A trace property (or method) and arguments

**Return Type**
An NAWindow object

**Default**
None

**Examples**
1) `win.ActiveTrace.Autoscale`
2) `Public trce as Trace  
   Set trce = Application.ActiveNAWindow.ActiveTrace`

**C++ Syntax**
`HRESULT get_ActiveTrace(ITrace* *pVal)`

**Interface**
INAWindow
AllowArbitrarySegments Property

Description
Enables you to setup a segment sweep with arbitrary frequencies. The start and stop frequencies of each segment can overlap other segments. Also, each segment can have a start frequency that is greater than its stop frequency which causes a reverse sweep over that segment. Learn more about Arbitrary Segment Sweep.

VB Syntax
```
segs.AllowArbitrarySegments = value
```

Variable (Type) - Description

<table>
<thead>
<tr>
<th>variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>segs</td>
<td>object</td>
<td>A Segments collection</td>
</tr>
<tr>
<td>value</td>
<td>boolean</td>
<td>(optional) True - Allows the setup of arbitrary segment sweep. False - Prevents the setup of arbitrary segment sweep.</td>
</tr>
</tbody>
</table>

Return Type
Boolean

Default
False

Examples
```
segs.AllowArbitrarySegments = True 'Write
AllowArbSegs = AllowArbitrarySegments 'Read
```

C++ Syntax
```
HRESULT get_AllowArbitrarySegments(VARIANT_BOOL *pVal)
HRESULT put_AllowArbitrarySegments(VARIANT_BOOL newVal)
```

Interface
ISegments3
AlternateSweep Property

Description
Sets sweeps to either alternate or chopped.

VB Syntax
object.AlternateSweep = value

Variable (Type) - Description

object Channel (object)
or
CalSet (object) - Read-only property

value (boolean) - Choose either:
False - Sweep mode set to Chopped - reflection and transmission are measured on the same sweep.
True - Sweep mode set to Alternate - reflection and transmission measured on separate sweeps. Improves Mixer bounce and Isolation measurements. Increases cycle time.

Return Type
boolean

Default
False (0)

Examples
chan.AlternateSweep = True 'Write
altSwp = chan.AlternateSweep 'Read

C++ Syntax
HRESULT AlternateSweep(VARIANT_BOOL *pVal)
HRESULT AlternateSweep(VARIANT_BOOL newVal)

Interface
IChannel
ICalSet3
## AmbientTemperature Property

**Description**
Sets and returns the temperature at which the current noise measurement is occurring. [Learn more.](#)

### VB Syntax

```vbnet
oxnoise.AmbientTemperature = value```

### Variable (Type) - Description
- **noiseCal**: A `NoiseCal` (object)
- **value**: (double) Ambient temperature in Kelvin.

### Return Type
Double

### Default
295

### Examples

```vbnet
noise.AmbientTemperature = 289 'Write

temp = noise.AmbientTemperature 'Read```

### C++ Syntax

```
HRESULT get_AmbientTemperature(Double* pValue)
HRESULT put_AmbientTemperature(Double pNewValue)
```

### Interface
INoiseCal

Last Modified:

| 6-Sep-2007 | MX New topic |
## Application Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the name of the Analyzer making measurements on the channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.Application</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>chan</code></td>
<td>A Channel (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>object</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>rfna = chan.Application</code> 'returns the Analyzer name'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Application(IApplication** Application)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel</td>
</tr>
</tbody>
</table>
### ArrangeWindows Property

**Description**
Sets the arrangement of all the windows. Overlay, Stack2, Split3 and Quad4 will create windows.

To control the state of one window, use `app.WindowState`.

**VB Syntax**
```
app.ArrangeWindows = value
```

**Variable**
- **Type**
- **Description**
  - `app` An `Application` (object)
  - `value` (enum NAWindowModes) - Choose from:
    - 0 - `naTile`
    - 1 - `naCascade`
    - 2 - `naOverlay`
    - 3 - `naStack2`
    - 4 - `naSplit3`
    - 5 - `naQuad4`

**Return Type**
Not Applicable

**Default**
`naTile`

**Examples**
```
app.ArrangeWindow = naTile    'Write
```

**C++ Syntax**
```
HRESULT put_ArrangeWindows(tagNAWindowModes newVal)
```

**Interface**
`IApplication`
### AttenuatorMode Property

**Description**
Sets or returns the mode of operation of the attenuator control for the specified port number. This command is automatically set to Manual when an Attenuator value is set.

**VB Syntax**
```
object.AttenuatorMode(portNum) = value
```

**Variable (Type) - Description**

| **object** | **Channel** (object) |
| **portNum** | (long) - Port number of attenuator control to be changed. |
| **value** | (enum NAModes) - Choose from: |

- **0 - naAuto** - Attenuator control set to automatic. The analyzer will set the attenuator control appropriately to deliver the specified power at the source.

**Return Type**
NAModes

**Default**
0 - Auto

**Examples**
```
chan.AttenuatorMode(1) = naAuto 'Write

attn = chan.AttenuatorMode(1) 'Read
```

**C++ Syntax**
```
HRESULT get_AttenuatorMode(long port, tagNAModes* pVal)
HRESULT put_AttenuatorMode(long port, tagNAModes newVal)
```

**Interface**
IChannel
ICalSet3
Attenuator Property

Description
Sets or returns the value of the source attenuator for the specified port number. Sending this command automatically sets AttenuatorMode to Manual.

VB Syntax
object. Attenuator(portNum) = value

Variable
(Type) - Description
object Channel (object)

or
CalSet (object) - Read-only property

portNum (long integer) - Port number of source attenuator to be changed.

value (double) - Attenuation value. The range of settable values depends on the PNA model. To determine the valid settings, do one of the following:

- See PNA models and options to see the range and step size for each model / option.
- To determine the maximum attenuator value use MaximumSourceStepAttenuator and . However, this will not tell you the attenuation step size.

If an invalid attenuation setting is entered, the PNA will select the next lower valid value. For example, if 19 is entered, then for an E8361A, 10 dB attenuation will be selected.

Return Type
Double

Default
20 dB

Examples
chan.Attenuator(1) = 20 'Write

attn = chan.Attenuator(cnum) 'Read

C++ Syntax
HRESULT get_Attenuator(long port, double *pVal)
HRESULT put_Attenuator(long port, double newVal)

Interface
IChannel
ICalSet3

Last Modified:

25-Oct-2007   Edit value text
30-Apr-2007   Minor edits
**AutoIFBWAdjustment Property**

**Description**  
Set and read auto IFBW adjustment ON | OFF state for Gain Compression measurements.

**VB Syntax**  
\[ gca.AutoIFBW Adjustment = value \]

**Variable**  
(Type) - Description

- **gca**  
  A [GainCompression](#) (object)

- **value**  
  (Boolean) - Auto IFBW adjustment state. Choose from:
  - False - Sets auto IFBW adjustment OFF
  - True - Sets auto IFBW adjustment ON

**Return Type**  
Boolean

**Default**  
ON

**Examples**

- `gca.AutoIFBWAdjustment = True` 'Write

- `aifbw = gca.AutoIFBWAdjustment` 'Read

**C++ Syntax**

- HRESULT get_AutoIFBWAdjustment(VARIANT_BOOL* bState)
- HRESULT put_AutoIFBWAdjustment(VARIANT_BOOL bState)

**Interface**  
IGainCompression

---

_Last Modified:_  
8-Nov-2007    MX New topic
AutoOrient Property

Description
Sets ECAL module automatic orientation ON or OFF.

VB Syntax
`obj.AutoOrient = bool`

Variable (Type) - Description
- `obj` (object)
  - `SMCType` or `VMCTYPE`
- `bool` (Boolean)
  - `True` - Set AutoOrientation ON
  - `False` - Set AutoOrientation OFF

Return Type
Boolean

Default
True

Examples
`Smc.AutoOrient = True`

C++ Syntax
- `HRESULT put_AutoOrient(VARIANT_BOOL bAutoOrient);`
- `HRESULT get_AutoOrient(VARIANT_BOOL *bAutoOrient);`

Interface
- `SMCType`
- `VMCTYPE`
About Auto Port Extensions

AutoPortExtConfig Property

**Description**
Sets the frequency span that is used to calculate Automatic Port Extension. Learn more about calculating Automatic Port Extension.

**VB Syntax**
```
fixture.AutoPortExtConfig = value
```

**Variable**
**Type** - Description
- **fixture** A Fixturing (object)
- **value** (ENUM as NAAutoPortExtConfig)
  - 0 naAPEC_CSPN - Use current span.
  - 1 naAPEC_AMKR - Use active marker frequency.
  - 2 naAPEC_USPN - Use custom user span. Use AutoPortExtSearchStart Property and AutoPortExtSearchStop Property to specify start and stop frequency.

**Return Type**
ENUM

**Default**
0 naAPEC_CSPN

**Examples**
```
fixture.AutoPortExtConfig = naAPEC_AMKR
value = fixture.AutoPortExtConfig 'Read
```

**C++ Syntax**
```
HRESULT get_AutoPortExtConfig(tagNAAutoPortExtConfig *pVal);
HRESULT put_AutoPortExtConfig(tagNAAutoPortExtConfig Val);
```

**Interface**
IFixturing2
AutoPortExtDCOffset Property

**Description**
Specifies whether or not to include DC Offset as part of automatic port extension. Learn more about [Automatic DC Offset](#). Only allowed when **AutoPortExtLoss Property** is set to ON.

**VB Syntax**
```
fixture.AutoPortExtDCOffset = bool
```

**Variable**
- **(Type)**: Description
  - `fixture` A **Fixturing** (object)
  - `bool` **True** - Includes DC Offset correction.
  - **False** - Does NOT include DC Offset correction.

**Return Type**
Boolean

**Default**
False

**Examples**
```
fixture.AutoPortExtDCOffset = True
value = fixture.AutoPortExtDCOffset 'Read
```

**C++ Syntax**
```
HRESULT get_AutoPortExtDCOffset(VARIANT_BOOL *pState);
HRESULT put_AutoPortExtDCOffset(VARIANT_BOOL bState);
```

**Interface**
IFixturing2
**AutoPortExtLoss Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies whether or not to include loss correction as part of automatic port extension. <a href="#">Learn more about Loss Compensation in port extension.</a></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.AutoPortExtLoss = bool</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A <a href="#">Fixturing</a> <em>(object)</em></td>
</tr>
<tr>
<td><code>bool</code></td>
<td><strong>True</strong> - Includes Loss correction. <strong>False</strong> - Does NOT include Loss correction.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fixture.AutoPortExtLoss = True</code></td>
</tr>
<tr>
<td></td>
<td><code>value = fixture.AutoPortExtLoss 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_AutoPortExtLoss(VARIANT_BOOL *pState);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_AutoPortExtLoss(VARIANT_BOOL bState);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFixturing2</td>
</tr>
</tbody>
</table>
## AutoPortExtSearchStart Property

### Description
Set the start frequency for custom user span. Only applies when `fixture.AutoPortExtConfig` = 0
naAPEC_CSPN.

Learn more about User Span.

### VB Syntax
```
fraction.AutoPortExtSearchStart = value
```

### Variable
- **(Type) - Description**
  - `fixture` A Fixturing (object)
  - `value` (Double) User span start value. Must be within the frequency range of the active channel and less than the value set by `AutoPortExtSearchStop Property`

### Return Type
Double

### Default
Start frequency of the current active channel.

### Examples
```
fraction.AutoPortExtSearchStart = 1E9
```

```
value = fraction.AutoPortExtSearchStart 'Read
```

### C++ Syntax
```
HRESULT get_AutoPortExtSearchStart(double *pdVal);
```

```
HRESULT put_AutoPortExtSearchStart(double dVal);
```

### Interface
IFixturing2
# AutoPortExtSearchStop Property

**Description**
Set the stop frequency for custom user span. Only applies when `fixture.AutoPortExtConfig = 0 naAPEC_CSPN`.

Learn more about User Span.

Only applies when `fixture.AutoPortExtConfig = 0 naAPEC_CSPN`.

**VB Syntax**

```
fixture.AutoPortExtSearchStop = value
```

**Variable**

- **Type** - Description
  - `fixture` A `Fixturing` (object)
  - `value` (Double ) User span stop value. Must be within the frequency range of the active channel and greater than the value set by `AutoPortExtSearchStart Property`

**Return Type**

Double

**Default**

Stop frequency of the current active channel.

**Examples**

```
fixture.AutoPortExtSearchStop = 1E9
```

```
value = fixture.AutoPortExtSearchStop 'Read
```

**C++ Syntax**

```cpp
HRESULT get_AutoPortExtSearchStop(double *pdVal );
HRESULT put_AutoPortExtSearchStop(double dVal);
```

**Interface**

`IFixturing2`
## AutoPortExtState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables and disables automatic port extensions on the specified port. All enabled ports will have their reference plane automatically adjusted after performing Automatic Port Extension.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.AutoPortExtState (port) = bool</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A <strong>Fixturing</strong> (object)</td>
</tr>
<tr>
<td><code>port</code></td>
<td><strong>(Integer)</strong> Port number to enable or disable.</td>
</tr>
<tr>
<td><code>bool</code></td>
<td><strong>(Boolean)</strong></td>
</tr>
<tr>
<td><code>True</code></td>
<td>Enables Auto Port Extensions</td>
</tr>
<tr>
<td><code>False</code></td>
<td>Disables Auto Port Extensions</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fixture.AutoPortExtState(1) = True</code></td>
</tr>
<tr>
<td></td>
<td><code>value = fixture.AutoPortExtState(2) </code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_AutoPortExtConfig(AutoPortExtState(short port, VARIANT_BOOL *pState );</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_AutoPortExtConfig(AutoPortExtState(short port, VARIANT_BOOL bVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFixturing2</td>
</tr>
</tbody>
</table>
### AuxiliaryTriggerCount Property

**Description**
Returns the number of aux trigger input / output connector pairs in the PNA.

**VB Syntax**
```
value = app.AuxiliaryTriggerCount
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Long Integer) Variable to store the returned value.</td>
</tr>
<tr>
<td></td>
<td>2 = N5242A models (PNA-X)</td>
</tr>
<tr>
<td></td>
<td>1 = All other PNA models</td>
</tr>
</tbody>
</table>

**app**
An Application (object)

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```
ioConns = app.AuxiliaryTriggerCount
```

**C++ Syntax**
```cpp
HRESULT AuxiliaryTriggerCount(long *count);
```

**Interface**
IAplication11

---

**Last Modified:**
14-Dec-2006

**MX New topic:**
**AuxTriggerScopeIsGlobal Property**

**Description**
Sets the Trigger OUT behavior to either Global or Channel. **Learn more about this setting.**

This command will cause the PNA to **Preset**.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

See the **AuxTrigger Object**.

**VB Syntax**

```
pref.AuxTriggerScopeIsGlobal = value
```

**Variable (Type) - Description**

- **pref** A **Preferences** (object)
- **value** (Boolean) - Choose from:
  - **True** - Trigger properties apply to ALL channels (Global).
    - Default setting for **E836x and PNA-L** models.
    - Allows use of **command** to configure the external trigger properties.
    - "Per Point" trigger property is not settable. Use the channel's **Point trigger** setting.
  - **False** - External Trigger properties apply to each channel independently.
    - Default setting for **PNA-X** models.
    - Must use **AuxTrigger** commands to configure the external trigger properties. **ExternalTriggerConnectionBehavior Property** will **NOT** work.
    - "Per Point" trigger output property is set using the channel's **Point trigger** setting **AND** **TriggerOutInterval Property**.

**Return Type**

Boolean

**Default**

- **True** - E836xB and PNA-L models
- **False** - PNA-X models

**Examples**

```
pref.AuxTriggerScopeIsGlobal = 1 'Write

auxTrigPref = pref.AuxTriggerScopeIsGlobal 'Read
```

**C++ Syntax**

```
HRESULT get_AuxTriggerScopeIsGlobal(VARIANT_BOOL * pref);
HRESULT put_AuxTriggerScopeIsGlobal(VARIANT_BOOL pref);
```
Last modified:

25-Feb-2008  Clarification
Jan 3, 2007  MX New command
**AveragingCount Property**

**Description**
Returns the number of sweeps that have been acquired and averaged into the measurements on this channel. `AveragingFactor` specifies the number of sweeps to average. `AveragingCount` indicates the progress toward that goal.

**VB Syntax**

```
value = chan.AveragingCount
```

**Variable (Type) - Description**

- **chan** A Channel (object)
- **value** (Long Integer) - Variable to store the returned count

**Return Type**
Long Integer

**Default**
Not Applicable

**Example**

```
avgcount = chan.AveragingCount
```

**C++ Syntax**

```
HRESULT get_AveragingCount(long* count)
```

**Interface**
IChannel
**AveragingFactor Property**

**Description** Specifies the number of measurement sweeps to combine for an average. Must also turn averaging ON by setting `chan.Averaging = 1`. Averaging is only allowed on ratioed measurements; not on single input measurements.

**VB Syntax**

```vbnet
chan.AveragingFactor = value
```

**Variable (Type) - Description**

- `chan` A Channel (object)

- `value` (Long Integer) - Number of measurement sweeps to average. Choose any number between 1 and 65536 ($2^{16}$).

**Return Type** Long Integer

**Default** 1

**Examples**

```vbnet
chan.AveragingFactor = 5 'Write
avgfact = chan.AveragingFactor ' Read
```

**C++ Syntax**

```cpp
HRESULT get_AveragingFactor(long *pVal)
HRESULT put_AveragingFactor(long newVal)
```

**Interface** IChannel
# Write/Read

## About Averaging

### Averaging Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns trace averaging ON or OFF for all measurements on the channel. Averaging is only allowed on ratioed measurements; not on single input measurements.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>chan.Averaging = state</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td>chan</td>
<td>A Channel <strong>(object)</strong></td>
</tr>
</tbody>
</table>
| state       | **(boolean)**  
| False       | - Turns averaging OFF                                                                                                      |
| True        | - Turns averaging ON                                                                                                       |
| Return Type | Boolean                                                                                                                   |
| Default     | False                                                                                                                     |
| Examples    | `chan.Average = True` **Write**                                                                                             |
|             | `averg = chan.Averaging` **Read**                                                                                           |
| C++ Syntax  | HRESULT get_Averaging(BOOL *pVal)  
|             | HRESULT put_Averaging(BOOL newVal)                                                                                         |
| Interface   | IChannel                                                                                                                  |
AvoidSpurs Property

**Description**
Sets and returns the state of the avoid spurs feature.

**VB Syntax**
```
mixer.AvoidSpurs = boolean
```

**Variable**
- **Type** - Description
- **mixer** A `Mixer` (object)
- **value** (Boolean) - State of avoid spurs feature. Choose from
  - `False` Avoid spurs OFF
  - `True` Avoid spurs ON

**Return Type**
Boolean

**Default**
False

**Examples**
```
mixer.AvoidSpurs = True 'Write
variable = mixer.AvoidSpurs 'Read
```

**C++ Syntax**
```
HRESULT get_AvoidSpurs(Bool *bVal)
HRESULT put_AvoidSpurs(Bool newVal)
```

**Interface**
IMixer3
**BalancedMode Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns whether the balanced transform is ON or OFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>balMeas.BalancedMode = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><em>balMeas</em> A <code>BalancedMeasurement</code> (object)</td>
</tr>
<tr>
<td></td>
<td><strong>value</strong> (Boolean) - State of balanced transform. Choose from</td>
</tr>
<tr>
<td></td>
<td><em>False</em> Balanced Transform OFF</td>
</tr>
<tr>
<td></td>
<td><em>True</em> Balanced Transform ON</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td><em>False</em></td>
</tr>
<tr>
<td>Examples</td>
<td><code>balMeas.BalancedMode = True</code> 'Write'</td>
</tr>
<tr>
<td></td>
<td><code>variable = balMeas.BalancedMode</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_BalancedMode(VARIANT_BOOL *bVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_BalancedMode(VARIANT_BOOL newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td><code>IBalancedMeasurement</code></td>
</tr>
</tbody>
</table>
**Write/Read**

**About Marker Search**

**BandwidthTarget Property**

**Description**
Sets the insertion loss value at which the bandwidth of a filter is measured (using BandwidthTracking or SearchFilterBandwidth). For example, if you want to determine the filter bandwidth 3 db below the bandpass peak value, set BandwidthTarget to -3.

**VB Syntax**

\[
\text{meas.BandwidthTarget} = \text{value}
\]

**Variable**

- **meas** (Type) - Description
  - A Measurement (object)
- **value** (single) - Target value. Choose any number between -500 and 500

**Return Type**

Single

**Default**

-3

**Examples**

\[
\begin{align*}
\text{meas.BandwidthTarget} &= -3 \quad \text{'Write} \\
\text{fbw} &= \text{meas.BandwidthTarget} \quad \text{'Read}
\end{align*}
\]

**C++ Syntax**

- HRESULT put_BandwidthTarget(float target)
- HRESULT get_BandwidthTarget(float* target)

**Interface**

IMeasurement
### BandwidthTracking Property

**Description**
Searches continually (every sweep) for the current BandwidthTarget (default is -3). To search the filter bandwidth for ONE SWEEP only (not continually), use meas.SearchFilterBandwidth.

This feature uses markers 1-4. To turn off these markers, either turn them off individually or DeleteAllMarkers.

The bandwidth statistics are displayed on the analyzer screen. To get the bandwidth statistics, use either GetFilterStatistics or FilterBW, FilterCF, FilterLoss, or FilterQ.

The analyzer screen will show either Bandwidth statistics OR Trace statistics; not both.

To restrict the search to a UserRange with the bandwidth search, first activate marker 1 and set the desired UserRange. Then send the SearchFilterBandwidth command. The user range used with bandwidth search only applies to marker 1 searching for the max value. The other markers may fall outside the user range.

#### VB Syntax

```vbnet
meas.BandwidthTracking = value
```

#### Variable

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>value</td>
<td>(boolean)</td>
</tr>
<tr>
<td>True</td>
<td>Turns bandwidth tracking ON</td>
</tr>
<tr>
<td>False</td>
<td>Turns bandwidth tracking OFF</td>
</tr>
</tbody>
</table>

#### Return Type

Boolean

#### Default

False

#### Examples

```vbnet
meas.BandwidthTracking = False 'Write
```

```vbnet
bwtrack = meas.BandwidthTracking 'Read
```

#### C++ Syntax

```csharp
HRESULT put_BandwidthTracking(VARIANT_BOOL state)
HRESULT get_BandwidthTracking(VARIANT_BOOL* state)
```

#### Interface

IMeasurement
BB_BalPort1Negative Property

**Description**
With a Balanced - Balanced topology, returns the PNA port number that is connected to the Negative side of the DUT's logical Port 1.

Use [SetBBPorts Method](#) to set the port mapping for a Balanced - Balanced topology.

**VB Syntax**
var = balTopology.BB_BalPort1Negative

**Variable**

- **Type**
- **Description**

  `var` (Long Integer) Variable to store the returned value.

**balTopology**
A [BalancedTopology](#) (object)

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**

```
variable = balTop.BB_BalPort1Negative 'Read
```

**C++ Syntax**

```
HRESULT get_BB_BalPort1Negative(long *bVal)
```

**Interface**
IBalancedTopology
BB_BalPort1Positive Property

**Description**
With a Balanced - Balanced topology, returns the PNA port number that is connected to the Positive side of the DUT's logical Port 1.

Use [SetBBPorts Method](#) to set the port mapping for a Balanced - Balanced topology.

**VB Syntax**
```vbnet
var = balTopology.BB_BalPort1Positive
```

**Variable (Type) - Description**
- `var` (Long Integer) Variable to store the returned value.

**balTopology**
A [BalancedTopology](#) (object)

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```vbnet
variable = balTop.BB_BalPort1Positive 'Read
```

**C++ Syntax**
```csharp
HRESULT get_BB_BalPort1Positive(long *bVal)
```

**Interface**
IBalancedTopology
**BB_BalPort2Negative Property**

**Description**
With a Balanced - Balanced topology, returns the PNA port number that is connected to the Negative side of the DUT’s logical Port 2.

Use [SetBBPorts Method](#) to set the port mapping for a Balanced - Balanced topology.

**VB Syntax**

```vbnet
var = balTopology.BB_BalPort2Negative
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>var</td>
<td>Long Integer</td>
<td>Variable to store the returned value.</td>
</tr>
</tbody>
</table>

**balTopology**
A BalancedTopology (object)

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**

```vbnet
variable = balTop.BB_BalPort2Negative   'Read
```

**C++ Syntax**

```cpp
HRESULT get_BB_BalPort2Negative(long *bVal)
```

**Interface**
IBalancedTopology
BB_BalPort2Positive Property

**Description**
With a Balanced - Balanced topology, returns the PNA port number that is connected to the Positive side of the DUT's logical Port 2.

Use SetBBPorts Method to set the port mapping for a Balanced - Balanced topology.

**VB Syntax**
```vbnet
var = balTopology.BB_BalPort2Positive
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>var</td>
<td>(Long Integer) Variable to store the returned value.</td>
</tr>
</tbody>
</table>

**balTopology**
A BalancedTopology (object)

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```vbnet
variable = balTop.BB_BalPort2Positive 'Read
```

**C++ Syntax**
```c++
HRESULT get_BB_BalPort2Positive(long *bVal)
```

**Interface**
IBalancedTopology
## BBalMeasurement Property

**Description**
Sets and returns the measurement for the Balanced - Balanced topology.

**VB Syntax**

```vbnet
balMeas.BBalMeasurement = value
```

**Variable**  
**Type** - Description

- `balMeas` A `BalancedMeasurement` (object)
- `value` (String) - Balanced - Balanced Measurement parameter. Not case sensitive. Choose from:

<table>
<thead>
<tr>
<th>Sdd11</th>
<th>Sdd12</th>
<th>Sdc11</th>
<th>Sdc12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sdd21</td>
<td>Sdd22</td>
<td>Sdc21</td>
<td>Sdc22</td>
</tr>
<tr>
<td>Scd11</td>
<td>Scd12</td>
<td>Scc11</td>
<td>Scc12</td>
</tr>
<tr>
<td>Scd21</td>
<td>Scd22</td>
<td>Scc21</td>
<td>Scc22</td>
</tr>
<tr>
<td>Imb1</td>
<td>Imb2</td>
<td>CMRR</td>
<td>-(Sdd21/Scc21)</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Sdd11

**Examples**

```vbnet
balMeas.BBalMeasurement = "Sdd11"  'Write
variable = balMeas.BBalMeasurement  'Read
```

**C++ Syntax**

```cpp
HRESULT get_BBalMeasurement(BSTR *pVal)
HRESULT put_BBalMeasurement(BSTR newVal)
```

**Interface**
IBalancedMeasurement
BeginResponse Property

Description
When constructing a limit line, specifies the amplitude value of the start of a limit segment.

VB Syntax
limtseg.BeginResponse = value

Variable (Type) - Description

limtseg A LimitSegment (object)

value (double) - Amplitude value. No units

Return Type
Double

Default
0

Examples
Set limtseg = meas.LimitTest(1)
limtseg.BeginResponse = 10 'Write

BegResp = limtseg.BeginResponse 'Read

C++ Syntax
HRESULT get_BeginResponse(double *pVal)
HRESULT put_BeginResponse(double newVal)

Interface
ILimitSegment
### BeginStimulus Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>When constructing a limit line, specifies the beginning X-axis value.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>limtseg.BeginStimulus = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>limtseg</code></td>
<td>A LimitSegment (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) - Stimulus value. No units</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>Set limtseg = meas.LimitTest(1)</code></td>
</tr>
<tr>
<td></td>
<td><code>limtseg.Type = naLimitSegmentType_Maximum</code></td>
</tr>
<tr>
<td></td>
<td><code>limtseg.BeginStimulus = 3e9</code></td>
</tr>
<tr>
<td></td>
<td><code>limtseg.EndStimulus = 4e9</code></td>
</tr>
<tr>
<td></td>
<td><code>limtseg.BeginResponse = 10</code></td>
</tr>
<tr>
<td></td>
<td><code>limtseg.EndResponse = 10</code></td>
</tr>
<tr>
<td><code>BegStim = limtseg.BeginStimulus</code></td>
<td><strong>Read</strong></td>
</tr>
</tbody>
</table>

#### C++ Syntax

- HRSetult `get_BeginStimulus(double *pVal)`
- HRSetult `put_BeginStimulus(double newVal)`

**Interface**

ILimitSegment
**BroadbandTuningSpan Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the frequency span for the broadband tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>embedLO.BroadbandTuningSpan = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>embedLO</code></td>
<td>An <code>EmbeddedLO</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Broadband frequency span in Hz.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>(Double)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>3 MHz</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>embedLO.BroadbandTuningSpan = 1E6 'write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = embedLO.BroadbandTuningSpan 'read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_BroadbandTuningSpan(double* span);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_BroadbandTuningSpan(double span);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEmbeddedLO</td>
</tr>
</tbody>
</table>

Last Modified: 13-Apr-2007    MX New topic
BucketNumber Property

Description
Sets or returns the bucket number (data point) for the active marker. When the markers are interpolated (non-discrete), the returned value is the nearest marker bucket position.

VB Syntax
mark.BucketNumber = value

Variable
(mark) - Description
mark A Marker (object)

value (long integer) - Data point. Choose any number between 0 and the measurement's number of data points - 1. For example, with Number of points = 201, choose between 0 and 200

Return Type
Long Integer

Default
The first marker is set to the middle of the span. Subsequent markers are set to the bucket number of the previously active marker.

Examples
mark.BucketNumber = 100 'moves the active marker to data point 100 -Write

pointNumber = mark.BucketNumber 'returns the data point number of the marker object. When the markers are interpolated (non-discrete), the returned value is the nearest marker bucket position.

C++ Syntax
HRESULT get_BucketNumber(long *pVal)
HRESULT put_BucketNumber(long newVal)

Interface
IMarker
### C0 Property

**Description**  
Sets and Returns the C0 (C-zero) value (the first capacitance value) for the calibration standard. To set the other capacitance values, use C1, C2, C3.  
For a detailed discussion of this value, search for App Note 8510-5B at [www.Agilent.com](http://www.Agilent.com).

**VB Syntax**  
`calstd.C0 = value`

**Variable**  
**Type** - Description  
- `calstd`  
  A CalStandard *(object)*. Use `calKit.GetCalStandard` to get a handle to the standard.
- `value`  
  *(single)* - Value for C0 in femtofarads (1E-15)

**Return Type**  
Single

**Default**  
Not Applicable

**Examples**  
```none  
calstd.C0 = 15 'Write the value of C0 to 15 femtofarads  
cap0 = calstd.C0 'Read the value of C0  
```

**C++ Syntax**  
```cpp  
HRESULT get_C0(float *pVal)  
HRESULT put_C0(float newVal)  
```

**Interface**  
ICalStandard
C1 Property

**Description**
Sets and Returns the C1 value (the second capacitance value) for the calibration standard. To set the other capacitance values, use C0, C2, C3.

For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

**VB Syntax**
`calstd.C1 = value`

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calstd</code></td>
<td>A CalStandard (object)</td>
<td>Use <code>calKit.GetCalStandard</code> to get a handle to the standard.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(single)</td>
<td>Value for C1.</td>
</tr>
</tbody>
</table>

**Return Type**
Single

**Default**
Not Applicable

**Examples**
- `calstd.C1 = 15 'Write the value of C1.`
- `cap1 = calstd.C1 'Read the value of C1.`

**C++ Syntax**

- `HRESULT get_C1(float *pVal)`
- `HRESULT put_C1(float newVal)`

**Interface**
ICalStandard
### C2 Property

**Description**
Sets and Returns the C2 value (the third capacitance value) for the calibration standard.

To set the other capacitance values, use `C0`, `C1`, `C3`.

For a detailed discussion of this value, search for App Note 8510-5B at [www.Agilent.com](http://www.Agilent.com).

**VB Syntax**

```
calstd.C2 = value
```

**Variable**

- **Type**: (object)
  - A CalStandard
- **Description**: Use `callKit.GetCalStandard` to get a handle to the standard.

- **Type**: (object)
  - A CalStandard
- **Description**: Use `callKit.GetCalStandard` to get a handle to the standard.

- **Type**: (single)
  - Value for C2.

**Return Type**

Single

**Default**

Not Applicable

**Examples**

```
calstd.C2 = 15 'Write the value of C2.
cap2 = calstd.C2 'Read the value of C2
```

**C++ Syntax**

```
HRESULT get_C2(float *pVal)
HRESULT put_C2(float newVal)
```

**Interface**

ICalStandard
**C3 Property**

**Description**
Sets and Returns the C3 value (the fourth capacitance value) for the calibration standard.

To set the other capacitance values, use **C0, C1, C2**

For a detailed discussion of this value, search for App Note 8510-5B at [www.Agilent.com](http://www.Agilent.com).

**VB Syntax**
```
calstd.C3 = value
```

**Variable (Type) - Description**

- **calstd** - A CalStandard *(object)*. Use calKit.GetCalStandard to get a handle to the standard.
- **value** - *(single)* - Value for C3.

**Return Type**
Single

**Default**
Not Applicable

**Examples**
```
calstd.C3 = 15 'Write the value of C3.
cap3 = calstd.C3 'Read the value of C3
```

**C++ Syntax**
```
HRESULT get_C3(float *pVal)
HRESULT put_C3(float newVal)
```

**Interface**
ICalStandard
### CalFactor Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the cal factor value associated with a power sensor cal factor segment.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calFactSeg.CalFactor = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>powerCalibrator</code></td>
<td><em>(object)</em> - A PowerSensorCalFactorSegment (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(double)</em> – Cal factor in percent. Choose any value between 1 and 150</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
</tbody>
</table>
| **Examples** | `calFactSeg.CalFactor = 98 'Write`  
`factor = calFactSeg.CalFactor 'Read` |
| **C++ Syntax** | `HRESULT put_CalFactor(Double newVal);`  
`HRESULT get_CalFactor(Double *pVal);` |
| **Interface** | `IPowerSensorCalFactorSegment` |
Write/Read

CalibrationType Property  Superseded

Description  Note: This command has been replaced by CalibrationTypeID property, which provides selection of Calibration Type by string.

Specifies the type of calibration to perform or apply to the active S-Parameter measurement. This command determine the ports involved in the CalType by the ports being used by the active measurement.

For example:

- If the measurement is an S23, it uses ports 2 and 3.
- If the measurement is an S22 it will use the legacy load port to figure out which two ports form the caltype. The legacy load port is set using CreateMeasurement.
- If naCalType_ThreePort_SOLT is specified on a 4-port PNA, an E_NA_DEPRECATED_COMMAND error is returned. There is no way to determine the intended three ports.
- If naCalType_FourPort_SOLT is specified on a 4-port PNA, it is obvious that the ports involved are ports 1,2,3, and 4.

Note: For FCA measurements, use CalibrationName and CalibrationTypeID.

VB Syntax  meas.CalibrationType = type

Variable  (Type) - Description

meas  A Measurement (object)

type  (enum NACalType) - Calibration type. Choose from:

- 0 - naCalType_Response_Open
- 1 - naCalType_Response_Short
- 2 - naCalType_Response_Thru
- 3 - naCalType_Response_Thru_And_Isol
- 4 - naCalType_OnePort
- 5 - naCalType_TwoPort_SOLT
- 6 - naCalType_TwoPortTRL
- 7 - naCalType_None
- 8 - naCalType_ThreePort_SOLT
- 9 - Custom
- 10 - naCalType_FourPort_SOLT

Return Type  NACalType
<table>
<thead>
<tr>
<th>Default</th>
<th>naCalType_None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examples</td>
<td>meas.CalibrationType = naCalType_Response_Open  <code>Write</code></td>
</tr>
<tr>
<td></td>
<td>meascal = meas.CalibrationType  <code>Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_CalibrationType (tagNACalType CalType)</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_CalibrationType (tagNACalType* pCalType)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
## CalibrationName Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the name of the current Cal Type.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = meas.CalibrationName</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td>value</td>
<td><em>(string)</em> - Variable to store the returned value.</td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>ct = meas.CalibrationName</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_CalibrationName(BSTR* CalibrationName);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
Read / Write

CalibrationPort Property - Obsolete

**Description**

Note: Beginning with Rev 6.0, this command is no longer necessary. Learn more. Because of improved calibration techniques, Both is always selected although a power meter measurement is performed only on port 1.

Specifies which SMC port to calibrate.

**VB Syntax**

```vbnet
SMC.CalibrationPort = value
```

**Variable** (Type) - Description

- **SMC**: SMCTYPE (object)
- **value**: (String) Port number to be calibrated. Choose from:
  - 1 - SMC forward
  - 2 - SMC reverse
  - Both

**Return Type**

String

**Default**

1

**Examples**

```vbnet
value = SMC.CalibrationPort = "Both"
```

**C++ Syntax**

```cpp
HRESULT put_CalibrationPort(BSTR port);
HRESULT get_CalibrationPort(BSTR *port);
```

**Interface**

SMCTYPE

VMCTYPE
CalibrationTypeID Property

Description
Note: This command replaces Calibration Type Property.
Sets or returns the current cal type for channel using a Cal Type Name.
This command is used to set the Cal Type after recalling a Cal Set. Learn more
You can also use the CLSID or GUID associated with the Cal Type.

Note: This command replaces Calibration Type Property.
Sets or returns the current cal type for channel using a Cal Type Name.
This command is used to set the Cal Type after recalling a Cal Set. Learn more
You can also use the CLSID or GUID associated with the Cal Type.

Variable (Type) - Description
meas A Measurement (object)

id (String) Cal type. Case sensitive. Use one of the following:

For Full Calibrations:
This command does not distinguish between TRL and SOLT. The same number of error terms is applied for both Cal Types.
"Full n Port(x,y,z...)"
where
n = the number of ports to calibrate
x,y,z = the port numbers to calibrate
For example:
"Full 7 Port(2,3,4,5,6,7,8)"

For Response Calibrations:
"Response(param)" OR
"ResponseAndIsolation(param)"
Where param =

- S-parameter. For example"

  - "Response(S21)"
  - "ResponseAndIsolation(A/R)"

- Single or ratioed receivers using either logical receiver notation or physical receiver notation. For example:

  - "Response(A)"
  - "ResponseAndIsolation(a3/b4)"

 VB Syntax
meas.CalibrationTypeID = id
For FCA Calibrations:

- "Scalar Mixer Cal"
- "SMC with NO Output Match Correction"
- "SMC with NO Input Match Correction"
- "SMC with NO Match Correction"
- "Vector Mixer Cal"
- "Characterize Mixer Only"

You can also use a ClassID or GUID. For example:

- CLSID - "VectorMixerCal.VCMCType"
- GUID - "{2061767B-0FE2-4F6F-86D0-9AB332B18DA5}"

For Gain Compression Cal

where r = receive port; s = source port

- "GCA 2P (r,s)" - full 2-port cal
- “GCA Enh Resp (r,s)” - Enhanced Response Cal

Return Type  String
Default        Not Applicable

Examples

Dim pna    
Dim m      
Set pna = CreateObject("AgilentPNA835x.Application")
Set m = pna.ActiveMeasurement
m.CalibrationTypeID = "Vector Mixer Cal"
M.ErrorCorrection = True
MsgBox m.CalibrationName

C++ Syntax

HRESULT get_CalibrationTypeID( BSTR* CalibrationTypeID );
HRESULT put_CalibrationTypeID( BSTR CalibrationTypeID );

Interface  IMeasurement2

Last modified:

11-Feb-2008  Fixed typo
9/12/06      MQ Added for multiport.
CalKitType Property

**Description**
Sets and returns a calibration kit type for calibration or to be used for kit modification. To get a handle to this kit, use app.ActiveCalKit.

There is also a CalKitType property on the GuidedCalibration, SMC, and VMC objects.

**VB Syntax**
```
object.CalKitType = value
```

**Variable (Type) - Description**

- **object**
  - calkit (object) or Application (object)

  **Note:** app.CalKitType and calkit.calKitType perform exactly the same function.

- **value**
  - (enum naCalKit) - Calibration Kit type. Choose from:
    - 1 - naCalKit_User1
    - 2 - naCalKit_User2
    - 3 - naCalKit_User3
    - 4 - naCalKit_User4
    - ...
    - 49 - naCalKit_User49
    - 50 - naCalKit_User50

  These enumerated values correspond with the calibration kit ID on the Advanced Cal Kit Modify dialog box.

  To change the cal kit name, use Name property.

**Return Type**
NACalKit

**Default**
Not Applicable

**Examples**
```
calkit.CalKitType = naCalKit_User27
kitype = app.CalKitType
```

**C++ Syntax**
```
HRESULT get_CalKitType(tagNACalKit *pVal);
HRESULT put_CalKitType(tagNACalKit newVal);
```

**Interface**
IApplication
ICalKit
CalKitType Property

Description
Sets and returns a calibration kit type for the specified port number to be used during the calibration.

**Note:** Sliding loads are not fully supported from the GuidedCalibration object. The **Measure** button must be pressed manually on the PNA.

VB Syntax

```
object.CalKitType (port) = value
```

Variable (Type) - Description

- **object**
  - (Type) - Description
  - **GuidedCalibration** (object)
  - **SMCType** (object)
  - **VMCType** (object)

- **port**
  - (Long) Port number to which the cal kit will be assigned.
  - For All Guided Cals, select port number.
  - For SMC and VMC calibrations:
    - 1 - Input port of the mixer under test.
    - 2 - Output port of the mixer under test.
  - For VMC cal only:
    - **Output port of MUT +1** - Output port of the calibration mixer. Generally this is port 3.
    - See the user interface (UI) equivalent of this command.

- **value**
  - (string) - Calibration Kit type.
  - Use **CompatibleCalKits** property for a list of valid Cal Kits.

Return Type
String

Default
Not Applicable

Examples

```
SMC.CalKitType(1) = naCalKit_User27
```

```
value = smc.CalKitType(1)
```

C++ Syntax

```
HRESULT get_CalKitType( long port, BSTR *calkit)
HRESULT put_CalKitType( long port, BSTR calkit)
```
Interface
  IG Guided Calibration
  SMCType
  VMCType

Last Modified:

13-Aug-2007  Added detail for port argument
## CalMethod Property

**Description**  
Sets and returns the method for performing calibration on a noise channel.

**VB Syntax**  

```vbnet
noise.CalMethod = value
```

**Variable**  
**Type** - Description  

- `noise`  
  A `NoiseCal` *(object)*

- `value`  
  *(string)* Cal Method. Choose from:
  
  - "VectorFull"
  - "SParameter"

**Return Type**  
String

**Default**  
"VectorFull"

**Examples**  

```vbnet
noise.CalMethod = "VectorFull"  'Write

calMethod = noise.CalMethod  'Read
```

**C++ Syntax**  

```cpp
HRESULT get_CalMethod(BSTR* pValue)
HRESULT put_CalMethod(BSTR pNewValue)
```

**Interface**  
`INoiseCal`

---

Last Modified:  
29-May-2007  
MN New topic
CalPower Property  Superseded

Description  This command is replaces by SetCalInfoEx Method. Specifies the power level that is expected at the desired reference plane (DUT input or output). This is not used for segment sweep with independent power levels or power sweeps.

VB Syntax  

\[
\text{value} = \text{powerCalibrator.CalPower (chan, sourcePort)}
\]

Variable  

**value**  (double) - Variable to store the returned Cal power value in dBm.

**powerCalibrator**  (object) - A SourcePowerCalibrator object

**chan**  (long integer) - Channel number of the PNA.

**sourcePort**  (long integer) - Source port number.

Use [GetPortNumber](#) to return the port number of a source that only has a string name, such as an [External Source](#).

Return Type  None

Default  0

Examples  

```javascript
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.CalPower = -10 'Write

power = powerCalibrator.CalPower 'Read
```

C++ Syntax  

```cpp
HRESULT get_CalPower(long channel, long sourcePort, double *pVal);
```

Interface  ISourcePowerCalibrator

---

Last Modified:

30-Apr-2007  Superseded
## Center Property

**Description**
Sets or returns the Center time of either Gating or Time Domain transform windows

**VB Syntax**
```vbnet
object.Center = value
```

**Variable (Type) - Description**

- `object` (object) As Gating
- `object` (object) As Transform

- `value` (double) - Center time in seconds. Choose any number between:
  
  $\pm \frac{\text{points}-1}{\text{frequency span}}$

**Return Type**
Double

**Default**
0

**Examples**

- `trans.Center = 4.5e-9` 'sets the Center time of a transform window' 
- `gate.Center = 4.5e-9` 'sets the Center time of a gating window' 

- `cnt = trans.Center` 'Read'

**C++ Syntax**

```cpp
HRESULT get_Center(double *pVal)
HRESULT put_Center(double newVal)
```

**Interface**

ITransform
IGating
Read-only

Center Property

Description Returns the stimulus value of the center data point for the measurement. This function does NOT work for segment sweep measurements. To understand how this property is useful, see IMeasurement2 Interface.

VB Syntax

```vbnet
value = meas.Center
```

Variable

- **value** (Double) - Variable to store the returned value.
- **meas** A Measurement (object)

Return Type Double

Default Not Applicable

Examples

```vbnet
Print meas.Center 'prints the center data point
```

C++ Syntax

```cpp
HRESULT get_Center(double * Val);
```

Interface IMeasurement2
### CenterFrequency Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the center frequency of the channel or Sets or returns the center frequency of the segment. See the Measurement2 Interface to learn how this method differs from meas.Center.</th>
</tr>
</thead>
</table>

#### VB Syntax

```vbnet
object.centerFrequency = value
```

#### Variable **(Type) - Description**

- **object**
  - A **Channel** (object)
  - or
  - A **Segment** (object)

- **value** **(double) - Center frequency in Hertz. Choose any number between the minimum and maximum frequencies of the analyzer.**

#### Return Type

Double

#### Default

Center of the frequency range

#### Examples

```vbnet
chan.centerFrequency = 4.5e9 'sets the center frequency of a linear sweep for the channel object -Write
centfreq = chan.centerFrequency 'Read
```

#### C++ Syntax

```cpp
HRESULT get_CenterFrequency(double *pVal)
HRESULT put_CenterFrequency(double newVal)
```

#### Interface

IChannel

ISegment
ChannelNumber Property

**Description**
Returns the Channel number of the Channel or Measurement object.

**VB Syntax**
`object.ChannelNumber`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code></td>
<td>A Channel (object) or A Measurement (object)</td>
</tr>
</tbody>
</table>

**Return Type**
Long Integer

**Default**
Not applicable

**Examples**

```vbnet
chanNum = chan.ChannelNumber 'returns the channel number
chanNum = meas.ChannelNumber 'returns the channel number of the measurement
```

**C++ Syntax**
`HRESULT get_ChannelNumber(long *pVal)`

**Interface**
IChannel
IMeasurement
### CharacterizeMixerOnly Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns whether to perform ONLY a mixer characterization.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>VMC.CharacterizeMixerOnly = bool</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>VMC</code></td>
<td><strong>VMCTYPE</strong> (object)</td>
</tr>
<tr>
<td><code>bool</code></td>
<td>(Boolean)</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Perform ONLY mixer characterization.</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Perform both mixer characterization and calibration.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td><code>value = VMC.CharacterizeMixerOnly</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT put_CharacterizeMixerOnly(VARIANT_BOOL bCharMixerOnly);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_CharacterizeMixerOnly(VARIANT_BOOL *bCharMixerOnly);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>VMCTYPE</td>
</tr>
</tbody>
</table>
**Write/Read**

**CharFileName Property**

**Description**
Specifies the mixer characterization (.S2P) file and immediately loads the file. Also specify the use of a characterization file with `LoadCharFromFile Property`.

**VB Syntax**

```
VMC.CharFileName = value
```

**Variable**

- **(Type)** - Description
  - **VMC** `VMCTYPE` (object)
  - **value** (String) Full path, file name, and extension of the mixer characterization file.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```
VMC.CharFileName = "C:\Program Files\Agilent\Network Analyzer\Documents\default.S2P"
```

**C++ Syntax**

```
HRESULT put_CharFileName(BSTR filename);
HRESULT get_CharFileName(BSTR *filename);
```

**Interface**
`VMCTYPE`
**CharMixerReverse Property**

**Description**  
Specifies the direction in which to characterize the calibration mixer. Learn more about the calibration mixer.

**VB Syntax**  
`VMC.CharMixerReverse = bool`

**Variable**  
(Variable) - Description

- **VMC**  
  `VMCTYPE (object)`

- **bool**  
  (Boolean)

  0 - Characterize the calibration mixer in the SAME direction as that specified in the mixer setup.

  1 - Characterize the calibration mixer in the REVERSE direction as that specified in the mixer setup.

**Return Type**  
Boolean

**Default**  
0

**Examples**  
`VMC.CharMixerReverse = 0`

**C++ Syntax**  
`HRESULT put_CharMixerReverse(VARIANT_BOOL bcharReverse);`

`HRESULT get_CharMixerReverse(VARIANT_BOOL *bcharReverse);`

**Interface**  
`VMCTYPE2`
### CitiContents Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the contents of subsequent citifile saves using app. <a href="#">SaveCitiDataData</a> or app. <a href="#">SaveCitiFormattedData</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pref.CitiContents = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>pref</code></td>
<td>A <a href="#">Preferences</a> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(string)</strong> - Contents that will be saved with subsequent save commands. Choose from:</td>
</tr>
<tr>
<td></td>
<td>&quot;Single&quot; - Single trace</td>
</tr>
<tr>
<td></td>
<td>&quot;Displayed&quot; - All displayed traces</td>
</tr>
<tr>
<td></td>
<td>&quot;Auto&quot; - All displayed traces</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>&quot;Auto&quot;</td>
</tr>
<tr>
<td>Examples</td>
<td><code>pref.CitiContents = &quot;Single&quot;</code> Write</td>
</tr>
<tr>
<td></td>
<td><code>content = pref.CitiContents</code> Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_CitiContents(BSTR *Contents)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_CitiContents(BSTR Contents)</td>
</tr>
<tr>
<td>Interface</td>
<td>IPreferences</td>
</tr>
</tbody>
</table>
CitiFormat Property

Description: Specifies the format of subsequent citifile saves using app.SaveCitiFormattedData.

VB Syntax: `pref.CitiFormat = value`

Variable (Type) - Description:
- `pref`: A `Preferences` (object)
- `value`: (string) - Format in which the citifile will be saved with subsequent save commands. Choose from:
  - "MA" - Linear Magnitude / degrees
  - "DB" - Log Mag / degrees
  - "RI" - Real / Imaginary
  - "Auto" - Format in which the trace is already displayed. If other than Log Mag, Linear Magnitude, or Real/Imag, then the format will be in Real/Imag.

Return Type: String

Default: "Auto"

Examples:
```
 pref.CitiFormat = "MA" 'Write
 format = pref.CitiFormat 'Read
```

C++ Syntax:
```
HRESULT get_CitiFormat(BSTR *Format)
HRESULT put_CitiFormat(BSTR Format)
```

Interface: IPreferences
CmnModeZConvPortImag Property

Description
Sets the imaginary part of the impedance value for the common port impedance conversion function.

VB Syntax
- fixture.CmnModeZConvPortImag(portNum) = value

Variable (Type) - Description
- fixture A Fixturing (object)
- portNum (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
- value (Double) Imaginary part of the Impedance value. Choose a value between 0 and 1E18.

Return Type
Double

Default
0

Examples
- fixture.CmnModeZConvPortImag(2) = 75 'Write
- value = fixture.CmnModeZConvPortImag(1) 'Read

C++ Syntax
- HRESULT get_CmnModeZConvPortImag( short portNum, double *pVal)
- HRESULT put_CmnModeZConvPortImag( short portNum, double newVal)

Interface
IFixturing2
### CmnModeZConvPortReal Property

**Description**  
Sets the real part of the impedance value for the common port impedance conversion function.

**VB Syntax**  
```vb
fixture.CmnModeZConvPortReal(portNum) = value
```

**Variable**  
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixture</td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td>portNum</td>
<td>(Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. <a href="#">Learn more about logical ports.</a></td>
</tr>
<tr>
<td>value</td>
<td>(Double) Real part of the impedance value. Choose a value between 0 and 1E18.</td>
</tr>
</tbody>
</table>

**Return Type**  
Double

**Default**  
See [Common Mode Port Z Conversion Default](#)

**Examples**  
```vb
fixture.CmnModeZConvPortReal(2) = 75 'Write
value = fixture.CmnModeZConvPortReal(1) 'Read
```

**C++ Syntax**  
```cpp
HRESULT get_CmnModeZConvPortImag( short portNum, double *pVal)
HRESULT put_CmnModeZConvPortImag( short portNum, double newVal)
```

**Interface**  
IFixturing2
Description

Sets the impedance value for the common port impedance conversion function. Set either this single value or set the real and imaginary parts separately. The imaginary part is set to 0.0 using this command.

VB Syntax

`fixture.CmnModeZConvPortZ0(portNum) = value`

Variable (Type) - Description

`fixture` A `Fixturing` (object)

`portNum` (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.

`value` (Double) Impedance value. Choose a value between 0 and 1E7.

Return Type

Double

Default

See Common Mode Port Z Conversion Default

Examples

`fixture.CmnModeZConvPortZ0(2) = 75 'Write`

`value = fixture.CmnModeZConvPortZ0(1) 'Read`

C++ Syntax

HRESULT get_CmnModeZConvPortImag( short portNum, double *pVal)
HRESULT put_CmnModeZConvPortImag( short portNum, double newVal)

Interface

`IFixturing2`
CmnModeZConvState Property

Description
Turns ON or OFF 4-port common port impedance conversion function. Must also set the fixture simulator function to ON using FixturingState Property.

VB Syntax
fixture.CmnModeZConvState = value

Variable
(Type) - Description
fixture  A Fixturing (object)
value  (Boolean)
False - Turns common port impedance conversion OFF
True - Turns common port impedance conversion ON

Return Type
Boolean

Default
False

Examples
fixture.CmnModeZConvState = False 'Write
value = fixture.CmnModeZConvState 'Read

C++ Syntax
HRESULT get_CmnModeZConvState( VARIANT_BOOL *pVal)
HRESULT put_CmnModeZConvState( VARIANT_BOOL newVal)

Interface
IFixturing2
Read-only

**CompatibleCalKits Property**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns a list of cal kits that are compatible with the connector type for the specified port. If two or more identical ECal modules are connected to the PNA, the returned list will include the serial numbers to distinguish the ECal modules.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```
value = obj.CompatibleCalKits (port)
```

**Variable (Type) - Description**

- **value** (Variant) Variable to store the returned list of Cal Kits.
- **obj** Any of the following:
  - `GuidedCalibration` (object)
  - `SMCType` (object)
  - `VMCType` (object)
- **port** (Long) Port number for which you want compatible kits.

**Return Type** Variant

**Default** Not Applicable

**Examples**

```
Dim kits As Variant
kits = MySMC.CompatibleCalKits(1)
```

**C++ Syntax**

```
HRESULT get_CompatibleCalKits(long port, VARIANT* Kits);
```

**Interface**

IGuidedCalibration

SMCType

VMCType
# CompressionAlgorithm Property

**Description**
Set and read the algorithm method used to compute gain compression.

**VB Syntax**

```vbnet
gca.CompressionAlgorithm = value
```

**Variable** 

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gca</code></td>
<td>A <code>GainCompression</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(tagNAGCACompressionAlgorithm)- Algorithm method. Choose from:</td>
</tr>
</tbody>
</table>

- naCompressionFromLinearGain (0)
- naCompressionFromMaximumGain (1)
- naBackoffCompression (2)
- naXYCompression (3)

**Return Type**
Enum

**Default**
naCompressionFromLinearGain (0)

**Examples**

- `gca.CompressionAlgorithm = naXYCompression` `Write`
- `compAlg = gca.CompressionAlgorithm` `Read`

**C++ Syntax**

```cpp
HRESULT get_CompressionAlgorithm(tagNAGCACompressionAlgorithm* pVal)
HRESULT put_CompressionAlgorithm(tagNAGCACompressionAlgorithm newVal)
```

**Interface**
IGainCompression

---

Last Modified:

11-Sep-2007    MX New topic
CompressionBackoff Property

**Description**
Set and read value for the BackOff compression algorithm.

**VB Syntax**

gca.CompressionBackoff = value

**Variable**

*Type* - Description

`gca` A GainCompression (object)

`value` (Double) Backoff value in dB. Choose from 30 to (-30)

**Return Type**
Double

**Default**
10

**Examples**

gca.CompressionBackoff = 7 'Write

acqMode = gca.CompressionBackoff 'Read

**C++ Syntax**

HRESULT get_CompressionBackoff(double* pValue)

HRESULT put_CompressionBackoff(double newValue)

**Interface**
IGainCompression

Last Modified:

21-Nov-2007 MX New topic
CompressionDeltaX Property

**Description**
Set and read the ‘X’ value in the delta X/Y compression algorithm.

**VB Syntax**
gca.CompressionDeltaX = value

**Variable**
- **Type**: Description
  - `gca` A **GainCompression** (object)
  - `value` (double) X value in dB. Choose from 30 to (-30)

**Return Type**
Double

**Default**
10

**Examples**
gca.CompressionDeltaX = 'Write

xDelta = gca.CompressionDeltaX 'Read

**C++ Syntax**
HRESULT get_CompressionDeltaX(double* pVal)
HRESULT put_CompressionDeltaX(double newVal)

**Interface**
IGainCompression

---

Last Modified:

11-Sep-2007    MX New topic
**Description**  
Set and read the “Y” value in the delta X/Y compression algorithm.

**VB Syntax**  
gca.CompressionDeltaY = value

**Variable**  
(Type) - Description

- gca  
  A **GainCompression** (object)

- value  
  (double) -

**Return Type**  
Double

**Default**  
9

**Examples**  
gca.CompressionDeltaY = 7 'Write

xDelta = gca.CompressionDeltaY 'Read

**C++ Syntax**  
HRESULT get_CompressionDeltaY(double* pVal)

HRESULT put_CompressionDeltaY(double newVal)

**Interface**  
IGainCompression

---

**Last Modified:**

11-Sep-2007  
MX New topic
CompressionInterpolation Property

**Description**
Sets whether or not interpolation should be performed on 2D measured compression data. Applies ONLY to 2D acquisition modes.

**VB Syntax**
```
gca.CompressionInterpolation = value
```

**Variable**
- **Type**
- **Description**
  - `gca`
    - A GainCompression (object)
  - `value`
    - (boolean) - Choose from:
      - **True**  Interpolate the results
      - **False**  Do NOT interpolate the results but return the value closest to compression.

**Return Type**
Boolean

**Default**
False

**Examples**
```
gca.CompressionInterpolation = True 'Write
compInt = gca.CompressionInterpolation 'Read
```

**C++ Syntax**
```
HRESULT get_CompressionInterpolation(VARIANT_BOOL* pVal)
HRESULT put_CompressionInterpolation(VARIANT_BOOL newVal)
```

**Interface**
IGainCompression

Last Modified:
11-Sep-2007  MX New topic
## CompressionLevel Property

**Description**
Set and read the desired gain reduction (from reference gain).
This value is used for [Compression Methods]: Compression from Linear Gain and Compression from Maximum Gain.

**VB Syntax**

```vbnet
 gca.CompressionLevel = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gca</code></td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) - Compression level in dB. Choose a value greater than 0.1 dB.</td>
</tr>
</tbody>
</table>

**Return Type**

Double

**Default**

1

**Examples**

```vbnet
 gca.CompressionLevel = 1.5  'Write
 compLevel = gca.CompressionLevel  'Read
```

**C++ Syntax**

```cpp
HRESULT get_CompressionLevel(double* pVal)
HRESULT put_CompressionLevel(double newVal)
```

**Interface**

IGainCompression

---

**Last Modified:**

11-Sep-2007  MX New topic
**Read / Write**

**ConnectorType Property**

**Description**
Sets or queries the connector type for the specified port.

**VB Syntax**
```
obj.ConnectorType (port) = value
```

**Variable**

**(Type) - Description**

- **obj**
  Any of the following:
  - `GuidedCalibration` (object)
  - `SMCType` (object)
  - `VMCType` (object)

- **port**
  (Long) Port number of the connector type.

- **value**
  (String) - Connector type.
  Use **ValidConnectorType Property** to list connector types.

**Return Type**
String

**Default**
None

**Examples**
```
Dim value As String
Value = MySMC.ConnectorType
```

**C++ Syntax**
```
HRESULT get_ConnectorType( long port, BSTR *connector)
HRESULT put_ConnectorType( long port, BSTR connector)
```

**Interface**
- IGuidedCalibration
- SMCType
- VMCType

---

**Last Modified:**

- 13-Aug-2007    Added detail to port argument
**ControlLines Property**

**Description**
Sets the control lines of the specified test set. Control lines, provided through the front panel connector of a test set, are used to control external equipment such as a part handler. See your test set documentation to learn more about control lines.

**VB Syntax**
```vbnet
tset.ControlLines (chNum) = value
```

**Variable**

**tset**
A `TestsetControl` object.

OR

An `E5091Testset` object.

**chNum**
(Integer) Channel number of the measurement.

**value**
(Double) Data value used to set control lines. Values are obtained by adding weights from the following table that correspond to individual lines.

<table>
<thead>
<tr>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>128</td>
</tr>
</tbody>
</table>

- The E5091A interprets SENS:MULT1:OUTP 0 as all lines LOW.
- All "Z" and "H" series test sets interpret SENS:MULT1:OUTP 0 as all lines HIGH.

Refer to your test set documentation for setting control line values.

**Return Type**
Variant

**Default**
0
Examples

'For a Z5623A K64 test set, the following sets line 3 and 4 OFF; all other lines ON.

testset1.ControlLines(2) = 12

See E5091A Example Program
See External Testset Program

C++ Syntax

HRESULT get_ControlLines(long channelNum, VARIANT *stateByte);
HRESULT put_ControlLines(long channelNum, VARIANT stateByte);

Interface

ITestsetControl
IE5091Testset

Last Modified:

17-Aug-2007    Added different test sets active High and low
### Count Property

**Description**
Returns the number of items in a collection of objects.

**VB Syntax**
`object.Count`

**Variable (Type) - Description**

- **object**
  - Any of the following (objects):
    - `CalFactorSegments collection`
    - `Cal Sets collection`
    - `Channels collection`
    - `E5091Testset Collection`
    - `ExternalTestsets Collection`
    - `LimitTest collection`
    - `Measurements collection`
    - `NaWindows collection`
    - `PowerLossSegments collection`
    - `PowerSensors collection`
    - `Segments collection`
    - `Traces collection`
    - `PowerMeterInterfaces Collection`

**Return Type**
Long Integer

**Default**
Not applicable

**Examples**
```
numofchans = chans.Count 'return the number of channels
```

**C++ Syntax**
```
HRESULT get_Count(long *p<interface>)
```

**Interface**
All listed above
**CouplePorts Property**

**Description**
Turns ON and OFF port power coupling. ON means the power level is the same for both ports. OFF means the power level may be set independently for each port.

**VB Syntax**

```
object.CouplePorts = value
```

**Variable (Type) - Description**

- **object**
  - Channel (object)
  - or
  - CalSet (object) - Read-only property

- **value**
  - (enum NAStates) Choose from:
    - 0 - NaOff - Turns coupling OFF
    - 1 - NaOn - Turns coupling ON

**Return Type**
Long Integer

- 1 - ON
- 0 - OFF

**Default**
NaON (1)

**Examples**

```
chan.CouplePorts = NaOff 'Write

couplport = chan.CouplePorts 'Read
```

**C++ Syntax**

```
HRESULT get_CouplePorts(tagNAStates *pState)
HRESULT put_CouplePorts(tagNAStates newState)
```

**Interface**

ICHannel

|CalSet3
**CoupleChannelParams Property**

**Description**
 Turns ON and OFF Time Domain Trace Coupling. All of the measurements in the specified channel are coupled.

- To select Transform parameters to couple, use `Trans.CoupledParameters Property`
- To select Gating parameters to couple, use `Gate.CoupledParameters Property`

**VB Syntax**

```vbnet
chan.CoupleChannelParams = state
```

**Variable**  
*(chan) - Description

- **chan**: A `Channel (object)`
- **state**: (boolean)
  - `False`: Turns Trace Coupling OFF
  - `True`: Turns Trace Coupling ON

**Return Type**
Boolean

**Default**
True

**Examples**

- `chan.CoupleChannelParams = False `Write`
- `couple = chan.CoupleChannelParams `Read`

**C++ Syntax**

```cpp
HRESULT get_CoupleChannelParams(VARIANT_BOOL *isCoupled);
HRESULT put_CoupleChannelParams(VARIANT_BOOL isCoupled);
```

**Interface**
IChannel5
### Coupled Property

**Description**
Sets and returns the state of coupling (ON or OFF) of this range to the primary range.

**VB Syntax**

```
FOMRange.Coupled = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>An <code>FOMRange</code> (object)</td>
</tr>
<tr>
<td>value</td>
<td>(boolean) - State of coupling.</td>
</tr>
<tr>
<td></td>
<td><code>True</code> - Couple range to primary range.</td>
</tr>
<tr>
<td></td>
<td><code>False</code> - Do NOT couple to primary range.</td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**
True

**Examples**

```vbnet
fomRange.Coupled = False 'this range is NOT coupled to the primary range.
```

```vbnet
coupl = fomRange.Coupled 'Read
```

**C++ Syntax**

```
HRESULT get_Coupled(VARIANT_BOOL *pVal)
HRESULT put_Coupled(VARIANT_BOOL pVal)
```

**Interface**
`IFOMRange`
### CoupledMarkers Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Reads the state of Coupled Markers (ON and OFF)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( \text{app.CoupledMarkers} = \text{state} )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>( \text{app} ) An Application (object)</td>
</tr>
<tr>
<td>( \text{state} ) (boolean)</td>
<td>( \text{False (0)} ) - Turns Coupled Markers OFF</td>
</tr>
<tr>
<td></td>
<td>( \text{True (1)} ) - Turns Coupled Markers ON</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td></td>
<td>( \text{False} ) - OFF</td>
</tr>
<tr>
<td></td>
<td>( \text{True} ) - ON</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( \text{app.CoupledMarkers} = \text{True} ) 'Write</td>
</tr>
<tr>
<td> </td>
<td>( \text{coupl} = \text{app.CoupledMarkers} ) 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_CoupledMarkers(VARIANT_BOOL bState)</td>
</tr>
<tr>
<td> </td>
<td>HRESULT get_CoupledMarkers(VARIANT_BOOL *bState)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
**Description**

Specifies the time domain gating parameters to be coupled. The settings for those parameters will be copied from the active measurement to all other measurements on the channel.

To turn coupling ON and OFF, use `CoupleChannelParams Property`.

To specify Transform parameters to couple, use `Transform.CoupledParameters Property`.

**VB Syntax**

```vbnet
gate.CoupledParameters = value
```

**Variable** (Type) - Description

- `trans` A `Gating (object)`
- `value` (Enum As NAGatingCoupledParams) - Parameters to couple. To specify more than one parameter, add the numbers. Choose from:
  1 - naGatingStimulusCoupled (Start, Stop, Center, and Span TIME settings.)
  2 - naGateStateCoupled (ON / OFF)
  4 - naGatingShapeCoupled (Minimum, Normal, Wide, and Maximum)
  8 - naGatingTypeCoupled (Bandpass and Notch)

**Return Type**

Enum

**Default**

29

**Examples**

```vbnet
gate.CoupledParameters = 15 'Couple all parameters
```

```vbnet
CP = gate.CoupledParameters 'Read
```

**C++ Syntax**

```
HRESULT get_CoupledParameters(long *lParams);
HRESULT put_CoupledParameters(long lParams);
```

**Interface**

IGating2
**CoupledParameters Property (Transform)**

**Description**
 Specifies the time domain transform parameters to be coupled. The settings for those parameters will be copied from the active measurement to all other measurements on the channel.

To turn coupling ON and OFF, use `CoupleChannelParams Property`

To specify Gating parameters to couple, use `Gate.CoupledParameters Property`

**VB Syntax**

```
trans.CoupledParameters = value
```

**Variable**

- **(Type)** - Description
  - `trans` A `Transform (object)`
  - `value` (Enum As NATransformCoupledParams) - Parameters to couple. To specify more than one parameter, add the numbers. Choose from:
    1 - `naTransformStimulusCoupled` (Start, Stop, Center, and Span TIME settings.)
    2 - `naTransformStateCoupled` (ON / OFF)
    4 - `naTransformWindowCoupled` (Kaiser Beta / Impulse Width)
    8 - `naTransformModeCoupled` (Low Pass Impulse, Low Pass Step, Band Pass)
    16 - `naTransformDistMkrUnitCoupled` (Distance maker Units)

**Return Type**

Enum

**Default**

29

**Examples**

```
trans.CoupledParameters = 31 'Couple all parameters

CP = trans.CoupledParameters 'Read
```

**C++ Syntax**

```
HRESULT get_CoupledParameters(long *lParams);
HRESULT put_CoupledParameters(long lParams);
```

**Interface**

`ITransform2`
Read-only

CustomChannelConfiguration Property

**Description**
Returns a handle to the custom application object on the active channel. You can either (1) use the handle directly to access measurement properties and methods, or (2) set a variable to the measurement object. The variable retains a handle to the original measurement.

Currently, the custom application objects to which this property provides access are:

- NoiseFigure Object
- GainCompression Object

**VB Syntax**
1) `set custChan = chan.CustomChannelConfiguration. <setting>`
or
2) `set custChan = app.ActiveChannel.CustomChannelConfiguration`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>custChan</td>
<td>A variable in which the handle to a custom application is returned. (object)</td>
</tr>
<tr>
<td>chan</td>
<td>A Channel (object)</td>
</tr>
</tbody>
</table>

**Return Type**
Custom application object

**Default**
None

**Examples**
See NoiseFigure or GainCompression examples

**C++ Syntax**
`HRESULT CustomChannelConfiguration(IDispatch** value);`

**Interface**
IChannel12

Last Modified:
16-Oct-2007 MX New topic
# CWFrequency Property

**Description**
Set the Continuous Wave (CW) frequency. Must first send `chan.SweepType = naCWTimeSweep`

**VB Syntax**

```vbnet
object.CWFrequency = value
```

**Variable**

- **Type** - Description
  - **object** One of the following:
    - `Channel` (object)
    - `FOMRange` (object) Range must be `UNCOPLED`.
    - `CalSet` (object) - Read-only property

See also [Measurement2 interface](#).

- **value** (double) CW frequency. Choose any number between:
  - the `minimum` and `maximum` frequency limits of the analyzer
  - Units are Hz

**Return Type**
Double

**Default**
1e9

**Examples**

```vbnet
chan.CWFrequency = 5e9 'Write

cwfreq = chan.CWFrequency 'Read
```

**C++ Syntax**

```cpp
HRESULT put_CWFrequency(double newVal)
HRESULT get_CWFrequency(double *pVal)
```

**Interface**
- `IChannel`
- `ICalSet3`
- `IFOMRange`
Delay Property

**Description**
Sets and Returns the electrical delay value for the calibration standard.

**VB Syntax**
`calstd.Delay (n) = value`

**Variable** *(Type) - Description*

- `calstd` A CalStandard *(object)*. Use `calKit.GetCalStandard` to get a handle to the standard.
- `n` Pulse Generator number.
- `value` *(single)* - Electrical delay in picoseconds

**Return Type**
Single

**Default**
Not Applicable

**Examples**
```
calstd.Delay = 12 'Write 12ps Delay

stdDelay = calstd.Delay 'Read the value of Delay
```

**C++ Syntax**

- `HRESULT get_Delay(float *pVal)`
- `HRESULT put_Delay(float newVal)`

**Interface**
ICalStandard

---

Last Modified:

5-Jan-2007    MX New topic
**Delay (pulse) Property**

**Description**
Sets the pulse delay - the amount of time before a new pulse begins.

**VB Syntax**
```
pulse.Delay(n) = value
```

**Variable**
- **Type**: Description
  - `pulse`  A **PulseGenerator** (object)
  - `n`  **(Integer)** Pulse generator number. Choose from 0 to 4.
    - 0 is the generator that pulses the ADC.
  - `value`  **(Double)** Delay value in seconds. Choose a value from about 33ns to about 70 seconds.

**Return Type**
Double

**Default**
0

**Examples**
```
pulse.Delay(1) = 1ms 'Write
```
```
value = pulse.Delay(4) 'Read
```

**C++ Syntax**
```
HRESULT get_Delay(integer pulse, double* delay);
HRESULT put_Delay(integer pulse, double delay);
```

**Interface**
IPulseGenerator

---

**Last Modified:**
1-Jan-2007  MX New topic
## Delay Property

**Description**  
Specifies the delay that should be applied by the PNA after the aux trigger input is received and before the acquisition is made.

**VB Syntax**  
`auxTrig.Delay = value`

**Variable (Type) - Description**

- **`auxTrig`**  
  An *AuxTrigger* *(object)*

- **`value`**  
  *(double)* - Delay value in seconds. Choose a value between 0 and 3.0 seconds.

**Return Type**  
Double

**Default**  
Not Applicable

**Examples**

- `auxTrig.Delay = 1.2  'Write 1.2s Delay`
- `value = auxTrig.Delay  'Read the value`

**C++ Syntax**

- `HRESULT get_Delay(double *val);`
- `HRESULT put_Delay(double val);`

**Interface**  
IAuxTrigger

---

**Last Modified:**

14-Dec-2006    MX New topic
**DelayIncrement Property**

**Description**
Sets the pulse delay increment. The delay increments with each pulse by the <value> amount.

For example, in this diagram the delay starts as 1. On the second pulse, delay=2. On the third pulse, delay=3.

**Important:** If \( D + W \) is greater than \( P \), then undefined PNA behavior results. There is NO error message or warning. Delay includes the incremented value.

This is useful for pulse profiling.

![Diagram showing delay increments](image)

**VB Syntax**

\[
pulse\text{.DelayIncrement}(n) = value
\]

**Variable**

**Type** - Description

- **pulse**
  A PulseGenerator (object)

- **n** (Integer) Pulse generator number. Choose from 0 to 4.
  0 is the generator that pulses the ADC.

- **value** (Double) Delay increment value in seconds.

**Return Type**

Double

**Default**

0

**Examples**

\[
pulse\text{.DelayIncrement}(1) = 1\text{ms} \ 'Write
value = pulse\text{.DelayIncrement}(4) \ 'Read
\]

**C++ Syntax**

HRESULT get_DelayIncrement(integer pulse, double* dInc);
HRESULT put_DelayIncrement(integer pulse, double dInc);

**Interface**

IPulseGenerator

---

Last Modified:

18-Jun-2007   MX New topic
## DeltaMarker Property

**Description**
Sets a marker as a delta marker. The reference marker must already be turned ON. See `meas.ReferenceMarkerState`.

**VB Syntax**

```
mark.DeltaMarker = state
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>A Marker (object)</td>
</tr>
<tr>
<td>state</td>
<td>(boolean) -</td>
</tr>
<tr>
<td></td>
<td>True - marker is a delta marker</td>
</tr>
<tr>
<td></td>
<td>False - marker is NOT a delta marker</td>
</tr>
</tbody>
</table>

**Return Type**

Boolean

**Default**

False

**Examples**

```
mark.DeltaMarker = True 'Write

delta = mark.DeltaMarker 'Read
```

**C++ Syntax**

```
HRESULT get_DeltaMarker(VARIANT_BOOL bState)
HRESULT put_DeltaMarker(VARIANT_BOOL *bState)
```

**Interface**

IMarker
### Description Property

Sets or returns the descriptive string assigned to the Cal Set. Change this string so that you can easily identify each Cal Set constructed.

#### Description

Sets or returns the descriptive string assigned to the Cal Set. Change this string so that you can easily identify each Cal Set constructed.

#### VB Syntax

```
CalSet.Description = value
```

#### Variable

- **Type**: Description
- **CalSet** (object) - A Cal Set object
- **value** (string) – Description of the Cal Set

#### Return Type

String

#### Default

“CalSet_n” where n is an integer number.

#### Examples

```
CalSet.Description = "My Cal Set" 'Write
desc = CalSet.Description 'Read
```

#### C++ Syntax

```
HRESULT get_Description(BSTR *pVal)
HRESULT put_Description(BSTR newVal);
```

#### Interface

ICalSet

---

Last modified:

Dec.12, 2006   MX New topic
**Description**
Write and read descriptive text associated with the configuration. This text is displayed in the path configuration dialog. Text is generally used to describe external connections that must be made manually to complete the configuration setup.

**VB Syntax**

```vbnet
pathConfig.DescriptiveText = text
```

**Variable**

- **name** *(String)* Variable to store the returned configuration name.
- **pathConfig** A `PathConfiguration` *(object)*
- **text** *(String)* Descriptive text enclosed in quotes.

**Return Type**
String

**Default**
Not Applicable

**Examples**

```vbnet
pathConf.DescriptiveText = "here are the instructions for connecting the device for this configuration"
```

**C++ Syntax**

```cpp
HRESULT get_DescriptionText(BSTR* pConnectionText);
HRESULT put_DescriptionText(BSTR connectionText);
```

**Interface**
IPathConfiguration

---

**Last Modified:**
14-Dec-2006   MX New topic
### DeviceInputPort Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Read the PNA port number which is connected to the DUT input. Use <a href="#">SetPortMap Method</a> to change the port mapping.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>gca.DeviceInputPort</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>gca</code> <strong>GainCompression</strong> (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>inPort = gca.DeviceInputPort</code> <code>Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_DeviceInputPort(int* pVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGainCompression</td>
</tr>
</tbody>
</table>

Last Modified: 11-Sep-2007  MX New topic
### InputLinearPowerLevel Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the input power at which Linear Gain and all S-parameters are measured.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>gca.InputLinearPowerLevel = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>gca</code></td>
<td>A <a href="#">GainCompression</a> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) Linear input power level in dBm. Choose a value from +30 to (-30).</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>- 25 dBm</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>gca.InputLinearPowerLevel = -10</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>LinPwr = gca.InputLinearPowerLevel</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_InputLinearPowerLevel(double* pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_InputLinearPowerLevel(double newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><a href="#">IGainCompression</a></td>
</tr>
</tbody>
</table>

Last Modified:

11-Sep-2007 MX New topic
### DeviceOutputPort Property

**Description**  
Read the PNA port number which is connected to the DUT Output.  
Use [SetPortMap Method](#) to change the port mapping.

#### VB Syntax

```vbnet
(gca).DeviceOutputPort
```

#### Variable

- **Type**: GainCompression
- **Description**: A GainCompression (object)

#### Return Type

**Integer**

#### Default

2

#### Examples

```vbnet
outPort = gca.DeviceOutputPort 'Read
```

#### C++ Syntax

```cpp
HRESULT get_DeviceOutputPort(int* pVal)
```

#### Interface

IGainCompression

---

**Last Modified:**  
11-Sep-2007  
MX New topic
**DiffPortMatch_C Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the Capacitance value of the differential matching circuit.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.DiffPortMatch_C(portNum) = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td><code>portNum</code></td>
<td>(Integer) Balanced (logical) port number. Choose from logical ports 1, 2, 3. <a href="#">Learn more about logical ports.</a></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Capacitance value in farads. Choose a value between <code>-1E18</code> to <code>1E18</code>.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fixture.DiffPortMatch_C(2) = 1e-6</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>value = fixture.DiffPortMatch_C(1)</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_DiffPortMatch_C( short portNum, double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_DiffPortMatch_C( short portNum, double newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFixturing2</td>
</tr>
</tbody>
</table>
### DiffPortMatch_G Property

**Description**
Sets the Conductance value of the differential matching circuit.

**VB Syntax**

```vbnet
fixture.DiffPortMatch_G(portNum) = value
```

**Variable**

- **(Type)** - Description
  - `fixture` A Fixturing (object)
  - `portNum` (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. [Learn more about logical ports.](#)
  - `value` (Double) Conductance value in siemens. Choose a value between $-1E18$ to $1E18$.

**Return Type**
Double

**Default**
0

**Examples**

```
fixture.DiffPortMatch_G(2) = 1e-3 'Write
value = fixture.DiffPortMatch_G(1) 'Read
```

**C++ Syntax**

```
HRESULT get_DiffPortMatch_G( short portNum, double *pVal)
HRESULT put_DiffPortMatch_G( short portNum, double newVal)
```

**Interface**
IFixturing2
## DiffPortMatch_L Property

**Description**  
Sets the Inductance value of the differential matching circuit.

**VB Syntax**  
`fixture.DiffPortMatch_L(portNum) = value`

**Variable**  
- **Type** - **Description**
  - `fixture`  
    A Fixturing (object)
  - `portNum`  
    (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
  - `value`  
    (Double) Inductance value in henries. Choose a value between `-1E18` to `1E18`.

**Return Type**  
Double

**Default**  
0

**Examples**  
```vbnet
fixture.DiffPortMatch_L(2) = 1e-3  'Write
value = fixture.DiffPortMatch_L(1)  'Read
```

**C++ Syntax**  
```c++
HRESULT get_DiffPortMatch_L( short portNum, double *pVal)
HRESULT put_DiffPortMatch_L( short portNum, double newVal)
```

**Interface**  
IFixturing2
### DiffPortMatch_R Property

**Description**
Sets the Resistance value of the differential matching circuit.

**VB Syntax**
```vbnet
fixture.DiffPortMatch_R(portNum) = value
```

**Variable**
- **Type**
- **Description**

  - **fixture**
    - **Type**: Fixturing (object)
  - **portNum**
    - **Type**: Integer
    - **Description**: Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
  - **value**
    - **Type**: Double
    - **Description**: Resistance value in ohms. Choose a value between -1E18 to 1E18.

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
fixture.DiffPortMatch_R(2) = 1e3 'Write
value = fixture.DiffPortMatch_R(1) 'Read
```

**C++ Syntax**
```cpp
HRESULT get_DiffPortMatch_R(short portNum, double *pVal)
HRESULT put_DiffPortMatch_R(short portNum, double newVal)
```

**Interface**
IFixturing2
**DiffPortMatchMode Property**

**Description**
Sets the differential matching circuit type. To select a user-defined circuit, specify IN ADVANCE the 2-port touchstone filename with `DiffPortMatch_UserFilename Property`. If you do not specify the appropriate file and you select USER, an error occurs and `naNO_CIRCUIT` is automatically selected.

**VB Syntax**
```
fraction.DiffPortMatchMode(pNum) = value
```

**Variable**
- **(Type)**: Description
- **fixture** (Type) - Description
  - A `Fixturing` (object)
- **pNum** (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
- **value** (Enum as NADiffPortMatchCircuitMode) Choose from:
  - 0 or `naSHUNT_L_SHUNT_C_CIRCUIT` - Specifies the circuit that consists of shunt L and shunt C.
  - 1 or `naUSER_FILE_CIRCUIT` - Specifies the user-defined circuit.
  - 2 or `naNO_CIRCUIT` - Specifies no-circuit.
- **Default** `naSHUNT_L_SHUNT_C_CIRCUIT`

**Examples**
```
fixture.DiffPortMatchMode(2) = naNO_CIRCUIT 'Write
value = fixture.DiffPortMatchMode(1) 'Read
```

**C++ Syntax**
```
HRESULT get_DiffPortMatchMode( short port, tagNADiffPortMatchCircuitMode *eVal)
HRESULT put_DiffPortMatchMode( short port, tagNADiffPortMatchCircuitMode eVal)
```

**Interface**
`IFixturing2`
### DiffPortMatchState Property

**Description**
Works ON or OFF 4-port differential port matching function. Must also be set fixture simulator function to ON using `FixturingState Property`.

**VB Syntax**

```vbnet
fixture.DiffPortMatchState = value
```

**Variable**  
- **Type**: (Fixturing (object))
- **value**: (Boolean)  
  - `False` - Turns differential port matching OFF
  - `True` - Turns differential port matching ON

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
fixture.DiffPortMatchState = False 'Write
value = fixture.DiffPortMatchState 'Read
```

**C++ Syntax**

```cpp
HRESULT get_DiffPortMatchState( VARIANT_BOOL *pVal)
HRESULT put_DiffPortMatchState( VARIANT_BOOL newVal)
```

**Interface**

IFixturing2
**DiffPortMatchUserFilename Property**

**Description**
Specifies the 2-port touchstone file in which the information on the user-defined differential matching circuit is saved. Following this command, send `DiffPortMatchCircuit Property`. If the specified file does not exist, an error occurs when you set the type of differential matching circuit to USER.

**VB Syntax**

```
fixture.DiffPortMatchUserFilename(pNum) = value
```

**Variable (Type) - Description**

- **fixture** A Fixturing (object)
- **pNum** (Integer) Balanced (logical) port number. Choose from logical ports 1, 2 or 3. Learn more about logical ports.
- **value** (String) Full path, file name, and extension (.s2P) of the de-embedding circuit. Files are typically stored in "C:\Program Files\Agilent\Network Analyzer\Documents"

**Return Type**
String

**Default**
Not Applicable

**Examples**

```
fixture.DiffPortMatchUserFilename(2) = "C:\Program Files\Agilent\Network Analyzer\Documents\myFile.s4p" 'Write

value = fixture.DiffPortMatchUserFilename(1) 'Read
```

**C++ Syntax**

```
HRESULT get_DiffPortMatchUserFilename( short port, BSTR *bstrFile)
HRESULT put_DiffPortMatchUserFilename( short port, BSTR bstrFile)
```

**Interface**
IFixturing2
DiffZConvPortImag Property

Description
Sets the imaginary part of the impedance value for the differential port impedance conversion function.

VB Syntax
```vbnet
fixture.DiffZConvPortImag(portNum) = value
```

Variable  (Type) - Description

- **fixture**  A *Fixturing* (object)
- **portNum** (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
- **value** (Double) Imaginary part of the Impedance value. Choose a value between 0 and 1E18

Return Type
Double

Default
0

Examples
```
fixture.DiffZConvPortImag(2) = 75 'Write
value = fixture.DiffZConvPortImag(1) 'Read
```

C++ Syntax
```
HRESULT get_DiffZConvPortImag( short portNum, double *pVal)
HRESULT put_DiffZConvPortImag( short portNum, double newVal)
```

Interface
IFixturing2
Write/Read

**DiffZConvPortReal Property**

**Description**
Sets the imaginary part of the impedance value for the differential port impedance conversion function.

**VB Syntax**
`fixture.DiffZConvPortReal(portNum) = value`

**Variable** *(Type) - Description*

*fixture*  
A *Fixturing* *(object)*

*portNum*  
*(Integer)* Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.

*value*  
*(Double)* Real part of the Impedance value. Choose a value between 0 and 1E18

**Return Type**
Double

**Default**
See *Differential Port Z Conversion Default*

**Examples**
```
fixture.DiffZConvPortReal(2) = 75 'Write
value = fixture.DiffZConvPortReal(1) 'Read
```

**C++ Syntax**
```
HRESULT get_DiffZConvPortReal( short portNum, double *pVal)
HRESULT put_DiffZConvPortReal( short portNum, double newVal)
```

**Interface**
IFixturing2
DiffZConvPortZ0 Property

Description
Sets the impedance value for the differential port impedance conversion function. Set either this single value or set the real and imaginary parts separately. The imaginary part is set to 0.0 using this command.

VB Syntax
```
fixture.DiffZConvPortZ0(portNum) = value
```

Variable
- **fixture** (Type) - Description
  - fixture: A Fixturing (object)
- **portNum** (Integer) Balanced (logical) port number. Choose from logical ports 1, 2, or 3. Learn more about logical ports.
- **value** (Double) Impedance value. Choose a value between 0 and 1E18

Return Type
Double

Default
See Differential Port Z Conversion Default

Examples
```
fixture.DiffZConvPortZ0(2) = 75 'Write
value = fixture.DiffZConvPortZ0(1) 'Read
```

C++ Syntax
```
HRESULT get_DiffZConvPortZ0( short portNum, double *pVal)
HRESULT put_DiffZConvPortZ0( short portNum, double newVal)
```

Interface
IFixturing2
**DiffZConvState Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns ON or OFF 4-port differential impedance conversion function. Must also set the fixture simulator function to ON using <code>FixturingState Property</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>fixture.DiffZConvState = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>fixture</td>
<td><code>fixture</code> A <a href="object">Fixturing</a></td>
</tr>
<tr>
<td>value</td>
<td><strong>(Boolean)</strong></td>
</tr>
<tr>
<td>False</td>
<td><code>False</code> - Turns differential impedance conversion OFF.</td>
</tr>
<tr>
<td>True</td>
<td><code>True</code> - Turns differential impedance conversion ON.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
</tbody>
</table>
| Examples    | `fixture.DiffZConvState = False` 'Write

`value = fixture.DiffZConvState` 'Read

**C++ Syntax**

`HRESULT get_DiffZConvState( VARIANT_BOOL *pVal)`

`HRESULT put_DiffZConvState( VARIANT_BOOL newVal)`

**Interface**

`IFixturing2`
**Format Property**

**Description**
Sets or returns the display format of the measurement.

**VB Syntax**
`meas.Format = value`

**Variable**
- **Type** - Description
  - `meas` (object) - A Measurement
  - `value` (enum `NADataFormat`) - Choose from:
    0 - `NADataFormat_LinMag`
    1 - `NADataFormat_LogMag`
    2 - `NADataFormat_Phase`
    3 - `NADataFormat_Polar`
    4 - `NADataFormat_Smith`
    5 - `NADataFormat_Delay`
    6 - `NADataFormat_Real`
    7 - `NADataFormat_Imaginary`
    8 - `NADataFormat_SWR`
    9 - `NADataFormat_PhaseUnwrapped`
    10 - `NADataFormat_InverseSmith`
    11 - `NADataFormat_Kelvin`
    12 - `NADataFormat_Fahrenheit`
    13 - `NADataFormat_Celsius`

**Return Type**
Long Integer

**Default**
1 - `NADataFormat_LogMag`

**Examples**
`meas.Format = NADataFormat_Real` 'Write
`fmt = meas.Format` 'Read

**C++ Syntax**
`HRESULT get_Format(tagDataFormat *pVal)`
`HRESULT put_Format(tagDataFormat newVal)`

**Interface**
`IMeasurement`

---

Last Modified:
1-Oct-2007  Added temperature formats
**DisplayAutomationErrors Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables or disables automation error messages from being displayed on the screen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>app.DisplayAutomationErrors = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Boolean)</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> allows error to show on display,</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> turns error off from display.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>True</td>
</tr>
<tr>
<td>Examples</td>
<td>Dim app As Application</td>
</tr>
<tr>
<td></td>
<td>Set app = New Application</td>
</tr>
<tr>
<td></td>
<td>app.DisplayAutomationErrors = False</td>
</tr>
<tr>
<td></td>
<td>'Turns off display</td>
</tr>
<tr>
<td></td>
<td>print app.DisplayAutomationErrors</td>
</tr>
<tr>
<td></td>
<td>'prints False</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_DisplayAutomationErrors(VARIANT_BOOL * Val);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_DisplayAutomationErrors(VARIANT_BOOL Val);</td>
</tr>
<tr>
<td>Interface</td>
<td>IApplication2</td>
</tr>
</tbody>
</table>
### DisplayGlobalPassFail Property

#### Description
Shows or hides the dialog which displays global pass/fail results. [Learn more about Global Pass/Fail.](#)

#### VB Syntax
```
app.DisplayGlobalPassFail = value
```

#### Variable (Type) - Description
- **app**: An [Application](#) (object)
- **value**: (Boolean)
  - **True**: displays the pass/fail dialog.
  - **False**: hides the pass/fail dialog.

#### Return Type
Boolean

#### Default
False

#### Examples
```
Dim app As Application
Set app = New Application
app.DisplayGlobalPassFail = true  'shows dialog
```

#### C++ Syntax
```
HRESULT get_DisplayGlobalPassFail(VARIANT_BOOL * Val);
HRESULT put_DisplayGlobalPassFail(VARIANT_BOOL Val);
```

#### Interface
IApplication6
DisplayRange Property

**Description**
Sets or returns the range to be displayed on the PNA x-axis. All traces in the channel have this same x-axis scaling.

**VB Syntax**

```vbnet
FOM.DisplayRange = value
```

**Variable**

- **object**
  - An FOM (object)

- **value**
  - (string) - Range to be displayed on the PNA x-axis. Case insensitive.

**Return Type**
String

**Default**
"Receivers"

**Examples**

```vbnet
fom.DisplayRange = "Source" 'sets the x-axis to the frequency range of "source"

disprange = fom.DisplayRange 'Read
```

**C++ Syntax**

```cpp
HRESULT get_DisplayRange(BSTR *pDspRange)
HRESULT put_DisplayRange(BSTR pDspRange)
```

**Interface**
IFOM

---

Last Modified:

- 7-Mar-2007  Changed to receivers
# Distance Property

**Description**  
Set or query marker distance on a time domain trace.

- The Write command moves the marker to the specified distance value. Once moved, you can read the **Y axis** value or **read the X-axis time** value. (Distance is calculated from the X-axis time value.)

- The Read command reads the distance of the marker.

- If the marker is set as delta, the WRITE and READ data is relative to the reference marker.

### VB Syntax

```vbnet
mark.Distance = value
```

**Variable**  
**(Type) - Description**

- **mark**  
  A Marker (object)

- **value**  
  (double) - Marker distance in the unit of measure specified with **DistanceMarkerUnit Property**

**Return Type**  
Double

**Default**  
Not Applicable

**Examples**

```vbnet
mark.Distance = 3e9 'Write
XVal = mark.Distance 'Read
```

### C++ Syntax

```cpp
HRESULT get_Distance(double *pVal);
HRESULT put_Distance(double newVal);
```

**Interface**  
IMarker2
DistanceMarkerMode Property

**Description**
Specifies the measurement type in order to determine the correct marker distance.

- Select Auto for S-Parameter measurements.
- Select Reflection or Transmission for arbitrary ratio or unratioed measurements.

This setting affects the display of ALL markers for only the ACTIVE measurement.

**VB Syntax**
```vbnet
trans.DistanceMarkerMode = value
```

**Variable** *(Type)* - Description

- **trans** *(object)*
- **value** *(enum As NADistanceMarkerMode)* - Choose from:
  - 0 - naDistanceMarkerModeAuto
  - 1 - naDistanceMarkerModeReflection
  - 2 - naDistanceMarkerModeTransmission

**Return Type**
Enum

**Default**
0 - naDistanceMarkerModeAuto

**Examples**
```vbnet
trans.DistanceMarkerMode = naDistanceMarkerModeReflection 'Write
DMM = trans.DistanceMarkerMode 'Read
```

**C++ Syntax**
```cpp
HRESULT get_DistanceMarkerMode(tagNADistanceMarkerMode *pVal);
HRESULT put_DistanceMarkerMode(tagNADistanceMarkerMode newVal);
```

**Interface**
ITransform2
DistanceMarkerUnit Property

Description
Specifies the unit of measure for the display of marker distance values. This setting affects the display of ALL markers for only the ACTIVE measurement (unless Distance Marker Units are coupled using CoupledParameters Property).

VB Syntax
```
thDistanceMarkerUnit = value
```

Variable
(Type) - Description
```
trans A Transform (object)

value (Enum As NADistanceMarkerUnit) - Distance Marker Units. Choose from
0 - naDistanceMarkerUnitMeter
1 - naDistanceMarkerUnitFeet
2 - naDistanceMarkerUnitInch
```

Return Type
Enum

Default
0 - naDistanceMarkerUnitMeter

Examples
```
trans.DistanceMarkerUnit = naDistanceMarkerUnitFeet 'sets the
U = trans.DistanceMarkerUnit 'Read
```

C++ Syntax
```
HRESULT get_DistanceMarkerUnit(tagNADistanceMarkerUnit *pVal);
HRESULT put_DistanceMarkerUnit(tagNADistanceMarkerUnit newVal);
```

Interface
ITransform2
Divisor Property

Description
Sets and returns the Divisor value to be used when coupling this range to the primary range.
This setting is valid only if the specified range is coupled to the primary range.

VB Syntax
FOMRange.Divisor = value

Variable
(object) - Description

object
An FOMRange

value
(Double) - Divisor value.-(Unitless)

Return Type
Double

Default
0

Examples
fomRange.Divisor = .5 'Write
Div = fomRange.Divisor 'Read

C++ Syntax
HRESULT get_Divisor(double *pVal)
HRESULT put_Divisor(double *pVal)

Interface
IFOMRange
Read-Write

Do1PortEcal Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specify ECAL or Mechanical calibration for the mixer characterization portion of a VMC calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>VMC.Do1PortEcal = bool</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>VMC</td>
<td>VMCType (object)</td>
</tr>
<tr>
<td>bool</td>
<td>(Boolean)</td>
</tr>
<tr>
<td>True</td>
<td>ECAL</td>
</tr>
<tr>
<td>False</td>
<td>Mechanical</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td>value = VMC.Do1PortEcal</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_Do1PortEcal(VARIANT_BOOL bDoEcal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_Do1PortEcal(VARIANT_BOOL *bDoEcal);</td>
</tr>
<tr>
<td>Interface</td>
<td>VMCType</td>
</tr>
</tbody>
</table>
Read-Write
Do2PortEcal Property

Description Specify ECAL or Mechanical calibration. For VMC, this selection only applies to the 2-port calibration portion. For mixer characterization (VMC), use Do1PortEcal Property.

VB Syntax

object.Do2PortEcal = bool

Variable (Type) - Description

object SMCTYPE (object) or VMCTYPE (object)

bool (Boolean)

True - ECAL
False - Mechanical

Return Type Boolean

Default False

Examples value = VMC.Do2PortEcal

C++ Syntax

HRESULT put_Do2PortEcal(VARIANT_BOOL bDoEcal);
HRESULT get_Do2PortEcal(VARIANT_BOOL *bDoEcal);

Interface SMCTYPE
VMCTYPE
**Domain Property**

**Description**
Returns the domain (frequency, time, power) of the measurement. To understand how this property is useful, see [IMeasurement2 Interface](#).

**VB Syntax**
```
value = meas.Domain
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Enum as NADomainType) - variable to store the returned value</td>
</tr>
<tr>
<td>0</td>
<td>Frequency</td>
</tr>
<tr>
<td>1</td>
<td>Time</td>
</tr>
<tr>
<td>2</td>
<td>Power</td>
</tr>
</tbody>
</table>

**meas**
A Measurement (object)

**Return Type**
Enum as NADomainType

**Default**
Not Applicable

**Examples**
```
Print meas.Domain 'prints the value of the domain enum
```

**C++ Syntax**
```
HRESULT get_Domain(tagNADomainType * Val);
```

**Interface**
IMeasurement2
### DUTTopology Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the device topology setting.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>balTopology.DUTTopology = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>balTopology</code></td>
<td>A <em>BalancedTopology (object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(enum NADUTTopology) - Choose either:</em></td>
</tr>
<tr>
<td></td>
<td>0 naSEBal: Single-Ended - Balanced measurement</td>
</tr>
<tr>
<td></td>
<td>1 naSESEBal: Single-Ended - Single-Ended - Balanced measurement</td>
</tr>
<tr>
<td></td>
<td>2 naBalBal: Balanced - Balanced measurement</td>
</tr>
<tr>
<td>Return Type</td>
<td>Enum as NADUTTopology</td>
</tr>
<tr>
<td>Default</td>
<td>naSEBal</td>
</tr>
<tr>
<td>Examples</td>
<td><code>balTop.DUTTopology = naSESEBal 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>DutTop = balTop.DUTTopology 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_DUTTopology(tagNADUTTopology* pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_DUTTopology(tagNADUTTopology newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IBalancedTopology</td>
</tr>
</tbody>
</table>
### DwellTime Property

**Description**
Sets or returns the dwell time at the start of each sweep point for all measurements in a channel. Dwell time is only available with Chan. `SweepGenerationMode` = `naSteppedSweep` (not `naAnalogSweep`).

Sets or returns the dwell time of a specified sweep segment.

**VB Syntax**
```
object.DwellTime = value
```

**Variable**
`object` - Description
- A Channel (object)
- or CalSet (object) - Read-only property
- or Segment (object)

`value` - (double) - Dwell Time in seconds. Choose any number between 0 and 86400

**Return Type**
Double

**Default**
0

**Examples**
```
chan.DwellTime = 3e-3 'sets the dwell time for the channel -Write
segs(3).DwellTime = 1e9 'sets the dwell time of segment 3 -Write
dwell = chan.DwellTime 'Read
```

**C++ Syntax**
```
HRESULT get_DwellTime(double *pVal)
HRESULT put_DwellTime(double newVal)
```

**Interface**
- IChannel
- ISegment
- |CalSet3
## ECALCharacterization Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the characterization data within an ECal module to be used for the SMC calibration. Learn more about <a href="#">ECal User Characterization</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( SMC.ECALCharacterization(mod) = value )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><strong>SMC</strong></td>
<td><strong>SMCType</strong> (object)</td>
</tr>
<tr>
<td><strong>module</strong></td>
<td>1 - ECal module</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0 - Factory Characterization</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( SMC.ECALCharacterization(1) = 2 )</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT put_ECALCharacterization( long moduleNumber, long characterization); HRESULT get_ECALCharacterization( long moduleNumber, long* characterization);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalibrator2</td>
</tr>
</tbody>
</table>
**ECALCharacterization Property**

**Description**
Specifies the characterization data within an ECAL module to be used, and the portion of the VMC calibration.

Learn more about [ECal User Characterization](#).

**VB Syntax**

```vba
VMC.ECALCharacterization (module, port) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>- Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMC</td>
<td>VMCType (object)</td>
</tr>
<tr>
<td>module</td>
<td>(long integer)</td>
</tr>
<tr>
<td>1</td>
<td>- ECAL module</td>
</tr>
<tr>
<td>port</td>
<td>(boolean)</td>
</tr>
<tr>
<td>True</td>
<td>- 2-port calibration portion of the VMC</td>
</tr>
<tr>
<td>False</td>
<td>- 1-port (mixer characterization portion of the VMC cal</td>
</tr>
<tr>
<td>value</td>
<td>(Long) – Characterization data within the ECAL module to be used for ECAL operations. Choose from:</td>
</tr>
<tr>
<td>0</td>
<td>– Factory Characterization</td>
</tr>
<tr>
<td>1</td>
<td>– UserCharacterization1</td>
</tr>
<tr>
<td>2</td>
<td>– UserCharacterization2</td>
</tr>
<tr>
<td>3</td>
<td>– UserCharacterization3</td>
</tr>
<tr>
<td>4</td>
<td>– UserCharacterization4</td>
</tr>
<tr>
<td>5</td>
<td>– UserCharacterization5</td>
</tr>
</tbody>
</table>

**Return Type**
Long

**Default**
0 - Factory Characterization

**Examples**

```vba
VMC.ECALCharacterization (1, True) = 4
```

**C++ Syntax**

```cpp
HRESULT put_ECALCharacterization( long moduleNumber, long characterization);
HRESULT get_ECALCharacterization( long moduleNumber, long* characterization);
```

**Interface**
ICalibrator2
Write/Read
ECALCharacterizationEx Property

**Description**

This property replaces ECALCharacterization Property. Specifies the characterization data within an ECal module to be used for the calibration. Learn more about ECal User Characterization.

**VB Syntax**

cal.ECALCharacterizationEx (module) = value

**Variable**

_Type_ - Description

*cal* Calibrator (object)

*module* (long integer) Optional argument. ECal module.

Choose from modules 1 through 8

Use IsECALModuleFoundEx to determine the number of modules connected to the PNA

Use GetECALModuleInfoEx to returns the model and serial number of each module.

*value* (Long) – Characterization data within the ECal module to be used for ECal operations. Choose from:

0 – Factory Characterization
1 – UserCharacterization1
2 – UserCharacterization2
3 – UserCharacterization3
4 – UserCharacterization4
5 – UserCharacterization5

**Return Type**

Long

**Default**

0 - Factory Characterization

**Examples**

cal.ECALCharacterizationEx (4) = 2

**C++ Syntax**

HRESULT put_ECALCharacterizationEx( long moduleNumber, long characterization);

HRESULT get_ECALCharacterizationEx( long moduleNumber, long* characterization);

**Interface**

ICalibrator4
ECALIsolation Property

**Description**

**Note:** The inherent isolation of the PNA is better than that attained with this command. ONLY use this command when using an external test set, and ONLY using a 8509x ECal module.

Specifies whether the acquisition of the ECal calibration should include isolation or not.

**VB Syntax**

`cal.ECALIsolation = value`

**Variable**

**(Type) - Description**

`cal` - A Calibrator *(object)*

`value` *(boolean)*

- False - Exclude Isolation
- True - Include Isolation

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
Dim oPNA as AgilentPNA835x.Application
Dim oCal as Calibrator
Set oPNA = CreateObject("AgilentPNA835x.Application", "MachineName")
Set oCal = oPNA.ActiveChannel.Calibrator
' Uncomment the following line to have the cal include isolation
' oCal.ECALIsolation = True
' Uncomment the following line to have the cal omit isolation
' oCal.ECALIsolation = False
oCal.DoECAL2Port ' Do the cal
```

**C++ Syntax**

```c
HRESULT put_ECALIsolation ( VARIANT_BOOL bIsolationState );
HRESULT get_ECALIsolation ( VARIANT_BOOL *bIsolationState );
```

**Interface**

Calibrator

---

Last Modified: 16-Apr-2007 Un-obsoleted
**ECALModuleNumberList Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a list of index numbers to be used for referring to the ECAL modules that are currently attached to the PNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>clist = cal.ECALModuleNumberList</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>clist</code></td>
<td>Variable to store the returned list of index numbers.</td>
</tr>
<tr>
<td><code>cal</code></td>
<td>Calibrator (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
| **Examples** | ```clist = cal.ECALModuleNumberList```  
'If 2 modules are attached to the PNA  
'then the returned list will be:  
1,2 |
| **C++ Syntax** | ```HRESULT get_ECALModuleNumberList(VARIANT *modules);``` |
| **Interface** | ICalibrator6 |
**EcalOrientation Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies which port of the ECAL module is connected to which port of the PNA when the AutoOrient property = False.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>$SMC.EcalOrientation \ (mod) = value$</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>$SMC$</td>
<td>SMCType (object)</td>
</tr>
<tr>
<td>$mod$</td>
<td>(Long)</td>
</tr>
<tr>
<td></td>
<td>1 - Use ECAL Module for the calibration.</td>
</tr>
<tr>
<td>$value$</td>
<td>(string) - Format this parameter in the following manner:</td>
</tr>
<tr>
<td></td>
<td>$Aw,Bx,Cy,Dz$</td>
</tr>
<tr>
<td></td>
<td>where</td>
</tr>
<tr>
<td></td>
<td>• A, B, C, and D are literal ports on the ECAL module</td>
</tr>
<tr>
<td></td>
<td>• w,x,y, and z are substituted for PNA port numbers to which the ECAL module port is connected.</td>
</tr>
<tr>
<td></td>
<td>Ports of the module which are not used are omitted from the string.</td>
</tr>
<tr>
<td></td>
<td>For example, on a 4-port ECAL module with</td>
</tr>
<tr>
<td></td>
<td>• port A connected to PNA port 2</td>
</tr>
<tr>
<td></td>
<td>• port B connected to PNA port 3</td>
</tr>
<tr>
<td></td>
<td>• port C not connected</td>
</tr>
<tr>
<td></td>
<td>• port D connected to PNA port 1</td>
</tr>
<tr>
<td></td>
<td>the string would be: $A2,B3,D1$</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>&quot;A1,B2&quot;</td>
</tr>
<tr>
<td>Examples</td>
<td>$SMC.EcalOrientation \ (1) = &quot;A2,B1&quot;$</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT put_EcalOrientation(long lModuleNum, BSTR orientation);</td>
</tr>
<tr>
<td></td>
<td>HRESULT get_EcalOrientation(long lModuleNum, BSTR *orientation);</td>
</tr>
<tr>
<td>Interface</td>
<td>SMCType</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
</tr>
</tbody>
</table>

998
### EcalOrientation1Port Property

**Description**  
For Mixer Characterization ONLY  
Specifies which port of the ECAL module is connected to which port of the PNA for the `Do1PortECAL` property when the `AutoOrient` property = False.

**VB Syntax**  
`VMC.EcalOrientation1Port (mod) = value`

**Variable**  
*(Type)* - Description

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VMC</strong></td>
<td><strong>VMCTYPE</strong> <em>(object)</em></td>
</tr>
</tbody>
</table>

| **mod** | *(Long)*  
1 - Use ECAL Module for the calibration. |

| **value** | *(string)* - Choose from:  
"A1" - ECAL module port A is connected to PNA port 1  
"B1" - ECAL module port A is connected to PNA port 1 |

**Return Type**  
String

**Default**  
"A1"  
If anything other than port 1 is specified, "B1" will be used. For example, if "A2" is specified, "B1" is used.

**Examples**  
`VMC.EcalOrientation1Port(1) = “B1”`

**C++ Syntax**  
`HRESULT put_EcalOrientation1Port(long lModuleNum, BSTR orientation);`  
`HRESULT get_EcalOrientation1Port(long lModuleNum, BSTR )`

**Interface**  
VMCTYPE
**Read/Write**

**EcalOrientation2Port Property**

**Description**
Specifies which port of the ECAL module is connected to which port of the PNA for the `Do2PortECAL` property when the `AutoOrient` property = False.

**VB Syntax**
```vbnet
VMC.EcalOrientation2Port (mod) = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>VMC</code></td>
<td><code>VMCType</code> (object)</td>
</tr>
<tr>
<td><code>mod</code></td>
<td>(Long) Module being used for the calibration. Choose from 1 or 2.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(string) - Format this parameter in the following manner: Aw,Bx,Cy,Dz where</td>
</tr>
<tr>
<td></td>
<td>- A, B, C, and D are literal ports on the ECAL module</td>
</tr>
<tr>
<td></td>
<td>- w,x,y, and z are substituted for PNA port numbers to which the ECAL module port is connected.</td>
</tr>
<tr>
<td></td>
<td>Ports of the module which are not used are omitted from the string.</td>
</tr>
<tr>
<td></td>
<td>For example, on a 4-port ECal module with</td>
</tr>
<tr>
<td></td>
<td>- port A connected to PNA port 2</td>
</tr>
<tr>
<td></td>
<td>- port B connected to PNA port 3</td>
</tr>
<tr>
<td></td>
<td>- port C not connected</td>
</tr>
<tr>
<td></td>
<td>- port D connected to PNA port 1</td>
</tr>
<tr>
<td></td>
<td>the string would be: A2,B3,D1</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
"A1,B2"

**Examples**
`VMC.EcalOrientation1Port(1) = "A2,B1"`

**C++ Syntax**
```cpp
HRESULT put_EcalOrientation2Port(long lModuleNum, BSTR orientation);
HRESULT get_EcalOrientation2Port(long lModuleNum, BSTR *orientation);
```
<table>
<thead>
<tr>
<th>Interface</th>
<th>VMCType</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1001</td>
</tr>
</tbody>
</table>

1001
Write/Read
ECALPortMapEx Property

**Description**
This property replaces ECALPortMap Property.
Specifies which ports of the ECal module are connected to which ports of the PNA for the DoECAL1PortEx and DoECAL2PortEx methods when the OrientECALModule property = False.

This setting remains until the PNA is restarted or this command is sent again.

**Note:** For guided calibrations where Orient is OFF and the same ECal module is used in more than one Connection Step, you are not allowed to specify how the ECal module is connected. Instead, the PNA determines the orientation. The PNA does not verify that you made the connection properly.

This command, and OrientECALModule_Property, can be used to perform ECal orientation using the Guided Calibration interface.

**VB Syntax**
```
cal.ECALPortMapEx (module) = value
```

**Variable**
**Type** - Description

- **cal** A Calibrator (object)
- **module** (long integer) Optional argument. ECal module.
  Choose from modules 1 through 8
  Use IsECALModuleFoundEx to determine the number of modules connected to the PNA
  Use GetECALModuleInfoEx to return the model and serial number of each module.
- **value** (string) - Format this parameter in the following manner:
  Aw,Bx,Cy,Dz
  where
  - A, B, C, and D are literal ports on the ECAL module
  - w,x,y, and z are substituted for PNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.
For example, on a 4-port ECal module with
- port A connected to PNA port 2
- port B connected to PNA port 3
- port C not connected
- port D connected to PNA port 1
DoECAL1PortEx or DoECAL2PortEx methods will fail if the port numbers passed to those methods are not in the string of this property and OrientECALModule property = False.

### Return Type
String

### Default
Not Applicable

### Examples
```vba
Dim cal As Calibrator
Dim sPortMap As String
Set cal = PNAapp.ActiveChannel.Calibrator
cal.ECALPortMapEx = "a2,b1" 'Write
sPortMap = cal.ECALPortMap 'Read
```

### C++ Syntax
```cpp
HRESULT put_ECALPortMapEx( long moduleNumber, BSTR strPortMap);
HRESULT get_ECALPortMapEx( long moduleNumber, BSTR *strPortMap);
```

### Interface
ICalibrator4

---

Last Modified:

7-May-2007 Added note about orient
ElecDelayMedium Property

**Description**
Sets or returns the electrical delay medium.

**VB Syntax**
`meas.ElecDelayMedium = value`

**Variable**
- **Type**: Measurement (object)
- **Description**
  - `meas` A Measurement
  - `value` (enum NACalStandardMedium) choose from
    - 0 - naCoax
    - 1 - naWaveGuide

**Return Type**
NACalStandardMedium

**Default**
Not Applicable

**Examples**
Print `meas.ElecDelayMedium` 'prints the value of the electrical delay medium

**C++ Syntax**
HRRESULT get_ElecDelayMedium(tagNACalStandardMedium *pVal);
HRRESULT put_ElecDelayMedium(tagNACalStandardMedium newVal);

**Interface**
IMeasurement2
# ElectricalDelay Property

**Description**
Sets the Electrical Delay for the active channel.

**VB Syntax**
```
meas.ElectricalDelay = value
```

**Variable**
- **meas** - A Measurement (object)
- **value** - Electrical Delay in seconds. Choose any number between -9.99 and 9.99

**Return Type**
Double

**Default**
0

**Examples**
```
meas.ElectricalDelay = 1e-3 'Write
edelay = meas.ElectricalDelay 'Read
```

**C++ Syntax**
```
HRESULT get_ElectricalDelay(double *pVal)
HRESULT put_ElectricalDelay(double newVal)
```

**Interface**
IMeasurement
Element Property

**Description**
Returns a handle to the specified PathElement object. Each element object contains a unique set of values.
The Value Property is used to set the value for each element.
See a list of configurable elements and values for various PNA models.

**VB Syntax**
Set elem = pathConfig.Element (element)

**Variable**
- **Type** - Description
  - *elem* (Object) IPathElement
  - *pathConfig* A PathConfiguration (object)
  - *element* (String) Configurable element. Use pathConfig.Elements to return a list of configurable elements or see a list of configurable elements for various PNA models.

**Return Type**
Object

**Default**
Not Applicable

**Examples**
Dim elem as PathElement
Set elem = app.ActiveChannel.PathConfiguration.Element("Src1")

**C++ Syntax**
HRESULT Element( BSTR elemName, IPathElement** ppElement);

**Interface**
IPathConfiguration

---

Last Modified:
14-Dec-2006  MX New topic
### Elements Property

**Description**
Returns an array containing the names of configurable elements. See a list of configurable elements and settings for various PNA models.

**VB Syntax**

```
values = pathConfig.Elements
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>values</td>
<td>(Variant array) Variable to store the array of configurable elements.</td>
</tr>
</tbody>
</table>

**pathConfig**
A PathConfiguration (object)

**Return Type**
Variant array

**Default**
Not Applicable

**Examples**

```
elems = pathConfig.Elements
```

**C++ Syntax**

```
HRESULT Elements( VARIANT* pElements );
```

**Interface**
IPathConfiguration

---

**Last Modified:**
14-Dec-2006

MX New topic
Embed4PortA Property

Description: Returns the PNA port number associated with 'a' based on the device topology. To see 'a' for all topologies, and to specify the port connections, use Embed4PortList Property. Specify topology using Embed4PortTopology Property.

VB Syntax: `value = fixture.Embed4PortA`

Variable: (Type) `fixture.Embed4PortA`

- value (Short Integer) Variable to store the returned PNA port number.

- `fixture` A Fixturing (object)

Return Type: Integer

Default: Not Applicable

Examples: `value = fixture.Embed4PortA 'Read`

C++ Syntax: `HRESULT get_Embed4PortA(short *portA);`

Interface: IFixturing2
## Embed4PortB Property

**Description**
Returns the PNA port number associated with 'b' based on the device topology.

To see 'b' for all topologies, and to specify the port connections, use [Embed4PortList Property](#).

Specify topology using [Embed4PortTopology Property](#).

**VB Syntax**
```
value = fixture.Embed4PortB
```

**Variable**
(Short Integer) Variable to store the returned PNA port number.

**Return Type**
Integer

**Default**
Not Applicable

**Examples**
```
value = fixture.Embed4PortB 'Read
```

**C++ Syntax**
```
HRESULT get_Embed4PortB(short *portB);
```

**Interface**
IFixturing2
Read-only

Embed4PortC Property

Description

Returns the PNA port number associated with 'c' based on the device topology.

To see 'c' for all topologies, and to specify the port connections, use Embed4PortList Property.

Specify topology using Embed4PortTopology Property

VB Syntax

\[
\text{value} = \text{fixture.Embed4PortC}
\]

Variable (Type) - Description

value (Short Integer) Variable to store the returned PNA port number.

fixture A Fixturing (object)

Return Type

Integer

Default

Not Applicable

Examples

\[
\text{value} = \text{fixture.Embed4PortC} \quad \text{'Read'}
\]

C++ Syntax

HRESULT get_Embed4PortC(short *portC);

Interface

IFixturing2
## Embed4PortD Property

**Description**
Returns the PNA port number associated with 'd' based on the device topology.

To see 'd' for all topologies, and to specify the port connections, use [Embed4PortList Property](#).

Specify topology using [Embed4PortTopology Property](#).

**VB Syntax**
```
value = fixture.Embed4PortD
```

**Variable**

**Type** - Description

- **value** (Short Integer) Variable to store the returned PNA port number.
- **fixture** A [Fixturing](#) (object)

**Return Type**
Integer

**Default**
Not Applicable

**Examples**
```
value = fixture.Embed4PortD 'Read
```

**C++ Syntax**
```
HRESULT get_Embed4PortD(short *portD);
```

**Interface**
IFixturing2
Embed4PortList Property

**Description**

Specifies the PNA port connections for ALL topologies. The port assignment is dependent on the DUT topology. All four port numbers are required. However, for:

- Topology A, only the first **two** arguments are valid,
- Topology B, only the first **three** arguments are valid,
- Topology C, **ALL** arguments are valid.

Specify topology using `Embed4PortTopology Property`.

Read the port assignments using the following commands. A, B, C, and D, refer to the port; NOT the topology.

- `Embed4PortA Property`
- `Embed4PortB Property`
- `Embed4PortC Property`
- `Embed4PortD Property`

---

**Topology A**

![Topology A Diagram](image1)

**Topology B**

![Topology B Diagram](image2)
**VB Syntax**

```vbnet
fixture.Embed4PortList = p1, p2, p3, p4
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixture</td>
<td>A Fixturing (object)</td>
</tr>
</tbody>
</table>

- **p1**: PNA Port number assigned to a in above graphic.
- **p2**: PNA Port number assigned to b in above graphic.
- **p3**: PNA Port number assigned to c in above graphic.
- **p4**: PNA Port number assigned to d in above graphic.

**Return Type**

Four Integers

**Default**

1, 2, 3, 4

**Examples**

```vbnet
fixture.4PortNetworkTopoCPortOrder = 4, 3, 2, 1 'Write
```

**C++ Syntax**

```cpp
HRESULT put_4PortNetworkTopoCPortOrder(short ChannelNum, short pPortA, short pPortB, short pPortC, short pPortD)
```

**Interface**

IFixturing2
## Embed4PortNetworkFilename Property

### Description
Specifies the 4-port touchstone file (*.s4p) in which the network to embed or de-embed resides. If the specified file does not exist, an error occurs when type command is sent. Following this command, send [Embed4PortNetworkMode Property](#).

**Note:** This command affects ALL measurements on the channel.

### VB Syntax
```vbnet
fixture.Embed4PortNetworkFilename(netNum) = value
```

### Variable
- **fixture** (A Fixturing object)
- **netNum** (Integer) Network position. Choose from 1 or 2. See [Embed4PortTopology Property](#)
- **value** (String) Full path, file name, and extension (.s4P) of the circuit.

Files are typically stored in "C:\Program Files\Agilent\Network Analyzer\Documents"

### Return Type
String

### Default
Not Applicable

### Examples
```vbnet
fixture.Embed4PortNetworkFilename(2) = "C:\Program Files\Agilent\Network Analyzer\Documents\myFile.s4p" 'Write

value = fixture.Embed4PortNetworkFilename(1) 'Read
```

### C++ Syntax
```cpp
HRESULT get_Embed4PortNetworkFilename( short networkNum, BSTR *filename);
HRESULT put_Embed4PortNetworkFilename( short networkNum, BSTR filename);
```

### Interface
IFixturing2
## Embed4PortNetworkMode Property

### Description
Specify the type of processing to take place on the specified 4-port network. First specify the network filename with `FSim.Embed4PortNetworkFilename Property`.

### VB Syntax
```vbnet
fixture.Embed4PortNetworkMode(netNum) = value
```

### Variable (Type) - Description
- **fixture** A `Fixturing (object)`
- **netNum** (Integer) Network position. Choose from 1 or 2. See `Embed4PortTopology Property`
- **value** (Enum as `NA4PortEmbedNetworkMode`) Processing mode. Choose from:
  - 0 or `naNO_NETWORK` - The same as disabling.
  - 1 or `naEMBED_NETWORK` - Add Network circuit.
  - 2 or `naDEEMBED_NETWORK` - Remove Network circuit

### Return Type
- **Enum**

### Default
- **naNO_NETWORK**

### Examples
```vbnet
fixture Embed4PortNetworkMode(1) = naNO_NETWORK 'Write

value = fixture Embed4PortNetworkMode(2) 'Read
```

### C++ Syntax
```cpp
HRESULT get_Embed4PortNetworkMode( short networkNum, tagNA4PortEmbedNetworkMode *eVal );
HRESULT put_Embed4PortNetworkMode( short networkNum, tagNA4PortEmbedNetworkMode eVal );
```

### Interface
- `IFixturing2`
Embed4PortState Property

**Description**  
Turns ON or OFF 4-port Network embedding for all ports on the channel.

**VB Syntax**  
`fixture.Embed4PortState = value`

**Variable (Type) - Description**

- **fixture** (Type)  
  A `Fixturing` (object)

- **value** (Boolean)  
  - *False* - Turns Embedding OFF
  - *True* - Turns Embedding ON

**Return Type**  
Boolean

**Default**  
False (OFF)

**Examples**

- `fixture.Embed4PortState = False `Write`
  - `value = fixture.Embed4PortState `Read`

**C++ Syntax**

- `HRESULT get_Embed4PortState(VARIANT_BOOL *pVal)`
- `HRESULT put_Embed4PortState(VARIANT_BOOL newVal)`

**Interface**  
`IFixturing2`
Write/Read

**Embed4PortTopology Property**

**Description**
Specifies the PNA / DUT topology. Learn more about these and other PNA/DUT configurations.

**VB Syntax**

```vbnet
fixture.Embed4PortTopology = value
```

**Variable**

**Type** - Description

- **fixture** A Fixturing (object)
- **value** (Enum as NA4PortEmbedTopology) PNA / DUT topology. Choose from:
  - 0 or **naTOPOLOGY_A** - 2 PNA/DUT Ports
  - 1 or **naTOPOLOGY_B** - 3 PNA/DUT Ports
  - 2 or **naTOPOLOGY_C** - 4 PNA/DUT Ports

**Return Type**
Enum

**Default**
**naTOPOLOGY_A** (2 PNA/DUT Ports)

**Examples**

```vbnet
fixture.Embed4PortTopology = naTOPOLOGY_A 'Write
```

```vbnet
value = fixture.Embed4PortTopology 'Read
```

**C++ Syntax**

```c++
HRESULT get_Embed4PortTopology( tagNA4PortEmbedTopology *eVal );
HRESULT put_Embed4PortTopology( tagNA4PortEmbedTopology eVal );
```

**Interface**
IFixturing2
# Enable Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Turns ON / OFF the trigger output.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>auxTrig.Enable = state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>auxTrig</code></td>
<td>An <code>AuxTrigger</code> (object)</td>
</tr>
<tr>
<td><code>state</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td></td>
<td><code>True</code> - Trigger Output ON</td>
</tr>
<tr>
<td></td>
<td><code>False</code> - Trigger Output OFF</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
</tbody>
</table>

**Examples**

```vbnet
auxTrig.Enable = True 'Write
value = auxTrig.Enable 'Read
```

**C++ Syntax**

```c++
HRESULT get_Enable(VARIANT_BOOL * enable);
HRESULT put_Enable(VARIANT_BOOL enable);
```

**Interface**

`IAuxTrigger`

---

**Last Modified:**

14-Dec-2006    MX New topic
Enabled Property

**Description**

Enables and disables (ON/OFF) the port mapping and control line output of the specified test set.

If the specified test set is not connected or not ON, then setting Enabled = True will report an error. All other properties can be set when the test set is not connected.

When this command is set to ON or OFF, then the display of the test set status bar (**ShowProperties Property**) is also set to ON or OFF.

**VB Syntax**

```vbnet
    tset.Enabled = value
```

**Variable**

- **tset** (Type): A **TestsetControl** object
- OR
- An **E5091Testset** object

- **value** (Boolean)
  - **True**: Enables test set control.
  - **False**: Disables test set control.

**Return Type**

Boolean

**Default**

False

**Examples**

- See E5091A Example Program
- See External Testset Program

**C++ Syntax**

```c++
    HRESULT get_Enabled(VARIANT_BOOL *state);
    HRESULT put_Enabled(VARIANT_BOOL state);
```

**Interface**

- ITestsetControl
- IE5091Testsets
EnableSourceUnleveledEvents Property

Description  Specifies whether or not to report Source Unleveled errors as system events. These events can trigger an OnSystemEvent call.

VB Syntax  

\[
\text{pref.EnableSourceUnleveledEvents} = \text{bool}
\]

Variable  (Type) - Description

\[
\text{pref} \quad \text{A Preferences (object)}
\]

\[
\text{bool} \quad \text{(Boolean)} - \text{Choose from:}
\]

\[
\text{False} - \text{Do NOT report Source Unleveled Errors.}
\]

\[
\text{True} - \text{Report Source Unleveled Errors.}
\]

Return Type  Boolean

Default  False

Examples  

\[
\text{pref.EnableSourceUnleveledEvents} = \text{False} \quad \text{Write}
\]

\[
\text{prefer} = \text{pref.EnableSourceUnleveledEvents} \quad \text{Read}
\]

C++ Syntax  

HRESULT put_EnableSourceUnleveledEvents(VARIANT_BOOL bsourcUnlEnable)

HRESULT get_EnableSourceUnleveledEvents(VARIANT_BOOL *bsourcUnlEnable)

Interface  IPreferences3

Last modified:  

Nov. 15, 2006  MX New command
Description: Set and read the action which should be taken at the end of the last frequency or power sweep in the measurement. This setting is used to protect a sensitive device from too much power during the sweep retrace.

VB Syntax:  
```
gca.EndOfSweepOperation = value
```

Variable: (Type) - Description
- **gca**: A [GainCompression](object) (object)
- **value**: ([NAGCAEndOfSweepOperation](#)

- **naStandard (0)**: Use the default PNA method. [Learn more](#).
- **naSetToStartPower (1)**: Sweep Start power
- **naSetToStopPower (2)**: Sweep Stop power.
- **naSetRFOff (3)**: Always turn power OFF while waiting.

Return Type: Enum

Default: naStandard

Examples:  
```
gca.EndOfSweepOperation = naSetToStartPower 'Write

eos = gca.EndOfSweepOperation 'Read
```

C++ Syntax:  
```
HRESULT get_EndOfSweepOperation(tagNAGCAEndOfSweepOperation* pVal)
HRESULT put_EndOfSweepOperation(tagNAGCAEndOfSweepOperation newVal)
```

Interface: IGainCompression

Last Modified: 11-Sep-2007  MX New topic
### ENRFile Property

**Description**
Sets and returns the name of the ENR file associated with the noise source.

**VB Syntax**

```vb
noise.ENRFile = value
```

**Variable**  

- `noise` (A **NoiseCal** object)
- `value` (string) Full path and ENR filename.

**Return Type**
String

**Default**
Not Applicable

**Examples**

```vb
noise.ENRFile = "c:\ProgramFiles\Agilent\Network Analyzer\Documents\ENR\346C.enr" 'Write

ENR = noise.ENRFile 'Read
```

**C++ Syntax**

```c++
HRESULT get_ENRFile(BSTR* pValue)
HRESULT put_ENRFile(BSTR pNewValue)
```

**Interface**
INoiseCal

---

**Last Modified:**  
29-May-2007   MN New topic
## ENRID Property

**Description**
Sets and returns ID of ENR table.

**VB Syntax**
```vbnet
enr.ENRSN = ID
```

### Variable - (Type) - Description

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>enr</code></td>
<td>An <code>ENRFile</code> (object)</td>
</tr>
<tr>
<td><code>ID</code></td>
<td>Identifier for the ENR table</td>
</tr>
</tbody>
</table>

### Return Type
- String

**Default**
- Not Applicable

**Examples**
[See example program](#)

### C++ Syntax
```
HRESULT get_ENRID(BSTR *Val);
HRESULT put_ENRID(BSTR Val);
```

**Interface**
- IENRFile

---

**Last Modified:**
- 2-Aug-2007  MX New topic
**ENRSN Property**

**Description**
Sets and returns the serial number of the noise source for which the ENR file applies.

**VB Syntax**
```
enr.ENRSN = serialNumber
```

**Variable**

**Type** - Description
- **enr** An **ENRFile** (object)
- **serialNumber** (String) Serial number of the noise source.

**Return Type**
String

**Default**
Not Applicable

**Examples**
See example program

**C++ Syntax**
```
HRESULT get_ENRSNBSTR *Val);
HRESULT put_ENRSN(BSTR Val);
```

**Interface**
IENRFile

---

Last Modified:

2-Aug-2007    MX New topic
## ErrorCorrection Property

**Description**
Sets (or returns) error correction ON or OFF for the measurement.

**VB Syntax**

```
meas.ErrorCorrection = value
```

**Variable**

- **meas** (Type) - Description
  - A Measurement (object)

- **value** (boolean)
  - **False** - Turns error correction OFF
  - **True** - Turns error correction ON

**Return Type**
Boolean

**Default**
See [Error Correction](#)

**Examples**

```
meas.ErrorCorrection = True 'Write

errcorr = meas.ErrorCorrection 'Read
```

**C++ Syntax**

```
HRESULT put_ErrorCorrection (VARIANT_BOOL bState)
HRESULT get_ErrorCorrection (VARIANT_BOOL *bState)
```

**Interface**
IMeasurement
### ErrorCorrection (Channel) Property

- **Description**: Attempts to sets error correction ON or OFF for all of the measurements on the channel. This setting may not be successful for some measurements because the Cal Set currently in place may not contain the appropriate calibration data. To read the error correction state for a measurement, use Error Correction Property.

- **VB Syntax**: `chan.ErrorCorrection = value`

- **Variable**
  - **chan**: A Channel (object)

- **value**
  - **(boolean)**
    - **False**: Turns error correction OFF
    - **True**: Turns error correction ON

- **Return Type**: Boolean

- **Default**: About Error Correction

- **Examples**: `chan.ErrorCorrection = True`

- **C++ Syntax**: `HRESULT put_ErrorCorrection (VARIANT_BOOL bState)`

- **Interface**: IChannel7
Write/Read

ExternalALC Property

Description
Sets or returns the source of the analyzer leveling control.

VB Syntax
app.ExternalALC = value

Variable (Type) - Description

app  An Application (object)

value (boolean) - Choose from:
True  - Leveling control supplied through the rear panel.
False - Leveling control supplied inside the analyzer

Return Type
Boolean

Default
False

Examples
app.ExternalALC = True 'Write

extALC = app.ExternalALC 'Read

C++ Syntax
HRESULT get_ExternalALC(VARIANT_BOOL *pVal)
HRESULT put_ExternalALC(VARIANT_BOOL newVal)

Interface IApplication
**ExternalTriggerConnectionBehavior Property**

**Description**
Configures the external triggering signal for the PNA.

- To control BNC1 and BNC2 with this command, then you **MUST** have `AuxTriggerScopelsGlobal` = True.
- **TriggerSource Property** is automatically set to External when `ExternalTriggerConnectionBehavior` is sent.
- Edge triggering is only available on some PNA models.
- For more information, see [External Triggering](#).

**VB Syntax**
```
trigsetup.ExternalTriggerConnectionBehavior (conn) = value
```

**Variable**
- **(Type)** - Description
  - **trigsetup** A **TriggerSetup** (object)
  - **conn** (**enum NATriggerConnection**) Rear Panel connector to send or receive trigger signals. Choose from:
    - Only one of the input connectors is active at a time. When a command is sent to one, the PNA automatically makes the other INACTIVE.
    - **0** - `naTriggerConnectionAUXT` Trigger IN from rear-panel [AUX IO connector Pin 19](#).
    - **1** - `naTriggerConnectionBNC1` Trigger IN from rear-panel [Trigger IN BNC connector](#).
    - **2** - `naTriggerConnectionBNC2` Trigger OUT to rear-panel [Trigger OUT BNC connector](#). Only useful in point sweep mode.
    - **3** - `naTriggerConnectionMATH` Trigger IN from rear-panel [Material Handler connector Pin 18](#).
  - **value** (**enum NAExternalTriggerBehavior**) -
    - **0** - `naTriggerInactive` - Disables the specified connector.
    - Choose from ONLY 1 through 4 when <conn> is set to either `naTriggerConnectionBNC1` or `naTriggerConnectionAUXT`
      - **1** - `naTriggerInEdgeNegative` - Triggers the PNA when receiving a negative going signal
      - **2** - `naTriggerInEdgePositive` - Triggers the PNA when receiving a positive going signal
      - **3** - `naTriggerInLevelLow` - Triggers the PNA when receiving a low level signal
      - **4** - `naTriggerInLevelHigh` - Triggers the PNA when receiving a High-level signal
    - Choose from ONLY 5 through 8 when <conn> is set to `naTriggerConnectionBNC2`.
      In addition to sending this command, you must also use `TriggerOutputEnabled Property` to enable...
the BNC2 output.

5 - **naTriggerOutPulsePositiveAfter** - Sends a POSITIVE going TTL pulse at the END of each point during the sweep.

6 - **naTriggerOutPulsePositiveBefore** - Sends a POSITIVE going TTL pulse at the START of each point during the sweep.

7 - **naTriggerOutPulseNegativeAfter** - Sends a NEGATIVE going TTL pulse at the END of each point during the sweep.

8 - **naTriggerOutPulseNegativeBefore** - Sends a NEGATIVE going TTL pulse at the START of each point during the sweep.

**Return Type**
Enum as NAEexternalTriggerBehavior

**Default**
BNC1 = **naTriggerInactive**
BNC2 = **naTriggerInactive**
AUXT = **naTriggerInLevelHigh**

When **Output is enabled**
BNC1 = **naTriggerInactive**
BNC2 = **naTriggerOutPulsePositiveAfter**
AUXT = **naTriggerInLevelHigh**

**Examples**
```
trigsetup.ExternalTriggerConnectionBehavior(naTriggerConnectionAUXT) =
naTriggerInLevelLow 'Write
```
```
trigBehav = trigsetup.ExternalTriggerConnectionBehavior
(naTriggerConnectionAUXT) 'Read
```

**C++ Syntax**
```c++
HRESULT get_ExternalTriggerConnectionBehavior(tagNATriggerConnection connection,tagNAExternalTriggerBehavior *trigger);
HRESULT put_ExternalTriggerConnectionBehavior(tagNATriggerConnection connection,tagNAExternalTriggerBehavior trigger);
```

**Interface**
ITriggerSetup

**Last Modified:**
25-Feb-2008    Added 'Global' note
ExternalTriggerDelay Property

Description
Sets and reads the trigger delay for all measurements in the CHANNEL. This delay is only applied while in app.Source = naTriggerSourceExternal and trigsetup.Scope = naChannelTrigger. After an external trigger is applied, the start of the sweep is delayed for the specified delay value plus any inherent latency.

To apply a trigger delay for all channels (Global), use TriggerDelay Property.

VB Syntax
chan.ExternalTriggerDelay = value

Variable  (Type) - Description
chan  A Channel (object)
value  Double- Trigger delay value in seconds. Range is from 0 to 107 seconds

Return Type
Double

Default
0

Examples
chan.ExternalTriggerDelay = .003  'Write
delay = chan.ExternalTriggerDelay  'Read

C++ Syntax
HRESULT get_ExternalTriggerDelay(double *delay);
HRESULT put_ExternalTriggerDelay(double delay)

Interface
IChannel6
## FilterBW Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the results of the SearchBandwidth method.</th>
</tr>
</thead>
</table>

**VB Syntax**

```vbnet
filtBW = meas.FilterBW
```

**Variable**

- `filtBW` *(single)* - Variable to store bandwidth data
- `meas` A Measurement *(object)*

**Return Type**

Single

**Default**

Not applicable

**Examples**

```vbnet
filterBW = meas.FilterBW 'Read
```

**C++ Syntax**

```cpp
HRESULT get_FilterBW(float* bw)
```

**Interface**

IMeasurement
### FilterCF Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Center Frequency result of the SearchBandwidth method.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>filtCF = meas.FilterCF</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>filtCF</code></td>
<td>(double) - Variable to store bandwidth CF data</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>filtCF = meas.FilterCF</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_FilterCF(double* centerFrequency)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
FilterLoss Property

**Description**
Returns the Loss value of the SearchBandwidth method.

**VB Syntax**
\[ \text{filtLoss} = \text{meas.FilterLoss} \]

**Variable**
- **filtLoss** (single) - Variable to store bandwidth Loss data
- **meas** A Measurement (object)

**Return Type**
Single

**Default**
Not applicable

**Examples**
\[ \text{filterLoss} = \text{meas.FilterLoss} \]

**C++ Syntax**
HRESULT get_FilterLoss(float* loss)

**Interface**
IMeasurement
**FilterQ Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Q (quality factor) result of the SearchBandwidth method.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>filtQ = meas.FilterQ</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>(Type) - Description</code></td>
</tr>
<tr>
<td><code>filtQ</code></td>
<td>(single) - Variable to store bandwidth Q data</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Single</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>filtQ = meas.FilterQ 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_FilterQ(float* quality)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
About PNA-X Pulsed Capabilities

FilterErrors Property

**Description**

Returns the error string associated with the digital filters. The return string has three fields separated by commas: "stage1 status, stage2 status, stage3 status"

Each of these fields can contain one or more of the following error codes:

- **NO ERROR**
- **NUMBER-OF-COEFFICIENTS** - the number of coefficients is excessive for that filter-stage
- **COEFFICIENT VALUE** - one or more coefficients are out of range for that filter-stage
- **SUM-OF-COEFFICIENTS** - the sum of all coefficients is excessive for that filter-stage,
- **FREQUENCY** - the frequency for Stage 1 is out of range (only applies stage1 field),
- **PARAMETER** - one or more parameters are out of range (only applies to stage 3 field)

**VB Syntax**

```
value = spm4.FilterErrors
```

**Variable (Type) - Description**

- **value** Variable to store the returned errors.
- **spm4** A SignalProcessingModuleFour (object)

**Return Type**

String

**Default**

Not Applicable

**Examples**

```
mode = spm4.FilterErrors 'Read
'example return strings"

NO ERROR, NO ERROR, NO ERROR
indicates no errors,

*SUM-OF-COEFFICIENTS, NO ERROR, NO ERROR
indicates that the sum of all filter coefficients exceed the maximum value for the Stage-1 filter,

*COEFFICIENT *SUM-OF-COEFFICIENTS, NO ERROR, *PARAMETER
indicates a problems with Stage 1 coefficients and a problem with one or more of the parameters associated with the Stage 3 filter.
```

**C++ Syntax**

```
HRESULT get_FilterErrors(BSTR* dspErrors);
```
**FilterMode Property**

**Description**
Sets and returns whether the PNA configures the 3-stage digital filter settings or they will be configured manually. When making manual settings, also send **ADCCaptureMode Property** which routes the IF through the 3-stage filter.

**VB Syntax**

```vb
spm4.FilterMode = value
```

**Variable**

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>spm4</code></td>
<td>A <strong>SignalProcessingModuleFour</strong> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum as NAModes) Filter mode. Choose from:</td>
</tr>
</tbody>
</table>

- **naAUTO**  PNA controls digital filter settings.
- **naMANUAL** You control digital filter settings using other **SignalProcessingModuleFour** commands.

**Return Type**

Enum

**Default**

`naAUTO`

**Examples**

```vb
spm4.FilterMode = naAUTO 'Write

mode = spm4.FilterMode 'Read
```

**C++ Syntax**

```c++
HRESULT get_FilterMode(tagNAModes* dspMode);
HRESULT put_FilterMode(tagNAModes dspMode);
```

**Interface**

`ISignalProcessingModuleFour`

---

**Last Modified:**

24-Jan-2007    MX New topic
FirmwareMajorRevision Property

Description
Returns the major firmware revision number as an integer. For example, given a firmware revision number A.03.30, this command returns 3.

VB Syntax
value = cap.FirmwareMajorRevision

Variable
(Type) - Description

value (Long) - Variable to store the returned integer value of the firmware revision number.

cap A Capabilities (object)

Return Type
Long

Default
Not Applicable

Examples
value = cap.FirmwareMajorRevision 'Read

C++ Syntax
HRESULT get_FirmwareMajorRevision(long * majorRev );

Interface
ICapabilities
FirmwareMinorRevision Property

Description
Returns the minor firmware revision number as an integer. For example, given a firmware revision number A.03.30, this command returns 30.

VB Syntax
value = cap.FirmwareMinorRevision

Variable (Type) - Description
value (Long) - Variable to store the returned decimal value of the firmware revision number.
cap A Capabilities (object)

Return Type
Long

Default
Not Applicable

Examples
value = cap.FirmwareMinorRevision 'Read

C++ Syntax
HRESULT get_FirmwareMinorRevision(long * minorRev);

Interface
ICapabilities
### FirmwareSeries Property

**Description**
Returns the alpha portion of the firmware revision number. For example, given a firmware revision number A.03.30, this command returns A.

**VB Syntax**

```
value = cap.FirmwareSeries
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(String) - Variable to store the returned alpha value of the firmware revision number.</td>
</tr>
</tbody>
</table>

| cap    | A Capabilities (object) |

**Return Type**
String

**Default**
Not Applicable

**Examples**

```
value = cap.FirmwareSeries 'Read
```

**C++ Syntax**

```
HRESULT get_FirmwareSeries(BSTR * series);
```

**Interface**
ICapabilities
### FixturingState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns all three fixturing functions (de-embedding, port matching, impedance conversion) ON or OFF for all ports on the specified channel. This does NOT affect port extensions.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>fixture.FixturingState = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Turns Fixturing ON</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Turns Fixturing OFF</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td><code>fixture.FixturingState = True</code> <strong>Write</strong></td>
</tr>
<tr>
<td></td>
<td><code>value = fixture.FixturingState</code> <strong>Read</strong></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_FixturingState(VARIANT_BOOL *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_FixturingState(VARIANT_BOOL newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IFixturing</td>
</tr>
</tbody>
</table>
### Read-only

**FootSwitch Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Reads the Footswitch Input (pin 20 of the AUX IO connector).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = AuxIO.Footswitch</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(boolean) - Variable to store the returned value</td>
</tr>
<tr>
<td>False</td>
<td>foot switch is released</td>
</tr>
<tr>
<td>True</td>
<td>footswitch is depressed</td>
</tr>
<tr>
<td><strong>AuxIO</strong></td>
<td>(object) - A Hardware Aux I/O object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fs = aux.Footswitch</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_FootSwitch (VARIANT_BOOL* State);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IHWAuxIO</td>
</tr>
</tbody>
</table>
FootswitchMode Property

**Description**  
Determines what occurs when the footswitch is pressed. For more information see the FootSwitch In pin description in the Auxiliary IO connector.

**VB Syntax**  
`AuxIo.FootSwitchMode = value`

**Variable (Type) - Description**  
`value`  
`enum NAFootSwitchMode`

0 - `naIgnoreFootswitch` - Footswitch presses are ignored.

1 - `naSweepTrigger` - Footswitch presses trigger a sweep. The PNA must be in Manual Trigger Mode.

2 - `naRecallNextState` - Footswitch presses recall an instrument state. When more than one state is available, then each footswitch press recalls the next state, then starts over from the beginning. It is possible for a recalled state to override the current mode. If the recalled state is IGNore, then mode changes and additional footswitch presses are ignored.

3 - `naRunMacro` - Footswitch presses load and run a macro. When more than one macro is available, then each footswitch press loads and runs the next macro, then starts over from the beginning. It is possible for a Macro to override the current mode. If the macro contains a Preset, then the mode changes to the default setting IGNore and additional footswitch presses are ignored.

**AuxIO (object) - A Hardware Aux I/O object**

**Return Type**  
NAFootSwitchMode

**Default**  
`0 - naIgnoreFootswitch`

**Examples**  
`auxIo.FootSwitchMode = naIgnoreFootSwitch`  
`Write`

**C++ Syntax**  
`HRESULT get_FootSwitchMode(NAFootSwitchMode *pFootSwitchMode )`

`HRESULT put_FootSwitchMode(NAFootSwitchMode newFootSwitchMode)`

**Interface**  
IHWAuxIO3
### Format Property (marker)

**Description**
Sets (or returns) the format of the marker.

**VB Syntax**
```
mark.Format = value
```

**Variable**
- **(Type)** - Description
  - `mark` A Marker *(object)*
  - `value` *(enum NAMarkerFormat)* - Choose from:
    - 0 - naMarkerFormat_LinMag
    - 1 - naMarkerFormat_LogMag
    - 2 - naMarkerFormat_Phase
    - 3 - naMarkerFormat_Delay
    - 4 - naMarkerFormat_Real
    - 5 - naMarkerFormat_Imaginary
    - 6 - naMarkerFormat_SWR
    - 7 - naMarkerFormat_LinMagPhase
    - 8 - naMarkerFormat_LogMagPhase
    - 9 - naMarkerFormat_RealImaginary
    - 10 - naMarkerFormat_ComplexImpedance
    - 11 - naMarkerFormat_ComplexAdmittance
    - 12 - naMarkerFormat_Kelvin
    - 13 - naMarkerFormat_Fahrenheit
    - 14 - naMarkerFormat_Celsius

**Return Type**
NAMarkerFormat

**Default**
1 - naMarkerFormat_LogMag

**Examples**
```
mark.Format = naMarkerFormat_SWR 'Write
fmt = mark.Format 'Read
```

**C++ Syntax**
- HRESULT `get_Format(tagNAMarkerFormat *pVal)`
- HRESULT `put_Format(tagNAMarkerFormat newVal)`

**Interface**
IMarker

---

Last Modified:
1-Oct-2007  Added temperature formats
## Frequency Property

**Description**
Sets or returns the frequency associated with a PowerSensorCalFactorSegment or
Sets or returns the frequency associated with a PowerLossSegment.

**VB Syntax**
```vbnet
object.Frequency = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code></td>
<td>PowerSensorCalFactorSegment or PowerLossSegment</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) – Frequency in units of Hz. This can be any non-negative value (limited by the maximum value of double).</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
seg.Frequency = 6e9  'Write
d = seg.Frequency  'Read
```

**C++ Syntax**
```c
HRESULT put_Frequency(double newVal);
HRESULT get_Frequency(double *pVal);
```

**Interface**

- IPowerSensorCalFactorSegment
- IPowerLossSegment
## FrequencySpan Property

**Description**
Sets or returns the frequency span of the channel.
Sets or returns the frequency span of the segment.

**VB Syntax**

```vbnet
object.FrequencySpan = value
```

**Variable**

- **Type:**
- **Description:**

  - `object`  A Channel *(object)*
  - or
  - A Segment *(object)*

- **value** *(double)* - Frequency span in Hertz. Choose any number between the **minimum** and **maximum** frequencies of the analyzer.

**Return Type**

- Double

**Default**

- Full frequency span of the analyzer

**Examples**

```vbnet
chan.FrequencySpan = 4.5e9 'sets the frequency span of a linear sweep for the channel object -Write

freqspan = chan.FrequencySpan 'Read
```

**C++ Syntax**

- `HRESULT get_FrequencySpan(double *pVal)`
- `HRESULT put_FrequencySpan(double newVal)`

**Interface**

- IChannel
- ISegment
Read / Write
FrequencyList Property

Description

VB Syntax  

\[ \text{guidedCal.FrequencyList} = \text{value} \]

Variable  

(Type) - Description

guidedCal  

GuidedCalibration (object)

value  

(Variant) -

Return Type

Variant

Default

Examples

Dim value
value = MySMC.FrequencyList

C++ Syntax

HRESULT get_FrequencyList(Variant *freqList)
HRESULT put_FrequencyList(Variant freqList)

Interface

IGuidedCalibration
### FrequencyOffsetDivisor Property  **Superseded**

**Description**
This method is replaced by properties on the FOMRange Object. Specifies (along with FrequencyOffsetMultiplier) the value to multiply by the stimulus. See other Frequency Offset properties.

**VB Syntax**
```
object.FrequencyOffsetDivisor = value
```

**Variable**
- **object** Channel (object)
  - or
  - CalSet (object) - Read-only property
- **value** (Double) - Divisor value. Range is 1 to 1000

**Return Type**
Double

**Default**
1

**Examples**
```
chan.FrequencyOffsetDivisor = 2 'Write
fOffsetDiv = chan.FrequencyOffsetDivisor 'Read
```

**C++ Syntax**
```
HRESULT get_FrequencyOffsetDivisor(double*pval)
HRESULT put_FrequencyOffsetDivisor(double newVal)
```

**Interface**
|IChannel2|
|CalSet3|
### FrequencyOffsetFrequency Property  Superseded

**Description**

This method is replaced by properties on the FOMRange Object. Specifies an absolute offset frequency in Hz. For mixer measurements, this would be the LO frequency. See other Frequency Offset properties.

**VB Syntax**

```
object.FrequencyOffsetFrequency = value
```

**Variable**

*(Type) - Description*

- **object**: Channel *(object)*
  - or
  - CalSet *(object)* - Read-only property

- **value**: *(Double)* - Offset value. Range is +/- 1000 GHz. (Offsets can be positive or negative.)

**Return Type**

Double

**Default**

0 Hz

**Examples**

```vbnet
chan.FrequencyOffsetFrequency = 2 'Write
fOffsetFreq = chan.FrequencyOffsetFrequency 'Read
```

**C++ Syntax**

```cpp
HRESULT get_FrequencyOffsetFrequency(double*pval)
HRESULT put_FrequencyOffsetFrequency(double newVal)
```

**Interface**

IChannel2

|CalSet3
**FrequencyOffsetMultiplier Property  Superseded**

**Description**
This method is replaced by properties on the FOMRange Object.

Specifies (along with FrequencyOffsetDivisor) the value to multiply by the stimulus. See other Frequency Offset properties.

**VB Syntax**

```
object.FrequencyOffsetMultiplier = value
```

**Variable**

- **object** Channel *(object)*
- **or**
- CalSet *(object)* - Read-only property

- **value** *(Double)* - Multiplier value. Range is 1 to 1000

**Return Type**

Double

**Default**

1

**Examples**

```
chan.FrequencyOffsetMultiplier = 2 'Write
fOffsetMult = chan.FrequencyOffsetMultiplier 'Read
```

**C++ Syntax**

```
HRESULT get_FrequencyOffsetMultiplier (double*pval);
HRESULT put_FrequencyOffsetMultiplier (double newVal);
```

**Interface**

- IChannel2
- ICalSet3
**FrequencyOffsetCWOverride Property  Superseded**

**Description**
This method is replaced by properties on the FOMRange Object. Establishes a fixed (CW) stimulus frequency while measuring the Response over a swept frequency range. For example, a fixed-frequency PNA stimulus may be applied to the RF input of a mixer whose local oscillator (LO) is being swept. Because the IF output of the mixer will be swept, the PNA receivers must also be swept.

See other Frequency Offset properties.

**VB Syntax**

```
object.FrequencyOffsetCWOverride = value
```

**Variable**  

(Variable) - Description

- **object**  
  Channel (object)

  or

  CalSet (object) - Read-only property

- **value**  
  (Enum as NaStates) - Choose from:
  
  - naOFF (0) - Turns CW override OFF
  - naON (1) - Turns CW override ON

**Return Type**  

Enum

**Default**  

0 Hz

**Examples**

```
chan.FrequencyOffsetCWOverride = 1 'Write

fOffsetOV = chan.FrequencyOffsetCWOverride 'Read
```

**C++ Syntax**

```
HRESULT get_FrequencyOffsetCWOverride (tagNAStates *pstate)
HRESULT put_FrequencyOffsetCWOverride (tag NAStates newState)
```

**Interface**

- IChannel2
  - |CalSet3
## FrequencyOffsetState Property  Superseded

### Description
This method is replaced by properties on the FOMRange Object. Enables Frequency Offset on ALL measurements that are present on the active channel. This immediately causes the source and receiver to tune to separate frequencies. The receiver frequencies are specified with other channel and offset settings. To make the stimulus settings, use Channel Start, Stop Frequency properties. See other Frequency Offset properties.

Tip: To avoid unnecessary errors, first make other frequency offset settings. Then turn Frequency Offset ON.

### VB Syntax
```vb
object.FrequencyOffsetState = value
```

### Variable
**Type** - Description

- `object` Channel *(object)*
- `or`
  - CalSet *(object)* - Read-only property

### value
*(Enum as NaStates)* - Choose from:

- `naOFF` (0) - Turns Frequency Offset OFF
- `naON` (1) - Turns Frequency Offset ON

### Return Type
Enum

### Default
`naOFF` (0)

### Examples
- `chan.FrequencyOffsetState = naON` *Write*
- `Foffset = chan.FrequencyOffsetState` *Read*

### C++ Syntax
```cpp
HRESULT FrequencyOffsetState(tag NAStates *pState);
HRESULT FrequencyOffsetState(tag NAStates newState)
```

### Interface
- IChannel2
- |CalSet3
# Shape Property

**Description**  Specifies the shape of the gate filter.

**VB Syntax**  
\[ gat\_Shape = value \]

**Variable**  
- **Type**: A Gating (object)
- **Description**: 
  - `gat` - A Gating (object)
  - `value` - (enum NAGateShape) - Choose from:
    - 0 - naGateShapeMaximum
    - 1 - naGateShapeWide
    - 2 - naGateShapeNormal
    - 3 - naGateShapeMinimum

**Return Type**  NAGateShape

**Default**  2 - Normal

**Examples**  
- `gat\_Shape = naGateShapeMaximum`  'Write
- `filterShape = gat\_Shape`  'Read

**C++ Syntax**  
- HRESULT get\_Shape(tagNAGateShape *pVal)
- HRESULT put\_Shape(tagNAGateShape newVal)

**Interface**  IGating
### Type (gate) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies the type of gate filter used.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$gat.Type = value$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>$gat$</td>
<td>A Gating (object)</td>
</tr>
<tr>
<td>$value$</td>
<td>(enum NAGateType) - Choose from:</td>
</tr>
<tr>
<td>0 - naGateTypeBandpass</td>
<td>Includes (passes) the range between the start and stop times.</td>
</tr>
<tr>
<td>1 - naGateTypeNotch</td>
<td>Excludes (attenuates) the range between the start and stop times.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>NAGateType</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Bandpass</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>$gate.Type = naGateTypeNotch$ 'Write</td>
</tr>
<tr>
<td></td>
<td>$filterType = gate.Type$ 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Type(tagNAGateType *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Type(tagNAGateType newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGating</td>
</tr>
</tbody>
</table>
### GPIBAddress Property

**Description**
Sets and returns the PNA GPIB address bus.

**VB Syntax**
```vbnet
app.GPIBAddress (bus) = value
```

**Variable**
- **app**
  - An **Application** (object)
- **bus**
  - (Short Integer) GPIB bus. MUST be set to 0.
- **value**
  - (Short Integer) GPIB Address on the PNA. Choose a value between 0 and 30.

**Return Type**
Short Integer

**Default**
16

**Examples**
- `address=app.GPIBAddress(0)` 'Read
- `app.GPIBAddress(0)=16` 'Write

**C++ Syntax**
```cpp
HRESULT get_GPIBAddress(short busIndex, short* address);
HRESULT put_GPIBAddress(short busIndex,short address);
```

**Interface**
IApplication8
**GPIBMode Property**

**Description**
Changes the analyzer to a GPIB system controller or a talker/listener on the bus. The analyzer must be the controller if you want to use it to send commands to other instruments. The analyzer must be a talker/listener if you want to send it commands from another PC.

**Note:** This command has no effect in PNAs with dedicated Controller and Talker/Listener GPIB connectors. [Learn more.](#)

**VB Syntax**
```vbnet
app.GPIBMode value
```

**Variable**
<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>app</em></td>
<td>An <a href="#">Application</a> (object)</td>
</tr>
<tr>
<td><em>value</em></td>
<td>(enum NAGPIBMode) - Choose either:</td>
</tr>
<tr>
<td></td>
<td>0 - naTalkerListener - the analyzer is a talker / listener</td>
</tr>
<tr>
<td></td>
<td>1 - naSystemController - the analyzer is the system controller</td>
</tr>
</tbody>
</table>

**Return Type**
Long Integer

**Default**
0 - naTalkerListener

**Examples**
```vbnet
app.GPIBMode = naTalkerListener 'Write
```
```vbnet
mode = app.GPIBMode 'Read
```

**C++ Syntax**
```cpp
HRESULT get_GPIBMode(tagGPIBModeEnum* eGpibMode)
HRESULT put_GPIBMode(tagGPIBModeEnum eGpibMode)
```

**Interface**
IApplication
### GPIBPortCount Property

**Description**
Returns the number of GPIB ports that are present on the PNA rear-panel.

- **1.1 GHz CPU board** = 2
- All other CPU boards = 1

**VB Syntax**

```vbnet
value = cap.GPIBPortCount
```

**Variable**

- **(Type)** - Description
  - **value** (Long) - Variable to store the returned integer value of the number of GPIB ports.

- **cap** - A `Capabilities` object

**Return Type**

Long

**Default**

Not Applicable

**Examples**

```vbnet
value = cap.GPIBPortCount 'Read
```

**C++ Syntax**

```c++
HRESULT get_GPIBPortCount(long * gpibPorts);
```

**Interface**

ICapabilities3
HandshakeEnable Property

Description
Turns handshake ON / OFF.
To enable handshake, the main trigger enable must ALSO be set using Enable.
When ON, PNA acquisition waits indefinitely for the input line to be asserted before continuing with the acquisition.

VB Syntax
auxTrig.HandshakeEnable = state

Variable (Type) - Description

auxTrig An AuxTrigger (object)
state (boolean) -
True - Handshake enabled
False - Handshake NOT enabled

Return Type
Boolean

Default
False

Examples
auxTrig.HandshakeEnable = True 'Write
value = auxTrig.HandshakeEnable 'Read

C++ Syntax
HRESULT get_HandshakeEnable(VARIANT_BOOL * enable);
HRESULT put_HandshakeEnable(VARIANT_BOOL enable);

Interface
IAuxTrigger

Last Modified:
14-Dec-2006    MX New topic
**ID Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the test set ID number. For GPIB testsets, the ID is equivalent to the GPIB address. For testset I/O testsets, the ID is the base address of the testset (0 for the first testset, 1 for the second, and so on).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>value = tset.ID</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>value</td>
<td>(Long) variable to store the returned information.</td>
</tr>
</tbody>
</table>
| testsets(1) | A `TestsetControl` object.  
OR  
An `E5091Testset` object. |
| **Return Type** | Long Integer |
| **Default** | Not Applicable |
| **Examples** | value = testset1.ID |
| | See E5091A Example Program  
See External Testset Program |
| **C++ Syntax** | HRESULT get_ID(long *idNumber); |
| **Interface** | ITestsetControl  
IE5091Testset |
### IDString Property

**Description**
Returns the ID of the analyzer, including the Model number, Serial Number, and the Software revision number.

**Note:** Beginning with Rev 6.01, this command now returns the software revision with 6 digits instead of 4. For example, A.06.01.02.

**VB Syntax**

```vbnet
value = app.IDString
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>value</td>
<td>(string) - variable to contain the returned ID string</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Not Applicable

**Examples**

```vbnet
id = app.IDString
```

**C++ Syntax**

```cpp
HRESULT IDString(BSTR* IDString)
```

**Interface**
IApplication
**IFBandwidthOption Property**

**Description**
Enables the IFBandwidth to be set on individual sweep segments. This property must be set True before `seg.IFBandwidth = value` is sent. Otherwise, this command will be ignored.

**VB Syntax**

```vbnet
segs.IFBandwidthOption = value
```

**Variable (Type) - Description**

- `segs` (A Segments collection `object`)
- `value` (boolean)
  - **True** - Enables variable IFBandwidth setting for segment sweep
  - **False** - Disables variable IFBandwidth setting for segment sweep

**Return Type**
Boolean

**Default**
False

**Examples**

```vbnet
segs.IFBandwidthOption = True 'Write
IFOption = IFBandwidthOption 'Read
```

**C++ Syntax**

```cpp
HRESULT get_IFBandwidthOption(VARIANT_BOOL *pVal)
HRESULT put_IFBandwidthOption(VARIANT_BOOL newVal)
```

**Interface**
ISegments
**Write/Read**

**IF Bandwidth Property**

**Description**
Sets or returns the IF Bandwidth of the channel.
Sets or returns the IF Bandwidth of the segment.
Returns the IF Bandwidth used in the Cal Set

**VB Syntax**

```vbnet
object.IFBandwidth = value
```

**Variable**

*object* - Channel (object)
   
or
   Segment (object)
   
or
   CalSet (object) - Read-only property

*value* - (double) - IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. ([Click to see the lists.](#)) If an invalid number is specified, the analyzer will round up to the closest valid number.

**Return Type**
Double

**Default**
See [Preset IFBW](#) for your PNA model.

**Examples**

```vbnet
chan.IFBandwidth = 3e3 'sets the IF Bandwidth of for the channel object to 3 kHz. -Write
seg.IFBandwidth = 5 'sets the IF Bandwidth of the segment to 5 Hz. -Write

ifbw = chan.IFBandwidth  -Read
```

**C++ Syntax**

```c++
HRESULT get_IFBandwidth(double *pVal);
HRESULT put_IFBandwidth(double newVal);
```

**Interface**

IChannel
ISegment
ICalSet3
### IFDenominator Property

**Description**
Sets or returns the denominator value of the IF Fractional Multiplier.
Only applies to 2 stage mixers.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**

```
mixer.IFDenominator = value
```

**Variable (Type) - Description**

- `mixer` A Mixer (object)
- `value` (long) IF Denominator value.

**Return Type** Long

**Default** 1

**Examples**

Print mixer.IFDenominator 'prints the value of the IFDenominator

**C++ Syntax**

```
HRESULT get_IFDenominator(long *pVal)
HRESULT put_IFDenominator(long newVal)
```

**Interface** IMixer

---

Last Modified:

10-Mar-2008   MX New topic
**IFFilterSampleCount Property**

**Description**
Note: This setting applies only to the E836X Opt. H11.

Sets or retrieves the number of taps in the IF filter when the `IFFilterSource` property is set to `naIFFilterSourceManual`.

**VB Syntax**

```vbnet
IfConfig.IFFilterSampleCount = value
```

**Variable**

*Type* - Description

- `IfConfig` *(object)*

- `value` *(long)* – The IF filter sample count. The minimum and maximum allowed values for this property can vary by model number, but can be queried using the `MinimumIFFilterSampleCount` and `MaximumIFFilterSampleCount` properties.

**Return Type**

Long Integer

**Default**

PNA Model number dependent

**Examples**

```vbnet
App.ActiveChannel.IFConfiguration.IFFilterSampleCount = 200 'Write
variable = App.ActiveChannel.IFConfiguration.IFFilterSampleCount 'Read
```

See an example program

**C++ Syntax**

```csharp
HRESULT get_IFFilterSampleCount( long * pSampleCount );
HRESULT put_IFFilterSampleCount( long sampleCount );
```

**Interface**

IIFConfiguration2
Description

Note: This setting applies only to the E836X Opt. H11.

Sets or returns the IF filter sample period time. This time is only used if the IFFilterSamplePeriodMode is set to naManual.

VB Syntax

IfConfig.IFFilterSamplePeriod = value

Variable (Type) - Description

IfConfig (object)

value (double) – The sample period time in seconds. Valid sample period times can be queried using the IFFilterSamplePeriodList property.

Return Type

Double

Default

PNA Model number dependent.

Examples

App.ActiveChannel.IFConfiguration.IFFilterSamplePeriod = .000006 'Write

variable = App.ActiveChannel.IFConfiguration.IFFilterSamplePeriod 'Read

See an example program

C++ Syntax

HRESULT get_IFFilterSamplePeriod( double * pSamplePeriod );

HRESULT put_IFFilterSamplePeriod( double samplePeriod );

Interface

IIFConfiguration2
**Description**  
Retrieves the list of available IF filter sample periods for the instrument.  
*Note:* This setting applies only to the E836X Opt. H11.

**VB Syntax**  
`variable = IfConfig.IFFilterSamplePeriodList`

**Variable** *(Type)* - Description

*variable* *(Array)* An array of permissible values that can be passed to the `IFFilterSamplePeriod` property.

*IfConfig*  
`IFConfiguration (object)`

**Return Type**  
Array

**Default**  
Not applicable

**Examples**

```vbnet
Dim Variable
Variable = App.ActiveChannel.IFConfiguration.IFFilterSamplePeriodList 'Read
MsgBox "First IF Sample Period Value: " & Variable(0)
```

See an example program

**C++ Syntax**  
`HRESULT get_IFFilterSamplePeriodList( SAFEARRAY ** ppSamplePeriodList );`

**Interface**  
`IIFConfiguration2`
## IFFilterSamplePeriodMode Property

### Description
Sets or returns the IF filter sample period mode.

**Note:** This setting applies only to the E836X Opt. H11.

### VB Syntax
```
IfConfig.IFFilterSamplePeriodMode = value
```

### Variable (Type) - Description

**IfConfig**  
*IFConfiguration (object)*

**value**  
*(enum NAModes)* -
- 0 - **naAuto** - IF filter sample period is chosen automatically.
- 1 - **naManual** - the IF filter sample period is the value specified by the IFSamplePeriod property.

### Return Type
NAModes

### Default
0 - naAuto

### Examples
```
App.ActiveChannel.IFConfiguration.IFFilterSamplePeriodMode = naAuto
'Write

variable = App.ActiveChannel.IFConfiguration.IFFilterSamplePeriodMode
'Read
```

### See an example program

### C++ Syntax
```
HRESULT get_IFFilterSamplePeriodMode( tagNAModes * pMode);
HRESULT put_IFFilterSamplePeriodMode ( tagNAModes mode );
```

### Interface
IIFConfiguration2
### IFFilterSource Property

**Description**
Sets or retrieves type of IF filter to be used.

*Note:* This setting applies only to the E836X Opt. H11.

**VB Syntax**

```vbnet
IfConfig.IFFilterSource = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>IFConfiguration (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(NAIFFilterSource) – The type of filter to use. Choose from:</td>
</tr>
<tr>
<td></td>
<td>naIFFilterSourceAuto – The IF filter type is automatically chosen.</td>
</tr>
<tr>
<td></td>
<td>naIFFilterSourceManual – The IF filter is a predetermined shape where the IFFilterSampleCount determines the number of taps in the filter.</td>
</tr>
</tbody>
</table>

**Return Type**

NAIFFilterSource

**Default**

naIFFilterSourceAuto

**Examples**

```vbnet
App.ActiveChannel.IFConfiguration.IFFilterSource = naIFFilterSourceManual 'Write
Variable = App.ActiveChannel.IFConfiguration.IFFilterSource 'Read
```

See an example program

**C++ Syntax**

```cpp
HRESULT get_IFFilterSource( tagNAIFFilterSource * pFilterSource );
HRESULT put_IFFilterSource(tagNAIFFilterSource filterSource );
```

**Interface**

IIFConfiguration2
Write/Read

IFGainLevel Property

Description

**Note:** This setting applies only to the E836X Opt. H11.
Manually sets the gain level for the specified receiver.

VB Syntax

```
IfConfig.IFGainLevel (id) = value
```

Variable

**Type** - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IfConfig</td>
<td>IFCoffiguration (object)</td>
<td>id</td>
</tr>
</tbody>
</table>
|          |                | Receiver for which to set the gain level. Choose from: 'A', 'B', 'R1', 'R2'.
|          |                | **Note:** The A and R1 receivers are always switched together. B and R2 are also always switched together. For example, if you specify 'A', R1 will also be switched. |

<table>
<thead>
<tr>
<th>Variable</th>
<th>(long Integer) Gain Level. Choose from:</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>0 - Lowest gain setting</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2 - Highest gain setting</td>
</tr>
</tbody>
</table>

Return Type

Long Integer

Default

0 (Lowest setting)

Examples

```
IfConfig.IFGainLevel("A") = 1
```

C++ Syntax

```
HRESULT get_IFGainLevel (BSTR IDString, *GainLevel)
HRESULT put_IFGainLevel (BSTR IDString, GainLevel)
```

Interface

IFConfiguration

Last Modified:

21-Sep-2007   Fixed example
IFGainMode Property

**Description**

Note: This setting applies only to the E836X Opt. H11.

Sets the gain state for ALL receivers to Auto or Manual.

**VB Syntax**

```
IfConfig.IFGainMode (id) = value
```

**Variable (Type) - Description**

*IfConfig* An *IFConfiguration* (object)

*id* Receivers for which to set the state. Choose 'ALL'.

*value* (enum as NAModes)

Choose from:

- 0 - naAUTO
- 1 - naMANUAL (use IfGainLevel Property to manually set gain level)

**Return Type**

NAModes

**Default**

0 - naAUTO

**Examples**

```
IfConfig.IFGainMode("ALL") = naAUTO
```

**C++ Syntax**

HRESULT get_IFGainMode (BSTR IDString, NAModes *gainMode)

HRESULT put_IFGainMode (BSTR IDString, NAModes gainMode)

**Interface**

IIFConfiguration
IFGateEnable Property

**Description**
Sets or retrieves the state of the IF Gate.

**Note:** This setting applies only to the E836X Opt. H11.

**VB Syntax**

```vbnet
IfConfig.IFGateEnable = value
```

**Variable**

(**Type** - Description)

- **IfConfig** - IFConfiguration (object)

**value** (**Boolean**) – The state of the IF Gate.

- **True** – The IF Gate is in use.
- **False** – The IF Gate is not in use.

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
App.ActiveChannel.IFConfiguration.IFGateEnable = True 'Write

variable = App.ActiveChannel.IFConfiguration.IFGateEnable 'Read
```

See an example program

**C++ Syntax**

```c
HRESULT get_IFGateEnable( VARIANT_BOOL * pGateEnabled );

HRESULT put_IFGateEnable( VARIANT_BOOL gateEnabled );
```

**Interface**

IIFConfiguration
**IFFrequency Property**

**Description**
Sets and returns the IF frequency for ALL receiver paths being used for the specified channel. To set this frequency, *IFFrequencyMode Property* must be set to OFF (Manual).

**VB Syntax**
```vbnet
IfConfig.IFFrequency = value
```

**Variable**
**Type** - Description

*IfConfig*  
An *IFConfiguration* *(object)*

*value*  
(double) IF Frequency. Use *MaximumIFFrequency* and *MinimumIFFrequency* to determine the range of value for this command.

**Return Type**
Double

**Default**
9 MHz

**Examples**
```
IfConfig.IFFrequency = 9.3e6
```

**C++ Syntax**
```cpp
HRESULT get_IFFrequency (double *pVal);
HRESULT put_IFFrequency (double pVal);
```

**Interface**
*IFConfiguration3*

---

Last Modified:

18-Jun-2007  MX New topic
Write/Read

IFFrequencyMode Property

**Description**
Sets and returns method for specifying the way the IF Frequency is determined.

**VB Syntax**
`IfConfig.IFFrequencyMode = value`

**Variable**
*(Type)* - Description

<table>
<thead>
<tr>
<th>IfConfig</th>
<th>An IFConfiguration (object)</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(enum as NAModes) IF Frequency mode. Choose from:</td>
</tr>
<tr>
<td>0 - naAUTO</td>
<td>PNA determines the setting for the IF frequency.</td>
</tr>
<tr>
<td>1 - naMANUAL</td>
<td>(use IFFrequency Property to manually set frequency.</td>
</tr>
</tbody>
</table>

**Return Type**
Enum

**Default**
0 naAuto

**Examples**
`IfConfig.IFFrequencyMode = naMANUAL`

**C++ Syntax**
`HRESULT get_IFFrequencyMode (tagNAModes* pdspMode);`
`HRESULT put_IFFrequencyMode (tagNAModes* pdspMode);`

**Interface**
IFConfiguration3

---

Last Modified:

18-Jun-2007 MX New topic
IFNumerator Property

**Description**
Sets or returns the numerator value of the IF Fractional Multiplier. Only applies to 2 stage mixers.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**

```
mixer.IFNumerator = value
```

**Variable**

- **mixer** (Type) - Description - A Mixer (object)
- **value** (Long)

**Return Type**

long

**Default**

Not Applicable

**Examples**

Print mixer.IFNumerator 'prints the value of the IFNumerator

**C++ Syntax**

```
HRESULT get_IFNumerator(long *pVal)
HRESULT put_IFNumerator(long newVal)
```

**Interface**

IMixer

---

Last Modified:

4-Mar-2008   Added note.
### IFSideband Property

**Description**
Sets or returns the value of the IF sideband, high or low. Only applies to 2 stage mixers.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**

```vbnet
mixer.IFSideband = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mixer</code></td>
<td>A Mixer (object)</td>
</tr>
</tbody>
</table>
| `value`        | (enum as FCASideBand) - Choose from:
|                | 0 - LOW |
|                | 1 - HIGH |

**Return Type**
FCASideBand

**Default**
0 - LOW

**Examples**

```vbnet
Print mixer.IFSideband 'prints the value of the IFSideband
```

**C++ Syntax**

```c++
HRESULT get_IFSideband(FCASideBand *pVal)
HRESULT put_IFSideband(FCASideBand newVal)
```

**Interface**
IMixer

---

Last Modified:

4-Mar-2008   Added note.
**Description**  
*Note:* This setting applies only to the E836X Opt. H11.

Sets the source path of the specified receiver. An error is returned if `<id>` is not valid, or if `<value>` is not valid for the specified `<id>`.

**VB Syntax**  
`IfConfig.IFSourcePath (id) = value`

**Variable**  
*Type* - Description

`IfConfig` An `IFConfiguration` *(object)*

*id* Receiver for which to set the gain level. Choose from: 'A', 'B', 'R1', 'R2'.

*Note:* The A and R1 receivers are always switched together. B and R2 are also always switched together. For example, if you specify "A", R1 will also be switched.

*value* *(Enum as NAIFSourcePath)* Source path. Choose from:

0 - `naNormalIFPath` - the PNA decides the appropriate IF input paths.
1 - `naExternalIFPath` - always use the rear panel IF inputs.

**Return Type**  
`NAIFSourcePath`

**Default**  
0 - `naNormalIFPath`

**Examples**  
`IfConfig.IFSourcePath('A') = naNormalIFPath`

**C++ Syntax**  
`HRESULT get_IFSourcePath (BSTR IDString, NAIFSourcePath *IFSourcePath)`
`HRESULT put_IFSourcePath (BSTR IDString, NAIFSourcePath IFSourcePath)`

**Interface**  
`IIFConfiguration`
**IFStartFrequency Property**

**Description**
Sets or returns the start frequency value of the mixer IF frequency.

*Only applies to 2 stage mixers.*

If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**

```
mixer.IFStartFrequency = value
```

**Variable (Type) - Description**

- `mixer` A Mixer *(object)*
- `value` *(double)* - Frequency in Hertz.

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```
Print mixer.IFStartFrequency 'prints the value of the IFStartFrequency
```

**C++ Syntax**

```
HRESULT get_IFStartFrequency(double *pVal)
HRESULT put_IFStartFrequency(double newVal)
```

**Interface**
IMixer

---

Last Modified:

4-Mar-2008  Added note.
IFStopFrequency Property

Description
Sets or returns the stop frequency value of the mixer IF frequency. Only applies to 2 stage mixers.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

VB Syntax

```
mixer.IFStopFrequency = value
```

Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mixer</td>
<td>A Mixer (object)</td>
</tr>
<tr>
<td>value</td>
<td>(double) - IF stop frequency in Hertz.</td>
</tr>
</tbody>
</table>

Return Type

Double

Default

Not Applicable

Examples

```
Print mixer.IFStopFrequency 'prints the value of the IFStopFrequency
```

C++ Syntax

```
HRESULT get_IFStopFrequency(double *pVal)
HRESULT put_IFStopFrequency(double newVal)
```

Interface

IMixer

Last Modified:

4-Mar-2008   Added note.
ImpedanceStates Method

**Description**
Sets the number of impedance states to use during calibrated measurements.

**VB Syntax**
```vbnet
noise.ImpedanceStates = value
```

**Variable**

- **Type**: NoiseFigure (object)
- **Description**:
  
**Value**

- **Type**: double
- **Description**: Impedance states. Choose between 4 and the maximum number allowed by the noise tuner device. If the specified number exceeds the capability of the device, the measurement will use the maximum number of states the device allows.

**Return Type**
Double

**Default**
4

**Examples**
```vbnet
noise.ImpedanceStates = 10 'Write
AvgNoise = noise.ImpedanceStates 'Read
```

**C++ Syntax**
```cpp
HRESULT get_ImpedanceStates(double* pVal)
HRESULT put_ImpedanceStates(double newVal)
```

**Interface**
INoiseFigure

---

Last Modified:
29-May-2007   MN New topic
ImpulseWidth Property

**Description**
Sets or returns the Impulse Width of Time Domain transform windows

**VB Syntax**
```
trans.ImpulseWidth = value
```

**Variable**

- `trans` (Type) - Description
  - A **Transform** (object)

- `value` (double) - Impulse Width in seconds. Range of settings depends on the frequency range of your analyzer.

**Return Type**
Double

**Default**
.98 / Default Span

**Examples**
```
trans.ImpulseWidth = 200e-12 'sets the Impulse width of a transform window -Write
IW = trans.ImpulseWidth 'Read
```

**C++ Syntax**
```
HRESULT get_ImpulseWidth(double *pVal)
HRESULT put_ImpulseWidth(double newVal)
```

**Interface**
ITransform
**IndexState Property**

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Determines the control of Material Handler connector Pin 20.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>handler.IndexState = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - <em>Description</em></td>
</tr>
<tr>
<td><code>handler</code></td>
<td><em>(object)</em> - A <a href="#">Handler I/O</a> object</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(boolean)</em></td>
</tr>
<tr>
<td></td>
<td>False - Pin 20 is controlled by Output Port B6</td>
</tr>
<tr>
<td></td>
<td>True - Pin 20 is controlled by the Index signal</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>handler.IndexState = False 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>bState = handler.IndexState 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_IndexState (BOOL *pVal);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_IndexState (BOOL newVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IHWMaterialHandlerI02</td>
</tr>
</tbody>
</table>

### About the Handler I/O Connector

- **Read/Write**
- **IndexState Property**

---

[Handler I/O](#): A component of the Handler I/O library.
### InputA Property - Obsolete

**Description**
This property has NO replacement and no longer works correctly. (Sept. 2004)
Sets a Port Extension value for Receiver A

**VB Syntax**

```vbnet
portExt.InputA = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>portExt</code></td>
<td>A Port Extension (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) - Port Extension value in seconds. Choose any number between -10 and 10</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
0

**Examples**

```vbnet
portExt.InputA = 10e-6 'Write
inA = portExt.InputA 'Read
```

**C++ Syntax**

```cpp
HRESULT get_InputA(double *pVal)
HRESULT put_InputA(double newVal)
```

**Interface**
IPortExtension
### InputB Property - Obsolete

**Description**  
This property has NO replacement and no longer works correctly. (Sept. 2004)

Sets the Port Extension value for Receiver B

**VB Syntax**  
`portExt.InputB = value`

**Variable**  
* **Type** - Description  
  - `portExt`  
    - A Port Extension *(object)*
  - `value`  
    - *(double)* - Port Extension value in seconds. Choose any number between -10 and 10

**Return Type**  
Double

**Default**  
0

**Examples**  
- `portExt.InputB = 10e-6` *Write*
- `inB = portExt.InputB` *Read*

**C++ Syntax**  

HRESULT get_InputB(double *pVal)  
HRESULT put_InputB(double newVal)

**Interface**  
IPortExtension
## InputC Property  **Obsolete**

| **Description** | This property has NO replacement and no longer works correctly. (Sept. 2004)  
Sets the Port Extension value for Receiver C |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>portExt.InputC = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>portExt</code></td>
<td>A Port Extension <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(double)</em> - Port Extension value in seconds. Choose any number between -10 and 10</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>portExt.InputC = 10e-6</code> <em>(Write)</em></td>
</tr>
<tr>
<td></td>
<td><code>inC = portExt.InputC</code> <em>(Read)</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_InputC(double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_InputC(double newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPortExtension</td>
</tr>
</tbody>
</table>
### InputDenominator Property

**Description**
Sets or returns the denominator value of the Input Fractional Multiplier.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**

```vbnet
mixer.InputDenominator = value
```

**Variable**

- **mixer**
  A Mixer  *(object)*

- **value**
  *(Long)* - Input denominator value.

**Return Type**

Long

**Default**

1

**Examples**

```vbnet
Print mixer.InputDenominator 'prints the value of the InputDenominator
```

**C++ Syntax**

```cpp
HRESULT get_InputDenominator(long *pVal)
HRESULT put_InputDenominator(long newVal)
```

**Interface**

IMixer

---

**Last Modified:**

4-Mar-2008   Added note.
**Write/Read**

**About Mixer Configuration**

---

**InputFixedFrequency Property**

**Description**
Sets or returns the mixer fixed Input frequency value.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**

```vbnet
mixer.InputFixedFrequency = value
```

**Variable**

- **Type**: Description

  - `mixer` A **Mixer** (object)
  
  - `value` (double) - Input Fixed Frequency in Hertz.

**Return Type**

- Double

**Default**

- Not Applicable

**Examples**

```vbnet
mixer.InputFixedFrequency = 1e9
```

**C++ Syntax**

```cpp
HRESULT get_InputFixedFrequency(double *pVal)
HRESULT put_InputFixedFrequency(double newVal)
```

**Interface**

- IMixer6

---

**Last Modified:**

4-Mar-2008    Added note.

---

1087
IsInputGreaterThanLO Property

**Description**
Specifies whether to use the Input frequency that is greater than the LO or less than the LO. To learn more, see the mixer setup dialog box help.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**
```vbnet
mixer.IsInputGreaterThanLO (LO) = bool
```

**Variable**

- **(Type)** - Description
- `mixer` - A Mixer (object)
- `LO` (Integer) - LO stage number
  - Choose from 1 (default) or 2
- `bool` (Boolean) -
  - **True** - Use the Input that is Greater than the specified LO.
  - **False** - Use the Input that is Less than the specified LO.

**Return Type**
Boolean

**Default**
True

**Examples**

```vbnet
mixer.IsInputGreaterThanLO(1) = True
```

**C++ Syntax**

```cpp
HRESULT get_IsInputGreaterThanLO(VARIANT_BOOL * val);
HRESULT put_IsInputGreaterThanLO(VARIANT_BOOL val);
```

**Interface**
IMixer2

---

**Last Modified:**
4-Mar-2008   Added note.
**InputNumerator Property**

**Description**  
Sets or returns the numerator value of the Input Fractional Multiplier.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**  
`mixer.InputNumerator = value`

**Variable**  
**Type** - **Description**

- `mixer` A Mixer **(object)**
- `value` **(Long)** - Input numerator value.

**Return Type**  
Long

**Default**  
1

**Examples**

```
Print mixer.InputNumerator 'prints the value of the InputNumerator
```

**C++ Syntax**

```
HRESULT get_InputNumerator(long *pVal)
HRESULT put_InputNumerator(long newVal)
```

**Interface**

IMixer

---

**Last Modified:**

4-Mar-2008  Added note.
InputPower Property

**Description**
Sets or returns the value of the Input Power.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**
mixer.InputPower = value

**Variable**
(Type) - Description

mixer  A Mixer (object)

value  (double) - Input power in dBm.

**Return Type**
Double

**Default**
-17dBm

**Examples**
Print mixer.InputPower  'prints the value of the InputPower

**C++ Syntax**
HRESULT get_InputPower(double *pVal)
HRESULT put_InputPower(double newVal)

**Interface**
IMixer

Last Modified:
4-Mar-2008    Added note.
## InputRangeMode Property

**Description**  
Sets or returns the Input sweep mode.  
If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**  
`mixer.InputRangeMode = value`

**Variable (Type) - Description**

- `mixer` A **Mixer** (object)
- `value` (Enum as **MixerRangeMode**) - Input sweep mode. Choose from:
  - 0 - **mixSwept**
  - 1 - **mixFixed**

**Return Type**  
Enum

**Default**

- 0 - mixSwept

**Examples**

```vbnet
mixer.InputRangeMode = mixSwept
```

**C++ Syntax**

```cpp
HRESULT get_InputRangeMode(long *pVal)
HRESULT put_InputRangeMode(long newVal)
```

**Interface**

*IMixer6*

---

**Last Modified:**

4-Mar-2008  
Added note.
InputStartFrequency Property

**Description**
Sets and returns the start frequency value of the mixer Input frequency.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**
mixer.InputStartFrequency = value

**Variable**
(Type) - Description

- mixer A Mixer (object)
- value (double) - Input start frequency in Hertz.

**Return Type**
Double

**Default**
Start frequency of the PNA

**Examples**
mixer.InputStartFrequency = Start_Freq

**C++ Syntax**
HRESULT get_InputStartFrequency(double *pVal)
HRESULT put_InputStartFrequency(double newVal)

**Interface**
IMixer

---

Last Modified:

4-Mar-2008    Added note.
### InputStopFrequency Property

**Description**
Sets and returns the stop frequency value of the mixer Input frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

**VB Syntax**
```vbnet
mixer.InputStopFrequency = value
```

**Variable**
- **mixer** (Type) - A Mixer
- **value** (Type) - Input stop frequency in Hertz.

**Return Type**
Double

**Default**
Stop frequency of the PNA

**Examples**
```vbnet
mixer.InputStopFrequency = Stop_Freq
```

**C++ Syntax**
```cpp
HRESULT get_InputStopFrequency(double *pVal)
HRESULT put_InputStopFrequency(double newVal)
```

**Interface**
IMixer

---

**Last Modified:**
4-Mar-2008   Added note.
InternalTestsetPortCount Property

Description
Returns the number of PNA test ports. This does not include the ports on an external test set.

VB Syntax
value = cap.InternalTestsetPortCount

Variable (Type) - Description

value (Long) - Variable to store the returned number of PNA test ports.

cap A Capabilities (object)

Return Type
Long

Default
Not Applicable

Examples
value = cap.InternalTestsetPortCount 'Read

C++ Syntax
HRESULT get_InternalTestsetPortCount(long *numPorts);

Interface
ICapabilities
Interpolate Correction Property

**Description**

Turns ON and OFF correction interpolation which calculates new error terms when stimulus values change after calibration.

When this property is ON and error correction is being applied, the calibration subsystem attempts to interpolate the error terms whenever the stimulus parameters are changed.

When this property is OFF under the same circumstances, error correction is turned OFF.

**VB Syntax**

```
meas.InterpolateCorrection = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>value</td>
<td>(boolean) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>True - Turns correction interpolation ON</td>
</tr>
<tr>
<td></td>
<td>False - Turns correction interpolation OFF</td>
</tr>
</tbody>
</table>

**Return Type**

Boolean

**Default**

True

**Examples**

```
meas.InterpolateCorrection = False
```

```
calInterpolate = InterpolateCorrection 'Read
```

**C++ Syntax**

HRESULT get_InterpolateCorrection(boolean *pVal)

HRESULT put_InterpolateCorrection(boolean newVal)

**Interface**

IMeasurement
## Interpolated Property

**Description**

Turns marker Interpolation ON and OFF. Marker interpolation enables X-axis resolution beyond the discrete data values. The analyzer will calculate the x and y-axis data values between discrete data points. Use `meas.Interpolate` to change interpolation of all markers in a measurement. This command will override the measurement setting.

**VB Syntax**

```vbnet
mark.Interpolated = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mark</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td>True</td>
<td>- Turns interpolation ON</td>
</tr>
<tr>
<td>False</td>
<td>- Turns interpolation OFF</td>
</tr>
</tbody>
</table>

**Return Type**

Boolean

**Default**

True

**Examples**

```vbnet
mark.Interpolated = True 'Write
```

```vbnet
interpolate = mark.Interpolated 'Read
```

**C++ Syntax**

```c++
HRESULT get_Interpolated(VARIANT_BOOL *pVal)
HRESULT put_Interpolated(VARIANT_BOOL newVal)
```

**Interface**

IMarker
InterpolateNormalization Property  Superseded

Description  Note: This property is replaced by DoReceiverPowerCal Method.

Turns ON and OFF normalization interpolation which calculates new divisor data when stimulus values change after normalization.

When this property is ON and normalization is being applied, the Normalization algorithm attempts to interpolate the divisor data whenever the stimulus parameters are changed.

When this property is OFF under the same circumstances, normalization is turned OFF.

Normalization is currently supported only on measurements of unratioed power for the purpose of performing a receiver power calibration.

VB Syntax  \textit{meas.InterpolateNormalization} = \textit{value}

Variable  \textbf{(Type)} - \textbf{Description}

\textit{meas}  \textbf{(object)} - A Measurement object

\textit{value}  \textbf{(boolean)}

False – Turns normalization interpolation OFF
True – Turns normalization interpolation ON

Return Type  Boolean

Default  False -OFF

Examples  \textit{meas.InterpolateNormalization} = \textit{False} \ 'Write

\textit{normalized} = \textit{meas.InterpolateNormalization} \ 'Read

C++ Syntax  HRESULT put\_InterpolateNormalization(VARIANT\_BOOL bState);
HRESULT get\_InterpolateNormalization(VARIANT\_BOOL *bState);

Interface  IMeasurement
### Interrupt Property

**Description**  
Reads the boolean that represents the state of the Interrupt In line (pin 13) on the external test set connector.

**VB Syntax**  
`value = ExtIO.Interrupt`

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>boolean</td>
<td>Variable to store the returned data</td>
</tr>
<tr>
<td><code>ExtIO</code></td>
<td>object</td>
<td>An <code>ExternalTestSetIO</code> object</td>
</tr>
</tbody>
</table>

**Return Type**  
Boolean

- **False** - indicates the line is being held at a TTL High
- **True** - indicates the line is being held at a TTL Low

**Default**  
Not Applicable

**Examples**

```vbnet
value = ExtIO.Interrupt
```

**C++ Syntax**

```cpp
HRESULT get_Interrupt(VARIANT_BOOL* bValue);
```

**Interface**  
`IHWExternalTestSetIO`
## IsContinuous Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Returns whether or not a channel is in continuous mode. To set the channel to continuous mode, use Continuous Method.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = chan.IsContinuous</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(boolean)</em> - Choose either:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Channel trigger is NOT set to continuous.</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Channel trigger IS set to continuous.</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>Channel <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>trig = chan.IsContinuous 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_IsContinuous (VARIANT_BOOL *bContinuous);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel3</td>
</tr>
</tbody>
</table>
IsECALModuleFoundEx Property

**Description**
This property replaces *IsECALModuleFound Property*. Returns true or false depending on whether communication was established between the PNA and the specified ECal module.

**VB Syntax**
```vbnet
modFound = cal.IsECALModuleFoundEx(module)
```

**Variable**
- **Type** - Description
  - `modFound` (boolean) - Variable to store the returned test result.
    - **True** - The PNA identified the presence of the specified ECal module.
    - **False** - The PNA did NOT identify the presence of the specified ECal module.
  - `cal` (object) - A Calibrator object
  - `module` (long integer) - ECal module. Choose from modules 1 through 8
    Use *GetECALModuleInfoEx* to return the model and serial number of each module.

**Return Type**
Boolean

**Default**
Not applicable

**Examples**
```vbnet
Set cal = pna.ActiveChannel.Calibrator
moduleFound = cal.IsECALModuleFoundEx(1)
```

**C++ Syntax**
```cpp
HRESULT get_IsECALModuleFoundEx(long moduleNumber, VARIANT_BOOL *bModuleFound);
```

**Interface**
ICalibrator4
IsFrequencyOffsetPresent Property

**Description**  Returns a value indicating the presence of Frequency Offset Option 080 in the remote PNA.

**VB Syntax**  

```vbnet
value = cap.IsFrequencyOffsetPresent
```

**Variable**  

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| value | (Boolean) - Variable to store the returned value  
| | True - Frequency Offset Option 080 is present  
| | False - Frequency Offset Option 080 is not present |

**cap**  A **Capabilities** (object)

**Return Type**  Boolean

**Default**  Not Applicable

**Examples**  

```vbnet
value = cap.isFrequencyOffsetPresent(1) 'Read
```

**C++ Syntax**  

```cpp
HRESULT get_IsFrequencyOffsetPresent(VARIANT_BOOL * present);
```

**Interface**  ICapabilities
### IsHold Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns whether or not a channel is in hold mode. To set the channel to hold mode, use Hold Method.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>value = chan.IsHold</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>value</td>
<td>(boolean) - Choose either:</td>
</tr>
<tr>
<td>False</td>
<td>Channel trigger is NOT set to hold.</td>
</tr>
<tr>
<td>True</td>
<td>Channel trigger IS set to hold.</td>
</tr>
<tr>
<td>chan</td>
<td>Channel (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>trig = chan.IsHold 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_IsHold ( VARIANT_BOOL *bHold);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel3</td>
</tr>
</tbody>
</table>
## IsMarkerOn Property

**Description**
Returns whether or not a marker was used for the specified tuning sweep.

**VB Syntax**

```vbnet
value = embedLODiag.IsMarkerOn (n)
```

**Variable**

**(Type) - Description**

- `value` (Boolean) Variable to store the returned data.

- `embedLODiag` An `EmbeddedLODiagnostic` (object)

- `n` Tuning sweep number. Use `NumberOfSweeps` to find the number of sweeps taken.

**Return Type**

(String)

**Default**
Not Applicable

**Examples**

```vbnet
data= embedLO.IsMarkerOn 3 'read
```

**C++ Syntax**

```cpp
HRESULT IsMarkerOn(long sweep, VARIANT_BOOL* markerOn);
```

**Interface**

IEmbeddedLODiagnostic

---

Last Modified:

13-Apr-2007    MX New topic
**IsolationIncrementAveraging Property**

**Description**
Value by which to increment (increase) the channel's averaging factor during measurement of isolation in an ECal calibration.

*Note:* If `<value>` is greater than 1 and the channel currently has averaging turned OFF, averaging will be turned ON only during the isolation measurements and with the averaging factor equal to `<value>`.

**VB Syntax**
```
cal.IsolationIncrementAveraging = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cal</code></td>
<td>A <code>Calibrator</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Incremental Averaging factor. The maximum averaging factor is 65536 ($2^{16}$).</td>
</tr>
</tbody>
</table>

**Return Type**
Long Integer

**Default**
8

**Examples**
```
oCal.IsolationAveragingIncrement = 16 'Write
```
```
avgIncr = oCal.IsolationAveragingIncrement ' Read
```

**C++ Syntax**
```
HRESULT get_IsolationAveragingIncrement(long *pVal);
HRESULT put_IsolationAveragingIncrement(long newVal);
```

**Interface**
ICalibrator7

---

**Last Modified:**
16-Apr-2007 MX New topic
### IsOn Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the ON</th>
<th>OFF state of Embedded LO.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>embedLO.IsOn = value</code></td>
<td></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
<td></td>
</tr>
<tr>
<td><code>embedLO</code></td>
<td>An <code>EmbeddedLO</code> (object)</td>
<td></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean)</td>
<td></td>
</tr>
<tr>
<td><code>False</code></td>
<td>- Turns Embedded LO OFF</td>
<td></td>
</tr>
<tr>
<td><code>True</code></td>
<td>- Turns Embedded LO ON</td>
<td></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>(Boolean)</td>
<td></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False (OFF)</td>
<td></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>embedLO.IsOn = True</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>data= embedLO.IsOn</code></td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_IsOn( VARIANT_BOOL* IsOn);</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HRESULT put_IsOn( VARIANT_BOOL IsOn);</td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEmbededLO</td>
<td></td>
</tr>
</tbody>
</table>

Last Modified:

13-Apr-2007   MX New topic
IsReceiverStepAttenuatorPresent Property

Description
Returns a value indicating the presence of Receiver step attenuators in the remote PNA.

VB Syntax
value = cap.IsReceiverStepAttenuatorPresent(n)

Variable
(Type) - Description
value (Boolean) - Variable to store the returned value.
True - Receiver step attenuators are present.
False - Receiver step attenuators are not present.

cap A Capabilities (object)

n (Long) - port number to query for step attenuators

Return Type
Boolean

Default
Not Applicable

Examples
value = cap.IsReceiverStepAttenuatorPresent(1) 'Read

C++ Syntax
HRESULT get_IsReceiverStepAttenuatorPresent(long portNumber, VARIANT_BOOL * present);

Interface
ICapabilities
### IsReferenceBypassSwitchPresent Property

**Description**
Returns a value indicating the presence of a Reference Bypass Switch in the remote PNA.

**VB Syntax**

```vbnet
value = cap.IsReferenceBypassSwitchPresent (n)
```

**Variable**

**Variable (Type) - Description**

- **value** (Boolean) - Variable to store the returned value.
  - True - Reference Bypass Switch is present.
  - False - Reference Bypass Switch is not present.
- **cap** A [Capabilities (object)]
- **n** (Long) - port number to query for reference bypass switch

**Return Type**
Boolean

**Default**
Not Applicable

**Examples**

```vbnet
value = cap.IsReferenceBypassSwitchPresent(1) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_IsReferenceBypassSwitchPresent(long portNumber, VARIANT_BOOL * present);
```

**Interface**
ICapabilities
### IsSParameter Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns true if measurement represents an S-Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = meas.IsSParameter</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement <em>(object)</em></td>
</tr>
<tr>
<td>value</td>
<td><strong>(Boolean)</strong></td>
</tr>
<tr>
<td></td>
<td>True - measurement is an S-Parameter</td>
</tr>
<tr>
<td></td>
<td>False - measurement is NOT an S-Parameter</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>True</td>
</tr>
<tr>
<td>Examples</td>
<td><code>print app.IsSParameter</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT IsSparameter([out, retval] VARIANT_BOOL * bVal);</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
### IterationsTolerance Property

**Description**  
This command, along with [MaximumIterationsPerPoint Property](#), deal with adjustments made to the source power.

Sets the maximum desired deviation from the sum of the test port power and the offset value. Power readings will continue to be made, and source power adjusted, until a reading is within this tolerance value or the max number of readings has been met. The last value to be read is the valid reading for that data point.

The following two commands allow for settling of power readings:

- [ReadingsPerPoint Property](#)
- [ReadingsTolerance Property](#)

**VB Syntax**

```
pwrCal.IterationsTolerance = value
```

**Variable**  
**(Type) - Description**

- `pwrCal` *(object)* - A [SourcePowerCalibrator](#) (object)
- `value` *(Double)* – Tolerance value in dBm. Choose any number between 0 and 5

**Return Type**

Double

**Default**

.05 dB

**Examples**

```vbnet
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.IterationsTolerance = .1 'Write
ReadTol = powerCalibrator.IterationsTolerance 'Read
```

**C++ Syntax**

```
HRESULT get_IterationsTolerance( double *pVal);
HRESULT put_IterationsTolerance( double newVal);
```

**Interface**

[ISourcePowerCalibrator3](#)

---

Last Modified:

17-Apr-2007 Clarified verbage
**KaiserBeta Property**

**Description**  
Sets or returns the Kaiser Beta of Time Domain transform windows

**VB Syntax**  
`trans.KaiserBeta = value`

**Variable**  
- **(Type)** - **Description**
  - `trans`  
    - A Transform **(object)**
  - `value`  

**Return Type**  
Single

**Default**  
0

**Examples**  
```
trans.KaiserBeta = 6 'sets the Kaiser Beta of a transform window
KB = trans.KaiserBeta 'Read
```

**C++ Syntax**  
```
HRESULT get_KaiserBeta(float *pVal)
HRESULT put_KaiserBeta(float newVal)
```

**Interface**  
ITransform
### L0 Property

**Description**
Sets and Returns the L0 (L-zero) value (the first inductance value) for the calibration standard.

To set the other inductance values, use L1, L2, L3.

For a detailed discussion of this value, search for App Note 8510-5B at [www.Agilent.com](http://www.Agilent.com).

**VB Syntax**
```vbnet
calstd.L0 = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

| calstd    | A CalStandard *(object)*. Use calKit.GetCalStandard to get a handle to the standard. |

| value     | *(single)* - Value for L0 in femtohenries(1E-15) |

**Return Type**
Single

**Default**
Not Applicable

**Examples**
```vbnet
calstd.L0 = 15 'Write the value of L0 = 15 femtohenries
Induct0 = calstd.L0 'Read the value of L0
```

**C++ Syntax**
```cpp
HRESULT get_L0(float *pVal)
HRESULT put_L0(float newVal)
```

**Interface**
ICalStandard
Description
Sets and Returns the L1 value (the second inductance value) for the calibration standard.
To set the other inductance values, use L0, L2, L3.
For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

VB Syntax
`calstd.L1 = value`

Variable (Type) - Description
`value` (single) - Value for L1.

Return Type
Single

Default
Not Applicable

Examples
`calstd.L1 = 15 'Write the value of L1`
`Induct1 = calstd.L1 'Read the value of L1`

C++ Syntax
HRESULT get_L1(float *pVal)
HRESULT put_L1(float newVal)

Interface
ICalStandard
### L2 Property

**Description**
Sets and Returns the L2 value (the third inductance value) for the calibration standard.

To set the other inductance values, use `L0`, `L1`, `L3`.

For a detailed discussion of this value, search for App Note 8510-5B at [www.Agilent.com](http://www.Agilent.com).

**VB Syntax**

```
calstd.L2 = value
```

**Variable**

- **(Type)**: Description
  - `value` — *(single)* - Value for L2.

**Return Type**

Single

**Default**

Not Applicable

**Examples**

```
calstd.L2 = 15 'Write the value of L2.
Induct2 = calstd.L2 'Read the value of L2
```

**C++ Syntax**

```cpp
HRESULT get_L2(float *pVal)
HRESULT put_L2(float newVal)
```

**Interface**

ICalStandard
L3 Property

### Description
Sets and Returns the L3 value (the third inductance value) for the calibration standard. To set the other inductance values, use L0, L1, L2.

For a detailed discussion of this value, search for App Note 8510-5B at [www.Agilent.com](http://www.Agilent.com).

### VB Syntax
```
calstd.L3 = value
```

### Variable (Type) - Description
- **value**: (single) - Value for L3.

### Return Type
Single

### Default
Not Applicable

### Examples
```
calstd.L3 = 15 'Write the value of L3.
Induct3 = calstd.L3 'Read the value of L3
```

### C++ Syntax
```
HRESULT get_L3(float *pVal)
HRESULT put_L3(float newVal)
```

### Interface
ICalStandard
### Label Property

**Description**
Sets and Returns the label for the calibration standard. The label is used to prompt the user to connect the specified standard.

**VB Syntax**

```vbnet
  calstd.Label = value
```

**Variable**

- **calstd**
  - A CalStandard **(object)**. Use calKit.GetCalStandard to get a handle to the standard.
- **value**
  - **(string)** - between 1 and 12 characters long. Cannot begin with a numeric.

**Return Type**
String

**Default**
Not Applicable

**Examples**

```vbnet
calstd.Label = "Short" 'Write
stdLabel = calstd.Label 'Read
```

**C++ Syntax**

- HRESULT get_Label(BSTR *pVal)
- HRESULT put_Label(BSTR newVal)

**Interface**
ICalStandard
# Label Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or gets the display label for a given channel/testset combination. The label appears in a status bar at the bottom of the PNA display when the <a href="#">ShowProperties</a> property is set to TRUE.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>tset.Label(chNum) = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>Type</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><em>tset</em> A <a href="#">TestsetControl</a> object.</td>
</tr>
<tr>
<td></td>
<td>Obtained from the <a href="#">ExternalTestsets</a> collection.</td>
</tr>
<tr>
<td></td>
<td><em>chNum</em> <em>(Integer)</em> Channel number of the measurement.</td>
</tr>
<tr>
<td></td>
<td><em>value</em> <em>(String)</em> The text of the label.</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>None</td>
</tr>
</tbody>
</table>
| Examples    | 'The following sets the label for channel 5 corresponding to a given testset object.'  

```vbnet
testset1.label(5) = 'High-power output'
```

See [External Testset Program](#)

| C++ Syntax | HRESULT get_Label(long channelNum, BSTR *pLabel);  |
|            | HRESULT put_Label(long channelNum, BSTR label);  |
| Interface  | ITestsetControl                                   |
**LastModified Property**

**Description**
Returns the time stamp of the last modification to this Cal Set.

This property always returns a time stamp based on the Greenwich Mean Time (GMT) regardless of the local time zone setting of the PNA.

The Cal Set properties that are viewed on the PNA user interface are converted to the local time of the PNA.

**VB Syntax**

```vbnet
value = Object.LastModified
```

**Variable**  
**Type** - Description

- `object` Channel *(object)*  
- `or`  
  - CalSet *(object)* - Read-only property

- `value` *(Variant)* – Variable to store the time stamp.

**Return Type**
Variant

**Default**
Not Applicable

**Examples**

```vbnet
date = CalSet.LastModified 'Read
```

**C++ Syntax**

```c++
HRESULT get_LastModified(VARIANT* datetime)
```

**Interface**
ICalSet3

---

Last Modified:

25-Apr-2007  Added GMT note
LimitTestFailed Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the results of limit testing for the measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>testFailed = meas.LimitTestFailed</td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>testFailed</td>
<td><strong>(boolean)</strong> Variable to store the returned value</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Limit Test Passed</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Limit Test Failed</td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement <strong>(object)</strong></td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False returned if there is no testing in progress.</td>
</tr>
<tr>
<td>Examples</td>
<td>Dim testRes As Boolean</td>
</tr>
<tr>
<td></td>
<td>testRes = meas.LimitTestFailed</td>
</tr>
<tr>
<td></td>
<td>MsgBox (testRes)</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_LimitTestFailed(VARIANT_BOOL* trueIfFailed)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
**EndStimulus Property**

**Description**  When constructing a limit line, specifies the stimulus value for the end of the segment.

**VB Syntax**  

```
limtseg.EndStimulus = value
```

**Variable**  

- **limtseg**: A LimitSegment *(object)*
- **value**: *(double)* - End Stimulus X-axis value. No units

**Return Type**  Double

**Default**  0

**Examples**  

```
Set limtseg = meas.LimitTest(1)
limtseg.EndStimulus = 8e9 'Write
EndStim = limtseg.EndStimulus 'Read
```

**C++ Syntax**  

- HRESULT get_EndStimulus(double *pVal)
- HRESULT put_EndStimulus(double newVal)

**Interface**  ILimitSegment
# EndResponse Property

**Description**
When constructing a limit line, specifies the amplitude value at the end of the limit segment.

**VB Syntax**
```vbnet
limtseg.EndResponse = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>limts</code></td>
<td>A LimitSegment (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) - Y-axis value of the End Response limit. No units</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
Set limtseg = meas.LimitTest(1)
limtseg.EndResponse = 10 'Write

EndResp = limtseg.EndResponse 'Read
```

**C++ Syntax**
```cpp
HRESULT get_EndResponse(double *pVal)
HRESULT put_EndResponse(double newVal)
```

**Interface**
ILimitSegment
**Type (limit) Property**

**Description**
Specifies the Limit Line type.

**VB Syntax**
`limt(index).Type = value`

**Variable**
- **(Type)** - Description
  - *limt* - A LimitSegment *(object)*
  - *index* - (variant) - Limit line number in the LimitTest collection
  - *value* - *(enum NALimitSegmentType)* - Limit Line type. Choose from:
    - 0 - `nalimitSegmentType_OFF` - turns limit line OFF
    - 1 - `nalimitSegmentType_Maximum` - limit line fails with a data point ABOVE the line
    - 2 - `nalimitSegmentType_Minimum` - limit line fails with a data point BELOW the line

**Return Type**
Long Integer

**Default**
0 - OFF

**Examples**
Set `limts = meas.LimitTest`

`limts.Type = nalimitSegmentType_Maximum` *Write*

`limitType = limts.Type` *Read*

**C++ Syntax**
- `HRESULT put_Type(tagNALimitSegmentType *pVal)`
- `HRESULT get_Type(tagNALimitSegmentType newVal)`

**Interface**
ILimitSegment
## LineDisplay Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns the display of limit lines ON or OFF. To turn limit TESTING On and OFF, use State Property.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note</strong></td>
<td>Trace data must be ON to view limit lines</td>
</tr>
</tbody>
</table>

### VB Syntax

```
limitst.LineDisplay = state
```

### Variable

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>limitst</td>
<td>A LimitTest (object)</td>
</tr>
<tr>
<td>state</td>
<td>(boolean)</td>
</tr>
<tr>
<td>False</td>
<td>Turns the display of limit lines OFF</td>
</tr>
<tr>
<td>True</td>
<td>Turns the display of limit lines ON</td>
</tr>
</tbody>
</table>

### Return Type

Long Integer

### Default

True

### Examples

```
Limttest.LineDisplay = true 'Write
lineDsp = Limttest.LineDisplay 'Read
```

### C++ Syntax

```
HRESULT get_LineDisplay(VARIANT_BOOL *pVal)
HRESULT put_LineDisplay(VARIANT_BOOL newVal)
```

### Interface

ILimitTest
Read/Write

LoadCharFromFile Property

Description
Sets and returns whether a Mixer characterization file is to be loaded. Specify and load the filename with CharFileName Property.

VB Syntax
`VMC.LoadCharFromFile = bool`

Variable (Type) - Description
- `VMC` (object)
- `bool` (Boolean)
- `True` - Load from file
- `False` - Perform mixer characterization

Return Type
Boolean

Default
False

Examples
`value = VMC.LoadCharFromFile`

C++ Syntax
```c++
HRESULT put_LoadCharFromFile(VARIANT_BOOL bLoadCharFromFile);
HRESULT get_LoadCharFromFile(VARIANT_BOOL *bLoadCharFromFile);
```

Interface
VMCType
### LoadPort Property

**Description**
Returns the load port number associated with an S-parameter reflection measurement. If the measurement is not a reflection S-parameter, the number returned by this property will have no meaning.

**VB Syntax**
\[\text{loadPort} = \text{meas}.\text{LoadPort}\]

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>loadPort</td>
<td>(long integer)</td>
<td>The reflection measurement's load port number.</td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```vba
Set meas = pna.ActiveMeasurement
loadPort = meas.LoadPort
```

**C++ Syntax**

```c++
HRESULT get_LoadPort(long *pPortNumber);
```

**Interface**
IMeasurement
Write/Read

LocalLockoutState Property

Description  Prevents use of the mouse, keyboard, and front panel while your program is running. Use of these controls while this property is set TRUE causes an error message on the PNA display. To prevent these messages, see About Error Messages.

VB Syntax  
```vbnet
app.LocalLockoutState = bool
```

Variable  
*Type* - Description

- **app**  An [Application](#) *(object)*
- **bool** *(boolean)* - Choose either:
  - **False** - User Interface is NOT locked out.
  - **True** - User Interface IS locked out.

Return Type  Boolean

Default  False

Examples  
```vbnet
app.LocalLockoutState = True 'Write
block = app.LocalLockoutState 'Read
```

C++ Syntax  
```c
HRESULT get_LocalLockoutState(VARIANT_BOOL *State);
HRESULT put_LocalLockoutState(VARIANT_BOOL *State);
```

Interface  IApplication4
### Locator Property

**Description**
Specifies the location of the power meter / sensor that is used during a source power calibration.

Use [Interface Property](#) to specify the type of interface.

**VB Syntax**
```
pwrMtrInterface.Locator = value
```

**Variable**

```
Class PowerMeterInterface

Public Property Locator As String
End Class
```

- **pwrMtrInterface** *(object)* - A [PowerMeterInterface](#) (object)
- **value** *(string)* - Location of the power meter / sensor, depending on the type of [Interface](#).
  - For **naGPIB**, address of the power meter. Choose any integer between 0 and 30.
  - For **naUSB**, the ID string of the power sensor. Use [USBPowerMeterCatalog Property](#) to see a list of ID strings of connected power sensors.
  - For **naLAN**, the hostname or IP address of the power meter.

**Return Type**
String

**Default**
Not applicable

**Examples**
```
pwrMeterInterface.Locator = "13"  'GPIB address
pwrMeterInterface.Locator = "Agilent Technologies,U2000A,MY12345678,X.01.16"  'USB ID string
pwrMeterInterface.Locator = "mymeter.agilent.com"  'LAN
```

**C++ Syntax**
```
HRESULT put_Locator( BSTR pValue );
HRESULT get_Locator(BSTR* pValue );
```

**Last Modified:**
24-Jul-2007   MX New topic
## LODeltaFound Property

**Description**
Returns the LO frequency delta from the specified tuning sweep.

**VB Syntax**
```vbscript
value = embedLODiag.LODeltaFound (n)
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Double) Variable to store the returned data.</td>
</tr>
</tbody>
</table>

**embedLODiag**
An `EmbeddedLODiagnostic` object

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>(Long) Tuning sweep number. Use <code>NumberOfSweeps</code> to find the number of sweeps taken.</td>
</tr>
</tbody>
</table>

**Default**
Not Applicable

**Examples**
```vbscript
data= embedLO.LODeltaFound 3 'read
```

**C++ Syntax**
```c++
HRESULT LODeltaFound(long sweep, double* deltaLO);
```

**Interface**
IEMbeddedLODiagnostic

---

Last Modified:

13-Apr-2007    MX New topic
LODenominator Property

Description
Sets or returns the denominator value of the LO Fractional Multiplier.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

VB Syntax
mixer.LODenominator (n) = value

Variable
(Type) - Description
mixer (A Mixer) (object)
value (LO denominator value) (Long)
n (LO stage number) (Long)

Return Type
Long

Default
1

Examples
Print mixer.LODenominator(1) ’prints the value of the first LODenominator

C++ Syntax
HRESULT get_LODenominator(long *pVal)
HRESULT put_LODenominator(long newVal)

Interface
IMixer

Last Modified:
4-Mar-2008  Added note.
LOFixedFrequency Property

Description  Sets or returns the LO frequency value.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

VB Syntax  mixer.LOFixedFrequency (n) = value

Variable  (Type) - Description
mixer  A Mixer (object)
value  (double) - LO Frequency in Hertz.
n  (Long) - LO stage number
Choose from 1 or 2

Return Type  Double

Default  0 Hz

Examples  Print mixer.LOFixedFrequency(1) 'prints the value of the first LO fixed frequency

C++ Syntax  HRESULT get_LOFixedFrequency(double *pVal)
HRESULT put_LOFixedFrequency(double newVal)

Interface  IMixer

Last Modified:
4-Mar-2008  Added note.
**LOFrequencyDelta Property**

**Description**
Sets and returns LO Frequency Delta. There is usually no need to set this value. Read this value to determine the difference between the LO Frequency that is stated in the Mixer dialog box and the last measured LO Frequency.

**VB Syntax**
```
embedLO.LOFrequencyDelta = value
```

**Variable (Type) - Description**
- `embedLO` An [EmbeddedLO](#) (object)
- `value` (Double) LO Frequency delta in Hertz.

**Return Type** (Double)

**Default**
Not Applicable

**Examples**
```
embedLO.LOFrequencyDelta = 0 'write

value = embedLO.LOFrequencyDelta 'read
```

**C++ Syntax**
```
HRESULT get_LOFrequencyDelta(double* val);
HRESULT put_LOFrequencyDelta(double val);
```

**Interface** IEmbeddedLO

---

Last Modified:

13-Apr-2007 MX New topic
LogMagnitudeOffset Property  Superseded

Description  Note: This property is replaced by DoReceiverPowerCal Method.

For Receiver Calibration - Sets or returns the value to offset the normalized unratioed power measurement data. The unratioed power measurement is effectively calibrated to the power level specified by the value of LogMagnitudeOffset as soon as the Normalization property is set to ON after calling the DataToDivisor method.

To offset the data trace magnitude a specified value, use MagnitudeOffset Property

VB Syntax  meas.LogMagnitudeOffset = value

Variable  (Type) - Description

meas  (object) - A Measurement object

value  (double) - Offset value in dBm.

Return Type  Double

Default  0

Examples  meas.LogMagnitudeOffset = -10 'Write (-10 dBm)

calpower = meas.LogMagnitudeOffset 'Read
meas.DataToDivisor 'Store meas data as measurement divisor
meas.Normalize = 1 'Measurement is now calibrated to -10 dBm

C++ Syntax  HRESULT put_LogMagnitudeOffset(double newVal);

HRESULT get_LogMagnitudeOffset(double *pVal);

Interface  IMeasurement
LOName Property

**Description**
Sets or returns the name of the PNA internal source or external source to use as the LO in an FCA measurement.

See [Remotely Specifying a Source Port](#).

**VB Syntax**
mixer.LOName (n) = value

**Variable**
- **mixer** (Type) - Description
  A Mixer (object)
- **n** (Long) - LO stage number
  Choose from 1 or 2
- **value** (string) - LO Source name. Use [SourcePort Property](#) to return a list of valid source ports. An external source must be configured and selected to be valid. Learn more about external source configuration.

**Return Type**
String

**Default**
"Not controlled"

**Examples**
mixer.LOName(1) = "MySource"

**C++ Syntax**
HRESULT get_LOName(string *pVal)
HRESULT put_LOName(string newVal)

**Interface**
MIXer

---

Last Modified:
23-Jul-2007   Updated for external source config.
**Description**
Sets or returns the numerator value of the LO Fractional Multiplier.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**
```
mixer.LONumerator (n) = value
```

**Variable**
- **mixer** (Object)
- **value** (Long) - LO denominator value
- **n** (Long) - LO stage number
  Choose from 1 or 2

**Return Type**
Long

**Default**
1

**Examples**
```
Print mixer.LONumerator(1) 'prints the value of the first LO Numerator
```

**C++ Syntax**
```
HRESULT get_LONumerator(long *pVal)
HRESULT put_LONumerator(long newVal)
```

**Interface**
IMixer

---

Last Modified:
4-Mar-2008   Added note.
LOPower Property

**Description**
Sets or returns the value of LO Power.

**VB Syntax**
`mixer.LOPower (n) = value`

**Variable (Type) - Description**

- `mixer` (object)
  - `n` (Long) - LO stage number
    - Choose from 1 or 2
  - `value` (double) - LO Power in dBm.

**Return Type**
Double

**Default**
-10dBm

**Examples**
`Print mixer.LOPower(1) 'prints the value of the LO Power`

**C++ Syntax**
`HRESULT get_LOPower(double *pVal)`
`HRESULT put_LOPower(double newVal)`

**Interface**
IMixer
LORangeMode Property

Description
Sets or returns the LO sweep mode.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

VB Syntax
mixer.LORangeMode (n) = value

Variable (Type) - Description
mixer A Mixer (object)
n (Long) - LO stage number.
Choose from 1 or 2

value (Enum as MixerRangeMode) - LO sweep mode. Choose from:
0 - mixSwept
1 - mixFixed

Return Type
Enum

Default
0 - mixSwept

Examples
mixer.LORangeMode(1)=mixSwept

C++ Syntax
HRESULT get_LORangeMode(long *pVal)
HRESULT put_LORangeMode(long newVal)

Interface
IMixer3

Last Modified:
4-Mar-2008  Added note.
# Loss Property

**Description**
Sets and Returns the insertion loss for the calibration standard.

**VB Syntax**
`calstd.loss = value`

**Variable**
- `value` (Type): Insertion loss in Gohms / sec. (Giga Ohms per second of electrical delay)

**Return Type**
Single

**Default**
Not Applicable

**Examples**
- `calstd.loss = 3.5 'Write 3.5 Gohms of loss`
- `stdLoss = calstd.loss 'Read the value of Loss`

**C++ Syntax**
- `HRESULT get_Loss(float *pVal)`
- `HRESULT put_Loss(float newVal)`

**Interface**
`ICalStandard`
## Loss (Source Power Cal) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the loss value associated with a PowerLossSegment.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>lossSeg.Loss = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>lossSeg</code></td>
<td>(object) - PowerLossSegment</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) – Loss value in dB. This can be any value between 0 and 200.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
</tbody>
</table>
| **Examples** | `lossSeg.Loss = 0.5 'Write`  
`lossVal = lossSeg.Loss 'Read` |
| **C++ Syntax** | `HRESULT put_Loss(Double newVal);`  
`HRESULT get_Loss(Double *pVal);` |
| **Interface** | IPowerLossSegment |
## LOStage Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the number of LO stages present in the mixer.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>mixer.<strong>LOStage</strong> = value</td>
</tr>
<tr>
<td>Variable</td>
<td><strong>mixer</strong> (Type) - A Mixer (object)</td>
</tr>
<tr>
<td></td>
<td><strong>value</strong> (Type) - (Long) Number of LO stages. Choose from 1 or 2</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long</td>
</tr>
<tr>
<td>Default</td>
<td>1</td>
</tr>
<tr>
<td>Examples</td>
<td>mixer.<strong>LOStage</strong> = 1 'sets the LO Stage value to 1</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_LOStage(long *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_LOStage(long newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMixer</td>
</tr>
</tbody>
</table>
### LOStartFrequency Property

**Description**
Sets or returns the LO start frequency value. This command can only be used with SMC (not VMC) measurements.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**

```
mixer.LOStartFrequency (n) = value
```

**Variable (Type) - Description**

- **mixer** A Mixer (object)
- **value** (double) - LO Start Frequency in Hertz.
- **n** (Long) - LO stage number

Choose from 1 or 2

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```
Print mixer.LOStartFrequency(1) 'prints the value of the first LO start frequency
```

**C++ Syntax**

```
HRESULT get_LOStartFrequency(long id, double *pVal)
HRESULT put_LOStartFrequency(long id,double newVal)
```

**Interface**
IMixer3

---

Last Modified:

4-Mar-2008 Added note.
LOStopFrequency Property

Description
Sets or returns the LO stop frequency value. This command can only be used with SMC (not VMC) measurements.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

VB Syntax
mixer.LOStopFrequency (n) = value

Variable (Type) - Description
mixer A Mixer (object)
value (double) - LO Stop Frequency in Hertz.
n (Long) - LO stage number
Choose from 1 or 2

Return Type Double

Default Not Applicable

Examples
PRINT mixer.LOStopFrequency(1) 'prints the value of the first LO stop frequency

C++ Syntax
HRESULT get_LOStopFrequency(long id, double *pVal)
HRESULT put_LOStopFrequency(long id, double newVal)

Interface IMixer3

Last Modified:
4-Mar-2008 Added note.
### About Magnitude Offset

**MagnitudeOffset Property**

**Description**
Offsets the data trace magnitude by the specified value.

To offset the data trace magnitude to a slope value that changes with frequency, use [MagnitudeSlopeOffset Property](#).

To implement a Receiver Cal offset, use [LogMagnitudeOffset property](#).

**VB Syntax**
```
meas.MagnitudeOffset = value
```

**Variable**

- **meas** *(object)* - A Measurement object
- **value** *(double)* - Offset value in dB.

**Return Type**
Double

**Default**
0

**Examples**
```
meas.MagnitudeOffset = 4 'Write
offs = meas.MagnitudeOffset 'Read
```

**C++ Syntax**
```
HRESULT put_MagnitudeOffset(double newVal);
HRESULT get_MagnitudeOffset(double *pVal);
```

**Interface**
IMeasurement4
**MagnitudeSlopeOffset Property**

**Description**
Offsets the data trace magnitude to a value that changes linearly with frequency. The offset slope begins at 0 Hz.

To offset the entire data trace magnitude by a specified value, use `MagnitudeOffset Property`.

**VB Syntax**

```
meas.MagnitudeSlopeOffset = value
```

**Variable**

**Type** - Description

- `meas` *(object)* - A `Measurement` object
- `value` *(double)* - Offset slope value in dB / 1GHz.

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
meas.MagnitudeSlopeOffset = 4 'Writes a slope offset of 4dB/1GHz.
offs = meas.MagnitudeSlopeOffset 'Read
```

**C++ Syntax**

```c++
HRESULT put_MagnitudeSlopeOffset(double newVal);
HRESULT get_MagnitudeSlopeOffset(double *pVal);
```

**Interface**

`IMeasurement4`
MarkerAnnotation Property

Description
Returns the Y-axis marker value from the specified tuning sweep. Use `IsMarkerOn` to confirm if a marker was used for the tuning sweep.

VB Syntax
```vbnet
value = embedLODiag.MarkerAnnotation(n)
```

Variable (Type) - Description

value  (String) Variable to store the returned data.

embedLODiag  An `EmbeddedLODiagnostic` object

n  (Long) Tuning sweep number. Use `NumberOfSweeps` to find the number of sweeps taken.

Default
Not Applicable

Examples
```plaintext
data = embedLO.MarkerAnnotation 3 'read
```

C++ Syntax
```cpp
HRESULT MarkerAnnotation(long sweep, BSTR* annotation);
```

Interface
`IEmbeddLODiagnostic`

Last Modified:
13-Apr-2007  MX New topic
## MarkerFormat Property

**Description**
Sets (or returns) the format of all the markers in the measurement. To override this setting for an individual marker, use `mark.Format`.

**VB Syntax**
```vbnet
meas.MarkerFormat = value
```

**Variable (Type) - Description**

- **meas**
  - A Measurement (object)

- **value**
  - (enum NAMarkerFormat) - Choose from:
    - 0 - `naMarkerFormat_LinMag`
    - 1 - `naMarkerFormat_LogMag`
    - 2 - `naMarkerFormat_Phase`
    - 3 - `naMarkerFormat_Delay`
    - 4 - `naMarkerFormat_Real`
    - 5 - `naMarkerFormat_Imaginary`
    - 6 - `naMarkerFormat_SWR`
    - 7 - `naMarkerFormat_LinMagPhase`
    - 8 - `naMarkerFormat_LogMagPhase`
    - 9 - `naMarkerFormat_RealImaginary`
    - 10 - `naMarkerFormat_ComplexImpedance`
    - 11 - `naMarkerFormat_ComplexAdmittance`
    - 12 - `naMarkerFormat_Kelvin`
    - 13 - `naMarkerFormat_Fahrenheit`
    - 14 - `naMarkerFormat_Centigrade`

**Return Type**
Long Integer

**Default**
1 - `naMarkerFormat_LogMag`

**Examples**
```vbnet
meas.MarkerFormat = naMarkerFormat_SWR 'Write

fmt = mark.Format 'Read
```

**C++ Syntax**
```cpp
HRESULT put_MarkerFormat(tagNAMarkerFormat NewFormat)
```

**Interface**
IMeasurement

**Last Modified:**
1144
1-Oct-2007  Added temperature formats
**InterpolateMarkers Method**

**Description**  
Turns All Marker Interpolation ON and OFF for the measurement. Marker interpolation enables X-axis resolution between the discrete data values. The analyzer will calculate the x and y-axis data values between discrete data points. To override this property for individual markers, use the Interpolated property.

**VB Syntax**  
`meas.Interpolate = value`

**Variable**  
*(Type)* - Description

- **meas**  
  A Measurement *(object)*

- **value** *(boolean)*
  - **False** - Turns interpolation OFF for all markers in the measurement
  - **True** - Turns interpolation ON for all markers in the measurement

**Return Type**  
Boolean

**Default**  
True (ON)

**Examples**  
`meas.Interpolate = 1`

**C++ Syntax**  
`HRESULT InterpolateMarkers(VARIANT_BOOL bNewVal)`

**Interface**  
IMeasurement
Number Property

Description  Returns the number of the marker.

VB Syntax  

marknum = mark.Number

Variable  (Type) - Description

marknum  (long) - Variable to store marker number

mark  A Marker (object)

Return Type  Long Integer

Default  Not applicable

Examples  

marknum = mark.Number 'Read

C++ Syntax  HRESULT get_Number(long *pVal)

Interface  IMarker
MarkerPosition Property

**Description**
Returns the X-axis marker position from the specified tuning sweep.

**VB Syntax**
```
value = embedLODiag.MarkerPosition (n)
```

**Variable**
- **value** *(Double)* Variable to store the returned data.
- **embedLODiag** An *EmbeddedLODiagnostic* *(object)*
- **n** *(Long)* Tuning sweep number. Use *NumberOfSweeps* to find the number of sweeps taken.

**Default**
Not Applicable

**Examples**
```
data= embedLO.MarkerPosition 3 'read
```

**C++ Syntax**
```
HRESULT MarkerPosition(long sweep, double *position);
```

**Interface**
IEembedLODiagnostic

---

Last Modified:
13-Apr-2007   MX New topic
MarkerReadout Property

**Description**
Enables or disables the readout of markers in the window. To show the marker on the screen use ShowMarkerReadout Method.

**VB Syntax**
```
win.MarkerReadout = state
```

**Variable (Type) - Description**

- `win` - A NAWindow (object)
- `state` - (boolean)
  - `True` - enables marker readout
  - `False` - disables marker readout

**Return Type**
Boolean

**Default**
True

**Examples**
```
win.MarkerReadout = True 'Write
State = app.ActiveNAWindow.MarkerReadout 'Read
```

**C++ Syntax**
```
HRESULT get_MarkerReadout(VARIANT_BOOL *pVal)
HRESULT put_MarkerReadout(VARIANT_BOOL newVal)
```

**Interface**
INAWindow
MarkerReadoutSize Property

**Description**
Specifies the size of font used when displaying Marker Readout in the selected window.

**VB Syntax**
```vbnet
win.MarkerReadoutSize = value
```

**Variable**

- **Type**: Description
  - **win**: A NAWindow (object)

- **value**
  - (enum NAFontSize)
    - 0 - naDefault - marker readout appears in default font size
    - 1 - naLarge - marker readout appears in large font size

**Return Type**
Long Integer

**Default**
naDefault

**Examples**
```vbnet
win.MarkerReadoutSize = naDefault 'write default size for marker readout

Dim Size As NAFontSize
Size = app.ActiveNAWindow.MarkerReadoutSize 'Read
```

**C++ Syntax**
```cpp
HRESULT get_MarkerReadoutSize(tagNAFontSize *pVal)
HRESULT put_MarkerReadoutSize(tagNAFontSize newVal)
```

**Interface**
INAWindow
## MarkerState Property

**Description**  
Sets or returns the ON / OFF state of the specified marker.

**VB Syntax**  
```
meas.MarkerState (n) = state
```

**Variable**  
(Type) - Description

- **meas**  
  A Measurement (object)

- **n**  
  (Long Integer) Marker number to turn on or off.

- **state**  
  (boolean) -  
  True - turns the specified marker ON  
  False - turns the specified marker OFF

**Return Type**  
Boolean

**Default**  
False

**Examples**  
```
meas.MarkerState(1) = True

reference = meas.MarkerState(2)
```

**C++ Syntax**  
```
HRESULT get_MarkerState(long markerNum, VARIANT_BOOL bState)
HRESULT put_MarkerState(long markerNum, VARIANT_BOOL* bState)
```

**Interface**  
IMeasurement3
## Type (Marker) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the marker type.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.Type = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum <code>NAMarkerType</code>) - Marker Type. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <code>naMarkerType_Normal</code> - the X-axis value for a normal marker will always be determined by the measurement data of the marker.</td>
</tr>
<tr>
<td></td>
<td>1 - <code>naMarkerType_Fixed</code> - retains and keeps its x-axis value at the time the marker type is set.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><code>naMarkerType_Normal</code></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.Type = naMarkerType_Normal</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>MrkType = mark.Type</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Type(tagNAMarkerType *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Type(tagNAMarkerType newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
### Stimulus Property

**Description**  
Sets and reads the X-Axis value of the marker. If the marker is a delta marker, the value will be relative to the reference marker.

**VB Syntax**  
`mark.Stimulus = value`

**Variable**  
**Type** - Description

- **mark**  
  A Marker *(object)*

- **value**  
  *(double)* - X-Axis value. Choose any number within the full span of the channel or User Range (if set).

**Return Type**  
Double

**Default**  
First activated Marker turns ON in the middle of the X-axis range. Subsequent markers turn ON at the position of the most recently active marker.

**Examples**  
`mark.Stimulus = 3e9 'Write`

`XVal = mark.Stimulus 'Read`

**C++ Syntax**  
`HRESULT get_Stimulus(double *pVal)`

`HRESULT put_Stimulus(double newVal)`

**Interface**  
`IMarker`
**Value Property**

**Description**
Reads the Y-axis value of the marker. If the marker is a delta marker, the value will be relative to the reference marker.

You cannot set the Y-axis value of a marker. The marker remains at the position at the time you set `marker.Type`.

**Note:** To accurately read the marker Y-axis value with trace smoothing applied, the requested format must match the displayed format. Otherwise, the returned value is unsmoothed data. For example, to read the smoothed marker value when measuring group delay, both the display format and the marker format must be set to (Group) Delay.

**VB Syntax**

```vbnet
YValue = mark.Value (format)
```

**Variable**

- **YValue**
  A variable to store the Y-axis value

- **mark**
  A Marker (`object`)

- **format**
  (`enum NAMarkerFormat`) - The format in which to return the marker's Y-axis value. The number in parenthesis following the format is the number of values that are returned in a variant array. Choose from:

  - 0 - `naMarkerFormat_LinMag` (1)
  - 1 - `naMarkerFormat_LogMag` (1)
  - 2 - `naMarkerFormat_Phase` (1)
  - 3 - `naMarkerFormat_Delay` (1)
  - 4 - `naMarkerFormat_Real` (1)
  - 5 - `naMarkerFormat_Imaginary` (1)
  - 6 - `naMarkerFormat_SWR` (1)
  - 7 - `naMarkerFormat_LinMagPhase` (2)
  - 8 - `naMarkerFormat_LogMagPhase` (2)
  - 9 - `naMarkerFormat_RealImaginary` (2)
  - 10 - `naMarkerFormat_ComplexImpedance` (3)
  - 11 - `naMarkerFormat_ComplexAdmittance` (3)
  - 12 - `naMarkerFormat_Kelvin` (1)
  - 13 - `naMarkerFormat_Fahrenheit` (1)
  - 14 - `naMarkerFormat_Celsius` (1)

**Return Type**
Variation - The previous list of formats indicates the number of values that are returned in a variant array.
<table>
<thead>
<tr>
<th>Default</th>
<th>Not applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Examples</strong></td>
<td><code>YVal = mark.Value</code> <em>Read</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_Value(tagNAMarkerFormat format, VARIANT *pVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>

Last modified:

- 1-Oct-2007       Added temperature formats
- Dec. 4, 2006     Added smoothing note
## MaximumFrequency Property

**Description**
Sets and Returns the maximum frequency for the calibration standard.

**VB Syntax**
`calstd.MaximumFrequency = value`

**Variable**
- **(Type) - Description**
  - `calstd` A CalStandard (**object**). Use calKit.[GetCalStandard](#) to get a handle to the standard.
  - `value` (**double**) - Maximum frequency in Hertz.

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```
calstd.MaximumFrequency = 9e9 'Write
maxFrequency = calstd.MaximumFrequency 'Read
```

**C++ Syntax**
```
HRESULT get_MaximumFrequency(double *pVal)
HRESULT put_MaximumFrequency(double newVal)
```

**Interface**
ICalStandard
| **Description** | Returns the maximum frequency of the remote PNA. |
| **VB Syntax** | `value = cap.MaximumFrequency` |
| **Variable** | **(Type) - Description** |
| `value` | (Double) - Variable to store the returned maximum frequency of the PNA. |
| `cap` | A [Capabilities](#) (object) |
| **Return Type** | Double |
| **Default** | Not Applicable |
| **Examples** | `value = cap.MaximumFrequency` 'Read' |
| **C++ Syntax** | `HRESULT get_MaximumFrequency(&frequencyMax);` |
| **Interface** | ICapabilities |
## MaximumFrequency (Source Power Cal) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Maximum usable frequency specified for the power sensor.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>pwrSensor.<strong>MaximumFrequency</strong> = value</td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td></td>
</tr>
<tr>
<td>pwrSensor (object)</td>
<td>A PowerSensor (object)</td>
</tr>
<tr>
<td>value (double)</td>
<td>Frequency in Hertz.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
</tbody>
</table>

### Examples

```vbnet
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.PowerSensors(1).MaximumFrequency = 6e9
'Write
MaxFreq = powerCalibrator.PowerSensors(1).MaximumFrequency
'Read
```

### C++ Syntax

```c++
HRESULT put_MaximumFrequency(double newVal);
HRESULT get_MaximumFrequency(double *pVal);
```

### Interface

IPowerSensor
**MaximumIFFilterSampleCount Property**

**Description**
Returns the maximum allowed value for the `IFFilterSampleCount` property for the queried PNA.

*Note:* This setting applies only to the E836X Opt. H11.

**VB Syntax**
```vbnet
value = IfConfig.MaximumIFFilterSampleCount
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>(long) The maximum allowed value that can be applied to the <code>IFFilterSampleCount</code> property.</td>
</tr>
</tbody>
</table>

**IfConfig**

`IFConfiguration` (object)

**Return Type**

Long Integer

**Default**

PNA Model number dependent

**Examples**

```vbnet
variable = App.ActiveChannel.IFConfiguration.MaximumIFFilterSampleCount
```

*See an example program*

**C++ Syntax**

```cpp
HRESULT get_MaximumIFFilterSampleCount( long * pMaxSampleCount );
```

**Interface**

`IIFConfiguration2`
### MaximumIFFrequency Property

**Description**
Returns the Maximum allowed value for the [IFFrequency Property](#).

**VB Syntax**
```vb
value = IfConfig.MaximumIFFrequency
```

**Variable**
- **(Type)**: `double`
- **Description**
  Variable to store the returned maximum allowed frequency that can be applied to the [IFFrequency Property](#).

**IfConfig**
- **Type**: `IFConfiguration` (object)

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```vb
val = App.ActiveChannel.IFConfiguration.MaximumIFFrequency 'Read
```

**C++ Syntax**
```cpp
HRESULT get_MaximumIFFrequency( double * pMaxFreq);
```

**Interface**
- `IIFConfiguration3`

---

Last Modified:

18-Jun-2007  MX New topic
**MaximumIterationsPerPoint Property**

**Description**
This command, along with `IterationsTolerance Property` deal with adjustments made to the source power.

Sets the maximum number of readings to take at each data point for iterating the source power. Power readings will continue to be made, and source power adjusted, until a reading is within the `IterationsTolerance` value or this max number of readings has been met. The last value to be read is the valid reading for that data point.

The following two commands allow for settling of power readings:
- **ReadingsPerPoint Property**
- **ReadingsTolerance Property**

**VB Syntax**

```
pwrCal.MaximumIterationsPerPoint = value
```

**Variable (Type) - Description**

- `pwrCal` (object) - A `SourcePowerCalibrator` (object)
- `value` (Long) – Maximum number of readings. Choose any number between 1 and 100.

**Return Type**
Long Integer

**Default**
5

**Examples**

```
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.MaximumIterationsPerPoint = 5 'Write
MaxReads = powerCalibrator.MaximumIterationsPerPoint 'Read
```

**C++ Syntax**

```
HRESULT get_MaximumIterationsPerPoint( long *pVal);
HRESULT put_MaximumIterationsPerPoint( long newVal);
```

**Interface**
ISourcePowerCalibrator3

---

**Last Modified:**

17-Apr-2007  Clarified verbage
# MaximumNumberOfChannels Property

**Description**
Returns the maximum possible number of channels that can be used in the PNA.

**VB Syntax**
```vbnet
value = cap.MaximumNumberOfChannels
```

**Variable**
- **value** *(Long)* - Variable to store the returned maximum value for number of channels.
- **cap** A `Capabilities` *(object)*

**Return Type**
Long

**Default**
Not Applicable

**Examples**
```vbnet
value = cap.MaximumNumberOfChannels 'Read
```

**C++ Syntax**
```cpp
HRESULT get_MaximumNumberOfChannels(long *maximumNumberOfChans);
```

**Interface**
ICapabilities2
MaximumNumberOfTracesPerWindow Property

**Description**
Returns the maximum possible number of traces that can reside in any window.

**VB Syntax**
```
value = cap.MaximumNumberOfTracesPerWindow
```

**Variable**
(Type) - Description

- `value` *(Long)* - Variable to store the returned maximum value for number of traces.
- `cap` - A [Capabilities](object)

**Return Type**
Long

**Default**
Not Applicable

**Examples**
```
value = cap.MaximumNumberOfTracesPerWindow 'Read
```

**C++ Syntax**
```
HRESULT get_MaximumNumberOfTracesPerWindow(long * maximumNumberOfTraces);
```

**Interface**
ICapabilities2
Description
Returns the maximum possible number of windows that can be present on the PNA screen.

VB Syntax
value = cap.MaximumNumberOfWindows

Variable (Type) - Description
value (Long) - Variable to store the returned maximum value for number of windows.
cap A Capabilities (object)

Return Type
Long

Default
Not Applicable

Examples
value = cap.MaximumNumberOfWindows 'Read

C++ Syntax
HRESULT get_MaximumNumberOfWindows(long * maximumNumberOfWindows);

Interface
ICapabilities2
MaximumNumberOfPoints Property

**Description**
Returns the maximum possible number of data points.

**VB Syntax**
```vbnet
value = obj.MaximumNumberOfPoints
```

**Variable**
- **Type**: Long
- **Description**: Variable to store the returned maximum value for number of points.

**Return Type**
Long

**Default**
Not Applicable

**Examples**
```vbnet
value = cap.MaximumNumberOfPoints 'Read
```

**C++ Syntax**
```csharp
HRESULT get_MaximumNumberOfPoints(long *maximumNumberOfPoints);
```

**Interface**
ICapabilities
IGainCompression

---

Last Modified:
3-Dec-2007    MX New topic
MaximumReceiverStepAttenuator Property

Description
Returns the maximum amount of receiver attenuation.

VB Syntax
value = cap.MaximumReceiverStepAttenuator(n)

Variable (Type) - Description
value (Double) - Variable to store the returned value of maximum receiver attenuation.

cap A Capabilities (object)

n (Long) - port number to query for step attenuators

Return Type
Double

Default
Not Applicable

Examples
value = cap.MaximumReceiverStepAttenuator.Read

C++ Syntax
HRESULT get_MaximumReceiverStepAttenuator(long portNumber, double * attenuation);

Interface
ICapabilities
### MaximumSourceALCPower Property

**Description**
Returns a value indicating the maximum amount of source ALC power.

**VB Syntax**

\[ value = cap.MaximumSourceALCPower(n) \]

- **Variable**
  - **Type**: (Double)
  - **Description**: Variable to store the returned value for the maximum amount of source ALC power.

- **cap**: A `Capabilities` (object)

- **n**: (Long) - source number to query for maximum ALC power

**Return Type**
Double

**Default**
Not Applicable

**Examples**

\[ value = cap.MaximumSourceALCPower \]

**C++ Syntax**

`HRESULT get_MaximumSourceALCPower(long sourceNum, double * power);`

**Interface**
ICapabilities
# MaximumSourceStepAttenuator Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Returns a value for the maximum amount of source attenuation.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.MaximumSourceStepAttenuator (n)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>(Description)</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(Double)</strong> - Variable to store the returned value for the maximum amount of source attenuation.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A <a href="#">Capabilities</a> (object)</td>
</tr>
<tr>
<td><code>n</code></td>
<td><strong>(Long)</strong> - port number to query for the maximum amount of source attenuation</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.MaximumSourceStepAttenuator 2</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_MaximumSourceStepAttenuator(long portNumber, double * attenuation );</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>

Last Modified:

19-Sep-2007  Added port arg to ex.
**MaxPreciseTuningIterations Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the maximum number of tuning iterations to achieve the precise tolerance.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>embedLO.MaxPreciseTuningIterations = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>embedLO</code></td>
<td>An <a href="object">EmbeddedLO</a> object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Maximum number of tuning iterations. Choose a number between</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>(Long)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>5</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>embedLO.MaxPreciseTuningIterations = 3 'write</code></td>
</tr>
<tr>
<td><code>value = embedLO.MaxPreciseTuningIterations 'read</code></td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_MaxPreciseTuningIterations(long* iter);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_MaxPreciseTuningIterations(long iter);</code></td>
</tr>
</tbody>
</table>

**Interface** | IEmbeddedLO |

---

Last Modified:

13-Apr-2007  MX New topic
## Mean Property

**Description**
Returns the mean value of the measurement. To retrieve all 3 statistics value at the same time, use `meas.GetTraceStatistics`.

**VB Syntax**

```vbnet
average = meas.Mean
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>average</td>
<td>(single) - Variable to store mean value</td>
</tr>
</tbody>
</table>

**meas**
A Measurement (object)

**Return Type**
Single

**Default**
Not applicable

**Examples**

```vbnet
Dim average as Single
average = meas.Mean 'Read
```

**C++ Syntax**

```cpp
HRESULT get_Mean(float* mean)
```

**Interface**
IMeasurement
Medium Property

**Description**
Sets and Returns the media type of the calibration standard.

**VB Syntax**

calstd.Medium = value

**Variable**

*calstd* A *CalStandard (object)*. Use *calKit.GetCalStandard* to get a handle to the standard.

*value* (enum NACalStandardMedium) - Medium of the transmission line of the standard. Choose from:
0 - naCoax - Coaxial Cable
1 - naWaveGuide

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**

calstd.Medium = naCoax 'Write

stdMedium = calstd.Medium 'Read

**C++ Syntax**

HRESULT get_Medium(tagNACalStandardMedium *pVal)
HRESULT put_Medium(tagNACalStandardMedium newVal)

**Interface**
ICalStandard
Write/Read

MinimumFrequency Property

**Description**
Sets and Returns the minimum frequency for the calibration standard.

**VB Syntax**
```vbnet
calstd.MinimumFrequency = value
```

**Variable**
- **(Type)**: A CalStandard *(object)*. Use calKit.GetCalStandard to get a handle to the standard.
- **value**: *(double)* - Minimum frequency in Hertz.

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```vbnet
calstd.MinimumFrequency = 300e3 'Write
minFrequency = calstd.MinimumFrequency 'Read
```

**C++ Syntax**
```cpp
HRESULT get_MinimumFrequency(double *pVal)
HRESULT put_MinimumFrequency(double newVal)
```

**Interface**
ICalStandard
MinimumFrequency Property

Description
Returns the minimum frequency of the remote PNA.

VB Syntax
value = cap.MinimumFrequency

Variable
(Type) - Description
value (Double) - Variable to store the returned minimum frequency of the PNA.

cap A Capabilities (object)

Return Type
Double

Default
Not Applicable

Examples
value = cap.MinimumFrequency 'Read

C++ Syntax
HRESULT get_MinimumFrequency(double *pVal)

Interface
ICapabilities
Minimum Frequency (Source Power Cal) Property

**Description**  
Minimum usable frequency specified for the power sensor.

**VB Syntax**  
`pwrSensor.MinimumFrequency = value`

**Variable**  
*(Type)* - Description

- `pwrSensor` *(object)* - A PowerSensor (object)
- `value` *(double)* - Frequency in Hertz.

**Return Type**  
Double

**Default**  
0

**Examples**  
Set `powerCalibrator = pna.SourcePowerCalibrator`
`powerCalibrator.PowerSensors(1).MinimumFrequency = 300e3`

'Write

MinFreq = powerCalibrator.PowerSensors(1).MinimumFrequency

'Read

**C++ Syntax**  
`HRESULT put_MinimumFrequency(double newVal);`

`HRESULT get_MinimumFrequency(double *pVal);`

**Interface**  
`IPowerSensor`
MinimumIFFilterSampleCount Property

Description
Returns the Minimum allowed value for the IFFilterSampleCount property for the queried PNA.

Note: This setting applies only to the E836X Opt. H11.

VB Syntax
value = IfConfig.MinimumIFFilterSampleCount

Variable (Type) - Description
value (long) The minimum allowed value that can be applied to the IFFilterSampleCount property.

IfConfig IFConfiguration (object)

Return Type Long Integer

Default PNA Model number dependent

Examples
variable = App.ActiveChannel.IFConfiguration.MinimumIFFilterSampleCount

See an example program

C++ Syntax
HRESULT get_MinimumIFFilterSampleCount( long * pMinSampleCount );

Interface IIFConfiguration2
**MinimumIFFrequency Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the minimum allowed value for the IFFrequency Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( value = IfConfig.MinimumIFFrequency )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td>( value )</td>
<td><strong>(double)</strong> Variable to store the returned minimum allowed frequency that can be applied to the IFFrequency Property.</td>
</tr>
<tr>
<td><strong>IfConfig</strong></td>
<td><strong>IFConfiguration</strong> (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>( \text{val} = \text{App.ActiveChannel.IFConfiguration.MinimumIFFrequency} )'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_MinimumIFFrequency( double * pMinFreq);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><strong>IIFConfiguration3</strong></td>
</tr>
</tbody>
</table>

Last Modified:

18-Jun-2007   MX New topic
MinimumNumberOfPoints Property

Description
Returns the minimum possible number of data points for a data trace.

VB Syntax
value = cap.MinimumNumberOfPoints

Variable (Type) - Description
value (Long) - Variable to store the returned minimum value for number of points.

cap A Capabilities (object)

Return Type
Long

Default
Not Applicable

Examples
value = cap.MinimumNumberOfPoints

C++ Syntax
HRESULT get_MinimumNumberOfPoints(double * minimumNumberOfPoints);

Interface
ICapabilities
MinimumReceiverStepAttenuator Property

Description
Returns a value indicating the minimum amount of receiver attenuation.

VB Syntax
value = cap.MinimumReceiverStepAttenuator (n)

Variable (Type) - Description
value (Double) - Variable to store the returned minimum value of receiver attenuation.

cap A Capabilities (object)

n (Long) - port number to query for minimum value of receiver attenuation

Return Type
Double

Default
Not Applicable

Examples
value = cap.MinimumReceiverStepAttenuator'Read

C++ Syntax
HRESULT get_MinimumReceiverStepAttenuator(long portNumber, double * attenuation );

Interface
ICapabilities
### MinimumSourceALCPower Property

**Description**
Returns a value indicating the minimum amount of source ALC power.

**VB Syntax**
```vb
value = cap.MinimumSourceALCPower(n)
```

**Variable (Type) - Description**
- **value** (Double) - Variable to store the returned minimum value of source ALC power.
- **cap** A Capabilities (object)
- **n** (Long) - source number to query for the minimum value of source ALC power

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```vb
value = cap.MinimumSourceALCPower 'Read
```

**C++ Syntax**
```cpp
HRESULT get_MinimumSourceALCPower(long sourceNum, double * power);
```

**Interface**
ICapabilities
## Mode Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the type of transform.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>trans.Mode = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>trans</code></td>
<td>A Transform (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NATransformMode) - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naTransformBandpassImpulse</td>
</tr>
<tr>
<td></td>
<td>1 - naTransformLowpassImpulse</td>
</tr>
<tr>
<td></td>
<td>2 - naTransformLowpassStep</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>NATransformMode</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0 - naTransformBandpassImpulse</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>trans.Mode = naTransformLowpassStep 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>transmode = trans.Mode 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Mode(tagNATransformMode *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Mode(tagNATransformMode newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ITransform</td>
</tr>
</tbody>
</table>
Multiplier Property

**Description**
Sets and returns the multiplier value to be used when coupling this range to the primary range. Learn more about multiplier value.

This setting is valid only if this range is coupled to the primary range.

**VB Syntax**

\[ \text{FOMRange.Multiplier} = \text{value} \]

**Variable (Type) - Description**

*object*  
An FOMRange (object)

*value*  
(Double) - Multiplier value.-(Unitless)

**Return Type**

Double

**Default**

1

**Examples**

\[ \text{fomRange.Multiplier} = 2 \ 'Write \]

\[ \text{Mult} = \text{fomRange.Multiplier} 'Read \]

**C++ Syntax**

HRESULT get_Multiplier(double *pVal)
HRESULT put_Multiplier(double pVal)

**Interface**

IFOMRange
### Write / Read

#### Name Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Sets or returns the Name of the Cal Set.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>CalSet.Name = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>CalSet</code></td>
<td>(object) - A <code>Cal Set</code> object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(string) - Name of the Cal Set.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

#### Examples

```vbscript
Dim pna
set pna=CreateObject("AgilentPNA835x.Application")

Dim calsets
set calsets=pna.getcalmanager.calsets

Dim c
for each c in calsets
wscript.echo c.name
' Changes the name of CalSet_1
if c.name="CalSet_1" then c.name="New"
next
```

#### C++ Syntax

```cpp
HRESULT get_Name(BSTR *name)
HRESULT put_Name(BSTR name);
```

#### Interface

`ICalSet4`
### Name (CalKit) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns a name for the selected calibration kit.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>calKit.Name = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>calKit</code></td>
<td>A CalKit <em>(object)</em>.</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(string)</em> - Calibration Kit name. Any string name, can include numerics, period, and spaces; any length (although the dialog box display is limited to about 30 characters).</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>calKit.Name = &quot;MyCalKit&quot;</code> <em>(Write)</em></td>
</tr>
<tr>
<td></td>
<td><code>KitName = calKit.Name</code> <em>(Read)</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Name(BSTR *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Name(BSTR newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalKit</td>
</tr>
</tbody>
</table>
### Name (PathConfig) Property

**Description**  
Returns the name of the current configuration only if NO individual element settings had been changed since selecting or storing a configuration. When element settings change, the path configuration name is cleared.

**VB Syntax**  
```
name = pathConfig.Name
```

**Variable (Type) - Description**

- **name**  
  (String) Variable to store the returned configuration name.

- **pathConfig**  
  A PathConfiguration (object)

**Return Type**  
String

**Default**  
'default' - name of the default factory configuration

**Examples**  
```
nname=pathConf.Name
```

**C++ Syntax**  
```c++
HRESULT get_Name(BSTR* ppName)
```

**Interface**  
IPathConfiguration

---

Last Modified:

14-Dec-2006   MX New topic
### Name Property

**Description**  
Returns the name of the current element object

**VB Syntax**  
```
name = pathElement.Name
```

**Variable**  
**Type** - Description

```
name  (String) Variable to store the returned element name.
```

*pathElement*  
A *PathElement* (object)

**Return Type**  
String

**Default**  
Not Applicable

**Examples**  
```
name=pathElement.Name
```

**C++ Syntax**  
```
HRESULT get_Name(BSTR* ppName)
```

**Interface**  
IPathElement

---

Last Modified:

15-Dec-2006   MX New topic
Name Property

Description  Returns the name of this FOM range object.

VB Syntax  

\[
\text{value} = \text{FOMRange.Name}
\]

Variable  (Type) - Description

value  (string) - Variable to store the returned range name.

FOMRange  An FOMRange (object)

Return Type  String

Default  Not Applicable

Examples  

Rname = fomRange.Name 'Read

C++ Syntax  HRESULT get_Name(BSTR *pRName)

Interface  IFOMRange

Last Modified:

8-Mar-2007  Major Modifications
## Name (Measurement) Property

**Description**
Sets (or returns) the Name of the measurement. Measurement names must be unique among the set of measurements. Measurement names cannot be an empty string.

**Note:** This is the same name as trace.Name; when one changes, the other changes.

### VB Syntax
```
meas.Name = value
```

### Variable
- **meas** (A Measurement **object**)
- **value** (**string**) - A user defined name of the measurement

### Return Type
String

### Default
"CH1_S11_1" - name of the default measurement

### Examples
```
meas.Name = "Filter BPass" 'Write

MName = meas.Name 'Read
```

### C++ Syntax
```
HRESULT get_Name(BSTR *pVal)
HRESULT put_Name(BSTR newVal)
```

### Interface
IMeasurement
### Name (trace) Property

**Description**
Sets or returns the name of the Trace. Use the trace name to identify the trace and refer to the trace in the collection.

*Note:* This is the same name as meas.Name; when one changes, the other changes.

<table>
<thead>
<tr>
<th>VB Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>trac.Name = value</code></td>
<td><em>trac</em> is a trace (object) and <em>value</em> is a string representing the trace name.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trac</td>
<td>A Trace (object)</td>
</tr>
<tr>
<td>value</td>
<td>(String) Trace name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>String</th>
</tr>
</thead>
</table>

| Default | "CH1_S11_1" - name of the default measurement |

<table>
<thead>
<tr>
<th>Examples</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>trace.Name = &quot;myTrace&quot;</code></td>
<td><em>Write</em></td>
</tr>
<tr>
<td><code>traceName = Name.Trace</code></td>
<td><em>Read</em></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C++ Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT put_Name(BSTR name)</td>
<td><em>Write</em></td>
</tr>
<tr>
<td>HRESULT get_Name(BSTR *name)</td>
<td><em>Read</em></td>
</tr>
</tbody>
</table>

**Interface**
ITrace
NetworkFilename Property

**Description**
Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement.

**VB Syntax**
`object.NetworkFilename(n) = filename`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object SMCTYPE (object)</code> or <code>VMCTYPE (object)</code></td>
<td></td>
</tr>
</tbody>
</table>

- `n` (Integer) Apply network to input or output of mixer. Choose from:
  - 1 - Input of mixer
  - 2 - Output of mixer

- `filename` (String) Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

**Return Type**
String

**Default**
Not Applicable

**Examples**

```
VMC.Filename(2) = "C:\Program Files\Agilent\Network Analyzer\Documents\WaveguideAdapt.S2P"
```

**C++ Syntax**

```cpp
HRESULT put_NetworkFilename(short networkNum, BSTR filename);
HRESULT get_NetworkFilename(short networkNum, BSTR *filename);
```

**Interface**

- SMCTYPE2
- VMCTYPE2

---

Last Modified:

29-Feb-2008 Several edits
NetworkMode Property

Description
Allow you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement. Learn more.

VB Syntax
object.NetworkMode(n) = value

Variable (Type) - Description
object SMCType (object) or VMCType (object)

n (Integer) Apply network to input or output of mixer. Choose from:
1 - Input of mixer
2 - Output of mixer

value (Enum as ENetworkMode) Choose from:
- NO_NETWORK - Do nothing with effects of S2P file
- EMBED_NETWORK - Add effects of S2P file from the measurement results.
- DEEMBED_NETWORK - Remove effects of S2P file from the measurement results.

Return Type
Enum

Default
NO_NETWORK

Examples
VMC.NetworkMode = EMBED_NETWORK

C++ Syntax
HRESULT put_NetworkMode(short networkNum, enum ENetworkMode networkMode);
HRESULT get_NetworkMode(short networkNum, enum ENetworkMode *networkMode);

Interface
SMCType2
VMCType2
## NoiseAverageFactor Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the averaging of the noise receiver.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>noise.NoiseAverageFactor = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>noise</code></td>
<td>A <a href="#">NoiseFigure</a> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(long integer)</strong> - Averaging value. Choose a number between 1 and 99.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1</td>
</tr>
</tbody>
</table>
| **Examples** | `noise.NoiseAverageFactor = 10` *Write*
| | `AvgNoise = noise.NoiseAverageFactor` *Read*
| **C++ Syntax** | HRESULT get_NoiseAverageFactor(long* pVal) |
| | HRESULT put_NoiseAverageFactor(long newVal) |
| **Interface** | INoiseFigure |

Last Modified:

29-May-2007    MN New topic
NoiseAverageState Property

Description  Turns Noise Averaging ON and OFF.

VB Syntax  

```
noise.NoiseAverageState = value
```

Variable  

- **noise**  (Type) - Description
  
  A NoiseFigure (object)

- **value**  (boolean) - Averaging state.
  
  - **False** - Turns Noise Averaging OFF
  - **True** - Turns Noise Averaging ON

Return Type  Boolean

Default  OFF

Examples  

```
noise.NoiseAverageState = OFF 'Write
NoiseAvgState = noise.NoiseAverageState 'Read
```

C++ Syntax  

```
HRESULT get_NoiseAverageState(VARIANT_BOOL* on);
HRESULT put_NoiseAverageState(VARIANT_BOOL on);
```

Interface  INoiseFigure

Last Modified:

21-Jun-2007  MX New topic
### NoiseBandwidth Property

**Description**
Set the bandwidth of the noise receiver.

**VB Syntax**

```vbnet
noise.NoiseBandwidth = value
```

**Variable**

- **Type** - Description
- `noise` A [NoiseFigure](#) (object)
- `value` (double) Bandwidth value. Choose from:
  - 800 KHz, 2 MHz, 4 MHz, 8 MHz, or 24 MHz.
  - Or the numerical equivalent, such as 8e6 and so forth.
  - If the value does not match one of these, it is rounded up to the next legal value.

**Return Type**
Double

**Default**
4 MHz

**Examples**

```vbnet
noise.NoiseBandwidth = 2E6 'Write

NoiseBW = noise.NoiseBandwidth 'Read
```

**C++ Syntax**

```c++
HRESULT get_NoiseBandwidth(double *pVal);

HRESULT put_NoiseBandwidth(double newVal);
```

**Interface**
INoiseFigure

---

**Last Modified:**
20-Sep-2007    MX New topic
**NoiseGain Property**

**Description**
Sets and reads the gain state of the noise receiver.

**VB Syntax**

```
noise.NoiseGain = value
```

**Variable** *(Type) - Description*

- **noise**
  A [NoiseFigure](#) *(object)*

- **value** *(long integer) - Gain value. Choose from:
  - 0 Low Gain  select if the gain of your DUT is relatively high (>35 dB).
  - 15 Medium Gain  select if the gain of your DUT is about average (20 dB to 45 dB).
  - 30 High Gain ..select if the gain of your DUT is relatively low (<30 dB).

If the value does not match one of these, it is rounded up to the next legal value.

Learn more about Noise Receiver Gain setting.

**Return Type**
Long Integer

**Default**
30

**Examples**

```
noise.NoiseGain = 30 'Write
GainNoise = noise.NoiseGain 'Read
```

**C++ Syntax**

```
HRESULT get_NoiseGain(long* pVal)
HRESULT put_NoiseGain(long newVal)
```

**Interface**
INoiseFigure

---

**Last Modified:**
29-May-2007  MN New topic
NoiseSourceCalKitType Property

**Description**
Set and read the Cal Kit that will be used for the Noise Source adapter.

An adapter is always necessary to connect a 346C Noise Source to the PNA port 2. Select a Cal Kit that is the same type and gender as the noise source connector.

If the Noise Source mates directly to PNA port 2, then set this type to "None".

**VB Syntax**

```
noise.NoiseSourceCalKitType = value
```

**Variable**

**Type** - Description

- `noise`  A **NoiseCal**(object)
- `value`  (string) Cal Kit type. To read possible cal kit strings for the adapter:

1. Change the port connector type to that of the noise source using: `ConnectorType`
2. Then read the possible cal kit strings for that port using: `CompatibleCalKits`

**Return Type**
String

**Default**
Not applicable

**Examples**

```
nose.NoiseSourceCalKitType = "N4691-60004 ECAL" Write
calKit = noise.NoiseSourceCalKitType 'Read
```

**C++ Syntax**

```
HRESULT get_NoiseSourceCalKitType(BSTR* pValue)
HRESULT put_NoiseSourceCalKitType(BSTR pNewValue)
```

**Interface**
INoiseCal

Last Modified:

29-May-2007  MN New topic
**NoiseSourceCold Property**

**Description**
Sets and returns the temperature of the noise source connector.

**VB Syntax**

```
noise.NoiseSourceCold = value
```

**Variable**

- **noise** *(Type)* - Description
  - **noise**
    - A **NoiseCal** *(object)*
  - **value** *(double)* - Noise source temperature in Kelvin.

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```
noise.NoiseSourceCold = 295 'Write

temp = noise.NoiseSourceCold 'Read
```

**C++ Syntax**

```cpp
HRESULT get_NoiseSourceCold(double* pTemp)
HRESULT put_NoiseSourceCold(double pNewTemp)
```

**Interface**
INOiseCal

---

Last Modified:

21-Jun-2007 MX New topic
## NoiseSourceConnectorType Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the Noise Source connector type and gender. The Agilent 346C has an &quot;APC 3.5 male&quot; connector.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>noise.NoiseSourceConnectorType = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>noise</code></td>
<td>A <strong>NoiseCal (object)</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(string)</strong> Connector type. Use <strong>ValidConnectorType</strong> to return a list of valid connector types.</td>
</tr>
<tr>
<td>Return Type</td>
<td><strong>String</strong></td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>noise.NoiseSourceConnectorType = &quot;APC 3.5 male&quot;</code> <strong>Write</strong></td>
</tr>
<tr>
<td></td>
<td><code>connector = noise.NoiseSourceConnectorType</code> <strong>Read</strong></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_NoiseSourceConnectorType(BSTR* pConnectorType)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_NoiseSourceConnectorType(BSTR pConnectorType)</code></td>
</tr>
<tr>
<td>Interface</td>
<td><strong>INoiseCal</strong></td>
</tr>
</tbody>
</table>

---

**Last Modified:**

29-May-2007   MN New topic
## NoiseSourceState Property

**Description**  
Sets and Reads the state of the noise source (ON and OFF).

**VB Syntax**  
```vb
app.NoiseSourceState = state
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>Application (object)</td>
<td></td>
</tr>
</tbody>
</table>
| state    | boolean   | False (0) - Turns Noise Source OFF  
|          |           | True (1) - Turns Noise Source ON  |

**Return Type**  
Boolean

**Default**  
False

**Examples**

```vb
app.NoiseSourceState = True 'Write

coupl = app.NoiseSourceState 'Read
```

**C++ Syntax**

```c++
HRESULT put_NoiseSourceState(VARIANT_BOOL bState)
HRESULT get_NoiseSourceState(VARIANT_BOOL *bState)
```

**Interface**  
IApplication13

---

Last Modified:

29-May-2007     MN New topic
NoiseTuner Property

### Description
Sets and returns the noise tuner identifier, which is an ECal model and serial number string.

### VB Syntax

```vbnet
noise.NoiseTuner = value
```

### Variable

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>noise</td>
<td>A NoiseFigure (object)</td>
</tr>
<tr>
<td>value</td>
<td>(string) Noise Tuner. Return the connected ECal identifiers by sending GetCalKitTypeString and passing the module number.</td>
</tr>
</tbody>
</table>

### Return Type
String

### Default
Not Applicable

### Examples

```vbnet
noise.NoiseTuner = "N4691-60004 ECal 02822" 'Write

noiseT = noise.NoiseTuner 'Read
```

### C++ Syntax

```cpp
HRESULT get_NoiseTuner(BSTR* pValue)
HRESULT put_NoiseTuner(BSTR pNewValue)
```

### Interface
INoiseFigure

---

Last Modified:

- 18-Jun-2007  MN New topic
### NoiseTunerIn Property

**Description**
Sets and returns the port identifier of the ECal noise tuner that is connected to the PNA Source.

**VB Syntax**

```vbnet
noise.NoiseTunerIn = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>noise</td>
<td>A <strong>NoiseFigure</strong> (object)</td>
</tr>
<tr>
<td>value</td>
<td>(string) Noise Tuner port identifier that is connected to the PNA source, as it is labeled on the ECal module. For example, for 2-port ECal modules, choose either &quot;A&quot; or &quot;B&quot;.</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Not Applicable

**Examples**

```vbnet
noise.NoiseTunerIn = "A" 'Write

EcalPort = noise.NoiseTunerIn 'Read
```

**C++ Syntax**

- `HRESULT get_NoiseTunerIn(BSTR* pValue)`
- `HRESULT put_NoiseTunerIn(BSTR pNewValue)`

**Interface**
INoiseFigure

---

**Last Modified:**

- 29-May-2007  MN New topic
### NoiseTunerOut Property

**Description**
Sets and returns the port identifier of the ECaI noise tuner that is connected to the DUT.

**VB Syntax**
```
noise.NoiseTunerOut = value
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>noise</code></td>
<td>A NoiseFigure (object)</td>
<td></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(string)</td>
<td>Noise Tuner port identifier that is connected to the DUT, as it is labeled on the ECal module. For example, for 2-port ECal modules, choose either &quot;A&quot; or &quot;B&quot;.</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Not Applicable

**Examples**
```
noise.NoiseTunerOut = "A" 'Write
EcalPort = noise.NoiseTunerOut 'Read
```

**C++ Syntax**
```
HRESULT get_NoiseTunerOut(BSTR* pValue)
HRESULT put_NoiseTunerOut(BSTR pNewValue)
```

**Interface**
INoiseFigure

---

**Last Modified:**
29-May-2007  MN New topic
### NominalIncidentPowerState Property

**Description**
Toggles the Nominal Incident Power setting ON and OFF. This setting is ONLY to be used with SMC measurements, not VMC. [Learn more about Nominal Incident Power](#).

**VB Syntax**
```vbnet
mixer.NominalIncidentPowerState = bool
```

**Variable**
- **(Type)** - Description
  - `mixer` A Mixer *(object)*
  - `bool` *(boolean)* - Nominal Incident Power State. Choose from:
    - 1 -(True) Turn nominal incident power ON
    - 0 -(False) Turn nominal incident power OFF

**Return Type**
Boolean

**Default**
0 -(False)

**Examples**
```vbnet
mixer.NominalIncidentPowerState = True 'sets NominalIncidentPowerState to ON
```

**C++ Syntax**
```cpp
HRESULT get_NominalIncidentPowerState(VARIANT_BOOL *pVal)
HRESULT put_NominalIncidentPowerState(VARIANT_BOOL val);
```

**Interface**
IMixer4
**NormalizePoint Property**

**Description**
Sets and returns the sweep data point around which to perform broadband and precise tuning.

**VB Syntax**
```vbnet
embedLO.NormalizePoint = value
```

**Variable**  
**EmbeddedLO** (object)

**value**  
(Long) Mixer Sweep data point. Choose a data point number between 1 and the max number of data points in the sweep that has the least amount of expected noise.

**Return Type**  
(Long)

**Default**  
Center point in the sweep span

**Examples**
```vbnet
embedLO.NormalizePoint = 101 'write
value = embedLO.NormalizePoint 'read
```

**C++ Syntax**
```cpp
HRESULT get_NormalizePoint(long *point);
HRESULT put_NormalizePoint(long point);
```

**Interface**
IEEmbeddedLO

---

Last Modified:

13-Apr-2007  MX New topic
Description
Returns the Number of the measurement. Measurement numbers are assigned internally.

Note: Measurement numbers are NOT the same as their number in the Measurements collection. Measurement number is used to identify the measurement associated with an event.

This property is used to identify measurements when events occur through the OnMeasurementEvent callback. For example:

OnMeasurementEvent (naEventId_MSG_LIMIT_FAILED, 3)

VB Syntax
measNum = meas.Number

Variable (Type) - Description
measNum (long) - variable to store the measurement number
meas A Measurement (object)

Return Type Long Integer

Default "1" - number of the default measurement

Examples
measNum = meas.Number

C++ Syntax
HRESULT get_Number(long *MeasurementNumber)

Interface IMeasurement
## Number Property

**Description**

Returns the number of the Auxiliary Trigger connector pair currently being used with the instance of the **AuxTrigger** object. Set the trigger pair with the AuxTrig object.

**VB Syntax**

```
value = auxTrig.Number
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>auxTrig</td>
<td>An <strong>AuxTrigger</strong> (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Long Integer) - Connector pair. PNA-X returns 1 or 2. E836x returns 1.</td>
</tr>
</tbody>
</table>

**Return Type**

Single

**Default**

Not Applicable

**Examples**

```
value = auxTrig.Number 'Read the value
```

**C++ Syntax**

```
HRESULT get_Number(long *auxID);
```

**Interface**

IAuxTrigger

---

Last Modified:

14-Dec-2006  MX New topic
**NumberOfFrequencyPoints Property**

- **Description**: Set and read the number of frequency points. Applies to all acquisition modes.

- **VB Syntax**: 
  
  ```vbnet
gca.NumberOfFrequencyPoints = value
  ```

- **Variable**
  
  - **Type**: (Type) - Description
  
  - `gca` A GainCompression (object)
  
  - `value` (integer) - Frequency points. Do not exceed the max number of points.
    
    For 2D sweeps, total = frequency x power. Max = 20,001
    
    For Smart sweep, total = frequency. Max = 10,000.

- **Return Type**: Integer

- **Default**: 201

- **Examples**

  ```vbnet
gca.NumberOfFrequencyPoints = 101 'Write
  ```

  ```vbnet
  freqPts = gca.NumberOfFrequencyPoints 'Read
  ```

- **C++ Syntax**: 

  ```cpp
  HRESULT get_NumberOfFrequencyPoints(int* pVal)
  HRESULT put_NumberOfFrequencyPoints(int newVal)
  ```

- **Interface**: IGainCompression

---

Last Modified:

11-Sep-2007   MX New topic
Write/Read

**NumberOfPoints Property**

**Description**
Sets or returns the Number of Points of the channel.
Sets or returns the Number of Points of the segment.
See the [Measurement2 Interface](#) to learn how this method differs from `meas.NumberOfPoints`.

**VB Syntax**

```
object.NumberOfPoints = value
```

**Variable (Type) - Description**

- **object**
  - Channel *(object)*
  - or
  - CalSet *(object)* - Read-only property

- **value** *(long)* - Number of Points.
  - For channel, choose any number from 1 to **20001**.
  - For segment, the total number of points in all segments cannot exceed **20001**. A segment can have as few as 1 point.

**Return Type**
Long Integer

**Default**
201 for channel
21 for segment

**Examples**

```
chan.NumberOfPoints = 201 'sets the number of points for all measurements in the channel. -Write

numofpts = chan.NumberOfPoints 'Read
```

**C++ Syntax**

```
HRESULT get_NumberOfPoints(long *pVal)
HRESULT put_NumberOfPoints(long newVal)
```

**Interface**

- IChannel
- ISegment
- ICalSet3

**Last Modified:**
21-Jun-2007  Increased max
**NumberOfPoints Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of data points of the measurement. To understand how this property is useful, see <a href="#">IMeasurement2 Interface</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = meas.NumberOfPoints</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Long)</em> - variable to store the returned value</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><strong>Print</strong> <code>meas.NumberOfPoints</code> <em>'prints the number of data points'</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_NumberOfPoints(long *pVal);</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
**NumberOfPorts Property**

**Description**  
Returns the number of hardware source ports on the PNA.

**VB Syntax**  
`value = app.NumberOfPorts`

**Variable**  
(`Type`) - Description

- **app**  
  An `Application` (object)

- **value**  
  `(long integer)` - variable to contain the returned value

**Return Type**  
(long integer)

**Default**  
Not Applicable

**Examples**  
`iNumPorts = app.NumberOfPorts`

**C++ Syntax**  
`HRESULT NumberOfPorts( long* NumPorts)`

**Interface**  
IApplication
### NumberOfPorts Property

**Description**
Returns the number of ports on the specified testset. Returns 0 if no test set is connected.

**VB Syntax**

```vbnet
value = tset.NumberOfPorts
```

**Variable (Type) - Description**

- **value** (Long) variable to store the returned information.
- **tset** A `TestsetControl` object.
  OR
  An `E5091Testset` object.

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**

```vbnet
value = testset1.NumberOfPorts
```

See [E5091A Example Program](#)

See [External Testset Program](#)

**C++ Syntax**

```c++
HRESULT get_NumberOfPorts(long *numberOfPorts);
```

**Interface**
- ITestsetControl
- IE5091Testset
## NumberOfPowerPoints Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Set and read the number of data points in each power sweep. Applies ONLY to 2D acquisition modes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>gca.NumberOfPowerPoints = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><code>&lt;object&gt; gains A GainCompression (object)</code></td>
</tr>
</tbody>
</table>
| `value`     | (integer) - Power points. Do not exceed the max number of points.  
For 2D sweeps, total = frequency x power. Max = 20,001  
For Smart sweep, total = frequency. Max = 10,000. |
| Return Type | Integer                                                                                         |
| Default     | 26                                                                                             |
| Examples    | `gca.NumberOfPowerPoints = 31 'Write`                                                           |
|             | `pwrPts = gca.NumberOfPowerPoints 'Read`                                                         |
| C++ Syntax  | HRESULT get_NumberOfPowerPoints(int* pVal)                                                      |
|             | HRESULT put_NumberOfPowerPoints(int newVal)                                                     |
| Interface   | IGainCompression                                                                               |

Last Modified:  
11-Sep-2007  MX New topic
**NumberOfSweeps Property**

**Description**
Returns the number of tuning sweeps used for the latest embedded LO measurement.

**VB Syntax**

```
value = embedLODiag.NumberOfSweeps
```

**Variable**

*value* *(Type)* - Description

(value) Variable to store the returned data.

**embedLODiag**
An EmbeddedLODiagnostic *(object)*

**Return Type** *(Long)*

**Default**
Not Applicable

**Examples**

```
data= embedLODiag.NumberOfSweeps 'read
```

**C++ Syntax**

```
HRESULT get_NumberOfSweeps(long * numSweeps);
```

**Interface**
IEembedLODiagnostic

Last Modified:

12-Apr-2007    MX New topic
**Offset Property**

**Description**
Sets and returns the offset value to be used when coupling this range to the primary range.

This setting is valid only if the specified range is **coupled** to the primary range.

**VB Syntax**

```
FOMRange.Offset = value
```

**Variable**  
*(Type)* - Description

- **object** An FOMRange *(object)*
- **value** *(Double)* - Offset value. *(Unitless)*

**Return Type**
Double

**Default**
0

**Examples**

```
fomRange.Offset = 1e9  'Write
```

```
Offs = fomRange.Offset  'Read
```

**C++ Syntax**

```
HRESULT get_Offset(double *pVal)
HRESULT put_Offset(double pVal)
```

**Interface**
IFOMRange

---

Last Modified:  
8-Mar-2007 Modified links.
### OffsetReceiverAttenuator Property

**Description**
Set and return whether to offset the reference receiver by the amount of receiver attenuation. [Learn more.](#)

This setting remains until changed again using this [or equivalent SCPI](#) command, or until the hard drive is changed or reformatted.

**VB Syntax**

```vbnet
pref.OffsetReceiverAttenuator = value
```

**Variable**

**Type** - Description

- **pref**
  - A [Preferences](#) (object)
- **value**
  - (Boolean) - Choose from:
    - **False** Do NOT offset the test port receivers.
    - **True** Offset the test port receivers.

**Return Type**

Boolean

**Default**

- **True** PNA-X models
- **False** E836xB and PNA-L models

**Examples**

```vbnet
pref.OffsetReceiverAttenuator = 1 'Write
Rcvroffset = pref.OffsetReceiverAttenuator 'Read
```

**C++ Syntax**

```cpp
HRESULT get_OffsetReceiverAttenuator(VARIANT_BOOL * val);
HRESULT put_OffsetReceiverAttenuator(VARIANT_BOOL val);
```

**Interface**

IPreferences6

---

**Last Modified:**

14-Jan-2007   MX New topic
OffsetSourceAttenuator Property

**Description**
Set and return whether to mathematically offset the reference receivers by the amount of source attenuation. Learn more.

This setting remains until changed again using this (or equivalent SCPI) command, or until the hard drive is changed or reformatted.

**VB Syntax**
```
pref.OffsetSourceAttenuator = value
```

**Variable**
- **(Type)** - Description
  - `pref` (Type) - Description
    - `pref` A Preferences (object)
  - `value` (Boolean) - Choose from:
    - `False` Do NOT offset the reference receivers.
    - `True` Offset the reference receivers.

**Return Type**
Boolean

**Default**
- **True** PNA-X models
- **False** E836xB and PNA-L models

**Examples**
```
pref.OffsetSourceAttenuator = 1 'Write

Rcvroffset = pref.OffsetSourceAttenuator 'Read
```

**C++ Syntax**
```
HRESULT get_OffsetSourceAttenuator(VARIANT_BOOL * val);
HRESULT put_OffsetSourceAttenuator(VARIANT_BOOL val);
```

**Interface**
IPreferences6

Last Modified:
14-Jan-2007  MX New topic
Read/Write

OmitIsolation Property

Description
Sets and returns whether Isolation portion of the calibration will be performed or not.

VB Syntax
obj.OmitIsolation = bool

Variable
(Type) - Description
obj | SMCType (object)
or
     | VMCType (object)
bool | (Boolean)
     | True - Isolation is NOT performed
     | False - Isolation is performed

ReturnType
Boolean

Default
True

Examples
value = SMC.OmitIsolation

C++ Syntax
HRESULT put_OmitIsolation (VARIANT_BOOL bState)
HRESULT get_OmitIsolation (VARIANT_BOOL *bState)

Interface
SMCType
VMCType
# OneReadoutPerTrace Property

**Description**
Either show marker readout of only the active trace or all of the traces simultaneously.

**VB Syntax**

```
win.OneReadoutPerTrace = state
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>win</code></td>
<td>A <code>NAWindow</code> (object)</td>
</tr>
</tbody>
</table>

**Value**

<table>
<thead>
<tr>
<th>(boolean)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>True</code></td>
<td>Shows the readout of only the active marker for each trace.</td>
</tr>
<tr>
<td><code>False</code></td>
<td>Shows up to 5 marker readouts per trace, up to 20 total readouts.</td>
</tr>
</tbody>
</table>

**Return Type**

Boolean

**Default**

False

**Examples**

```
win.OneReadoutPerTrace = True 'Write
State = app.ActiveNAWindow.OneReadoutPerTrace 'Read
```

**C++ Syntax**

```
HRESULT get_OneReadoutPerTrace(VARIANT_BOOL *pVal)
HRESULT put_OneReadoutPerTrace(VARIANT_BOOL newVal)
```

**Interface**

`INAWindow`
## Options Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a string identifying the analyzer option configuration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>value = app.Options</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>app</code></td>
<td>An <strong>Application</strong> <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(string)</em> - variable to contain the returned string</td>
</tr>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>availOptions = app.Options</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT Options(BSTR* OptionString)</code></td>
</tr>
<tr>
<td>Interface</td>
<td><strong>IApplication</strong></td>
</tr>
</tbody>
</table>
Write/Read

OrientECALModule Property

**Description**
Specifies if the PNA should perform orientation of the ECal module during calibration. Orientation is a technique by which the PNA automatically determines which ports of the module are connected to which ports of the PNA. Orientation begins to fail at very low power levels or if there is much attenuation in the path between the PNA and the ECal module.

**Note:** For guided calibrations where Orient is OFF and the same ECal module is used in more than one Connection Step, you are not allowed to specify how the ECal module is connected. Instead, the PNA determines the orientation. The PNA does not verify that you made the connection properly.

This setting remains until the PNA is restarted or this command is sent again.

This command, and ECALPortMapEx, can be used to perform ECal using the Guided Calibration interface.

**VB Syntax**
```vbnet
cal.OrientECALModule = value
```

**Variable**
*(Type) - Description*
- **cal**: A Calibrator (object)
- **value**: (boolean)
  - **False**: DoECAL1PortEX and DoECAL2PortEx methods will use the value of the ECALPortMapEx property to determine the port connections.
  - **True**: DoECAL1PortEX and DoECAL2PortEx methods will use auto Orientation technique to determine port connections.

**Return Type**
Boolean

**Default**
True

**Examples**
Dim cal As Calibrator
Dim bOrient As Boolean
Set cal = PNAapp.ActiveChannel.Calibrator
cal.OrientECALModule = False 'Write
bOrient = cal.OrientECALModule 'Read

**C++ Syntax**
```cpp
HRESULT put_OrientECALModule(VARIANT_BOOL bOrient);
HRESULT get_OrientECALModule(VARIANT_BOOL *bOrient);
```

**Interface**
ICalibrator3

Last Modified:
7-May-2007 Added note about Guided Cal.
**OutputFixedFrequency Property**

**Description**
Sets or returns the mixer output fixed frequency value.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**

```vbnet
mixer.OutputFixedFrequency = value
```

**Variable (Type) - Description**

- `mixer` A Mixer *(object)*
- `value` *(double)* - Output Fixed Frequency in Hertz.

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```vbnet
Print mixer.OutputFixedFrequency 'prints the output fixed frequency value of the mixer.
```

**C++ Syntax**

```cpp
HRESULT get_OutputFixedFrequency(double *pVal)
HRESULT put_OutputFixedFrequency(double newVal)
```

**Interface**
IMixer3

---

**Last Modified:**
4-Mar-2008  Added note.
### Description
Switches an input to one of the valid outputs on the specified E5091A. The following are valid input/output combinations. If a combination other than these are sent, an "invalid argument" error will occur.

<table>
<thead>
<tr>
<th>Input</th>
<th>Valid Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td>T1 - If Port 2 already is connected to T1, then Port 2 will be switched to T2.</td>
</tr>
<tr>
<td>2</td>
<td>T1 - If Port 1 already is connected to T1, then Port 1 will be switched to A.</td>
</tr>
<tr>
<td></td>
<td>T2</td>
</tr>
<tr>
<td>3</td>
<td>R1+</td>
</tr>
<tr>
<td></td>
<td>R2+</td>
</tr>
<tr>
<td></td>
<td>R3+ - If option 007 (7port), R2 is selected.</td>
</tr>
<tr>
<td>4</td>
<td>R1-</td>
</tr>
<tr>
<td></td>
<td>R2-</td>
</tr>
<tr>
<td></td>
<td>R3- - If option 007 (7port), R2 is selected.</td>
</tr>
</tbody>
</table>

**Note:** Do not confuse the similar Testset.**OutputPorts Property**, which sets or gets the port mapping for ALL ports.

### VB Syntax
```
testsets(1).OutputPort (chNum,input) = output
```

### Variable *(Type)* - Description
- **testsets(1)**: An item from **Testsets (collection)**
  - Learn how to [identify a testset in the collection](#).
- **chNum**: *(Long)* Channel number of the measurement.
- **input**: *(Long)* Testset Input port. Choose from 1|2|3|4.
- **output**: *(Enum as NAE5091OutputPort)* Output port to switch to specified Input. Choose from:
  - 0 or **naE5091PortA** - Port A
  - 1 or **naE5091PortT1** - Port T1
  - 2 or **naE5091PortT2** - Port T2
<table>
<thead>
<tr>
<th>3 or naE5091PortR1</th>
<th>Port R1</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 or naE5091PortR2</td>
<td>Port R2</td>
</tr>
<tr>
<td>5 or naE5091PortR3</td>
<td>Port R3</td>
</tr>
</tbody>
</table>

**Return Type**  
Enum

**Default**  
Not Applicable

**Examples**  
See Example Program

**C++ Syntax**  
HRESULT get_OutputPort(long channelNum, long inputPort, E5091OutputPort *outPort);
HRESULT put_OutputPort(long channelNum, long inputPort, E5091OutputPort outPort);

**Interface**  
IE5091Testsets
OutputPorts (Cal Set) Property

Description
Returns the port mapping for the Cal Set.

VB Syntax
`portMap = calset.OutputPorts`

Variable (Type) - Description
`portMap` (String) Variable to store the returned string. The returned values are the physical ports. The POSITION of the returned values corresponds to the logical ports.

For example, with an N44xx test set, if the returned string is "PNA 1,TS 2,PNA 2, TS 4" this means:

- PNA 1 is assigned to logical port 1
- TS 2 is assigned to logical port 2
- PNA 2 is assigned to logical port 3
- TS 4 is assigned to logical port 4

`calset` A `Cal Set` object.

Return Type String

Default Depends on the test set.

Example `portMap = calset.OutputPorts`

C++ Syntax
`HRESULT get_OutputPorts(BSTR *mapping);`

Interface `ICalset5`

Last modified:

9/18/06 MQ Added for multiport
### Description
Sets or gets the port mappings for ALL ports. An “invalid argument” error will occur if you attempt to set an illegal port combination.

Refer to your testset documentation for valid port combinations.

**Note:** Do not confuse the similar E5091.**OutputPort Property**, which sets or gets the port mapping for a single port.

### VB Syntax
```
tset.OutputPorts(chNum) = portList
```

### Variable
**(Type)** - **Description**

- `tset` A `TestsetControl` object.
- `chNum` *(Long)* Channel number of the measurement.
- `portList` *(String)* A comma-separated list of port mappings. Spaces are ignored at the beginning and end of this text, and before or after commas. Space characters in other locations are not ignored.

### Return Type
String

### Default
Not Applicable

### Examples
See External Testset Program

### C++ Syntax
```
HRESULT get_OutputPorts(long channelNum, BSTR *outPorts);
HRESULT put_OutputPorts(long channelNum, BSTR outPorts);
```

### Interface
`ITestsetControl`
OutputRangeMode Property

**Description**
Sets or returns the Output sweep mode.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the **Calculate** and **Apply** commands as you would do from the user interface.

**VB Syntax**
mixer.OutputRangeMode = value

**Variable**
(Type) - Description
- **mixer**: A **Mixer** (object)
- **value**: (Enum as **MixerRangeMode**) - Output sweep mode. Choose from:
  - 0 - **mixSwept**
  - 1 - **mixFixed**

**Return Type**
Enum

**Default**
0 - mixSwept

**Examples**
mixer.OutputRangeMode = mixSwept

**C++ Syntax**
HRESULT get_OutputRangeMode(long *pVal)
HRESULT put_OutputRangeMode(long newVal)

**Interface**
IMixer6

---

Last Modified:

4-Mar-2008   Added note.
### OutputSideband Property

**Description**
Sets or returns the value of the output sideband, high or low.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the [Calculate](#) and [Apply](#) commands as you would do from the user interface.

**VB Syntax**

```vbnet
mixer.OutputSideband = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mixer</td>
<td>A Mixer (object)</td>
</tr>
<tr>
<td>value</td>
<td>(FCASideBand) -</td>
</tr>
<tr>
<td></td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - LOW</td>
</tr>
<tr>
<td></td>
<td>1 - HIGH</td>
</tr>
</tbody>
</table>

**Return Type**
FCASideBand

**Default**
LOW

**Examples**

```vbnet
Print mixer.OutputSideband 'prints the value of the OutputSideband
```

**C++ Syntax**

```c++
HRESULT get_OutputSideband(FCASideBand *pVal)
HRESULT put_OutputSideband(FCASideBand newVal)
```

**Interface**
IMixer

---

Last Modified:
4-Mar-2008   Added note.
**OutputStartFrequency Property**

**Description**
Sets or returns the mixer output start frequency.
If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

**VB Syntax**
mixer.OutputStartFrequency = value

**Variable (Type) - Description**

* mixer: A Mixer (object)
* value: (double) - Output Start Frequency in Hertz.

**Return Type**
Double

**Default**
Not Applicable

**Examples**
Print mixer.OutputStartFrequency 'prints the value of the OutputStartFrequency

**C++ Syntax**
HRESULT get_OutputStartFrequency(double *pVal)
HRESULT put_OutputStartFrequency(double newVal)

**Interface**
IMixer

---

Last Modified:

4-Mar-2008  Added note.
## OutputStopFrequency Property

### Description
Sets or returns the mixer Output Stop frequency.

If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

### VB Syntax
```
mixer.OutputStopFrequency = value
```

### Variable
- **mixer** - A Mixer (object)
- **value** - (double) - Output stop frequency in Hertz.

### Return Type
Double

### Default
Not Applicable

### Examples
```vbnet
Print mixer.OutputStopFrequency 'prints the value of the OutputStopFrequency
```

### C++ Syntax
```cpp
HRESULT get_OutputStopFrequency(double *pVal)
HRESULT put_OutputStopFrequency(double newVal)
```

### Interface
IMixer

---

**Last Modified:**

4-Mar-2008   Added note.
### Parameter Property

**Description**  
Returns the measurement Parameter. To change the parameter, use `meas.ChangeParameter`

**VB Syntax**  
`measPar = meas.Parameter`

**Variable**  
- **`measPar`** *(string)* - Variable to store Parameter string
- **`meas`** A Measurement *(object)*

**Return Type**  
String

**Default**  
Not applicable

**Examples**  
```
measPar = meas.Parameter 'Read
```

**C++ Syntax**  
`HRESULT get_Parameter(BSTR *pVal)`

**Interface**  
IMeasurement
### Parameter (Embedded LO) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the name of the parameter of the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = embedLODiag.Parameter (n)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><strong>(String) Variable to store the returned data.</strong></td>
</tr>
<tr>
<td><code>embedLODiag</code></td>
<td>An <strong>EmbeddedLODiagnostic (object)</strong></td>
</tr>
<tr>
<td><code>n</code></td>
<td><strong>(Long) Tuning sweep number. Use <strong>NumberOfSweeps</strong> to find the number of sweeps taken.</strong></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>data = embedLO.Parameter 3 'read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Parameter(long sweep, BSTR * param);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEmbeddedLODiagnostic</td>
</tr>
</tbody>
</table>

**Last Modified:**

13-Apr-2007    MX New topic
Read-only

Parent Property

**Description**  Returns a handle to the parent object of the collection object being referred to in the statement. The parent property allows the user to traverse from an object back up the object hierarchy.

**VB Syntax**  
`collection.Parent`

**Variable (Type) - Description**

- `collection`  
  - `CalFactorSegments collection`
  - `Channels collection`
  - `E5091Testset Collection`
  - `ExternalTestsets Collection`
  - `Measurements collection`
  - `NaWindows collection`
  - `PowerLossSegments collection`
  - `PowerSensors collection`
  - `Segments collection`
  - `Traces collection`
  - `PathConfigurationManager`

**Return Type**  Object

**Default**  Not Applicable

**Examples**

```vbnet
parentobj = chans.Parent 'returns a handle to the parent object (Application) of the chans collection. ~Read
```

**C++ Syntax**

```cpp
HRESULT get_Parent(IApplication* *pApplication); //IChannels, IChannel, IMeasurements and INAWindows
HRESULT get_Parent(IChannel* *pChannel); //ITraces
HRESULT get_Parent(INAWindow* *pWindow); //ISegments
HRESULT get_Parent(IPowerSensor* *pSensor); //ICalFactorSegments
HRESULT get_Parent(ISourcePowerCalibrator* *pCalibrator); //IPowerSensors and IPowerLossSegments
```

**Interface**  All listed above
## Read/Write

### PassFailLogic Property

**Description**
Sets and reads the logic of the PassFail line on the **HANDLER IO connector** (pin 33) and **AUX IO connector** (pin 12). [Learn more about Global Pass/Fail.](#)

**Note:** This line is connected to both the Handler IO and Aux IO in the PNA. Therefore, this command will affect both of these connectors in the same way.

### VB Syntax

```vbnet
object.PassFailLogic = value
```

**Variable**

<table>
<thead>
<tr>
<th><strong>(Type)</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code></td>
<td>An Aux I/O or Handler I/O object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum as NARearPanelIOLogic) Choose from:</td>
</tr>
</tbody>
</table>

- **naPositiveLogic** - Causes the PassFail line to have positive logic (high = pass, low = fail).
- **naNegativeLogic** - Causes the PassFail line to have negative logic (high = fail, low = pass).

**Return Type**

Long Integer

**Default**

naPositiveLogic

**Examples**

```vbnet
aux.PassFailLogic = naNegativeLogic 'Write
Text1.Text = aux.PassFailLogic 'Read
```

**C++ Syntax**

```c++
HRESULT put_PassFailLogic ( tagNARearPanelIOLogic Mode );
HRESULT get_PassFailLogic ( tagNARearPanelIOLogic* Mode );
```

**Interface**

IHWAuxIO

IHWMaterialHandlerIO
PassFailMode Property

**Description**
Sets and reads the mode of the PassFail line on the **HANDLER IO connector** (pin 33) and **AUX IO connector** (pin 12). [Learn more about Global Pass/Fail.]

**Note:** This line is connected to both the Handler IO and Aux IO in the PNA. Therefore, this command will affect both of these connectors in the same way.

**VB Syntax**

```vbnet
object.PassFailMode = value
```

**Variable** *(Type) - Description*

- **object** *(object)* - An Aux I/O or Handler I/O object
- **value** *(enum as NAPassFailMode)* - Choose from:
  - 0 - naDefaultPassNoWaitMode - the line stays in PASS state. When a device fails, then the line goes to fail IMMEDIATELY.
  - 1 - naDefaultPassWaitMode - the line stays in PASS state. When a device fails, then the line goes to fail after the Sweep End line is asserted.
  - 2 - naDefaultFailWaitMode - the line stays in FAIL state. When a device passes, then the line goes to PASS state after the Sweep End line is asserted.

**Return Type**
Long Integer

**Default**
0 - naDefaultPassNoWaitMode

**Examples**

```vbnet
HWAuxIO.PassFailMode = naDefaultPassNoWaitMode 'Write
mode = HWAuxIO.PassFailMode 'Read
```

**C++ Syntax**

```cpp
HRESULT put_PassFailMode ( tagNAPassFailMode Mode );
HRESULT get_PassFailMode ( tagNAPassFailMode* Mode );
```

**Interface**
IHWAuxIO
IHWMaterialHandlerIO
### Read/Write

#### PassFailPolicy Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the policy used to determine how global pass/fail is computed. <a href="#">Learn more about Global Pass/Fail</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>object.PassFailPolicy = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td>object</td>
<td><em>(object)</em> - An <em>Aux I/O</em> or <em>Handler I/O</em> object</td>
</tr>
<tr>
<td>value</td>
<td><em>(enum as NAPassFailPolicy)</em> Choose from:</td>
</tr>
<tr>
<td>0 - naPolicyAllTests</td>
<td>- Pass/Fail Status returns PASS if all tests on all measurements pass.</td>
</tr>
<tr>
<td>1 - naPolicyAllMeas</td>
<td>- Pass/Fail Status returns PASS if all measurements have associated tests, and all tests pass. FAIL is returned if even one measurement has no associated limit test. Only those measurements which are not in HOLD mode contribute to the pass/fail result.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>naPolicyAllTests</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>matHndler.PassFailPolicy = naPolicyAllTests 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>policy = aux.PassFailPolicy 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_PassFailPolicy ( tagNAPassFailPolicy Policy);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_PassFailPolicy ( tagNARearPanelIOLogic* Policy);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IHWAuxIO4</td>
</tr>
<tr>
<td></td>
<td>IHWMaterialHandlerIO2</td>
</tr>
</tbody>
</table>
Read/Write

PassFailScope Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and reads the Scope of the PassFail line on the HANDLER IO connector (pin 33) and AUX IO connector (pin 12). Learn more about Global Pass/Fail.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note</strong></td>
<td>The PassFail line is connected to both the Handler IO and Aux IO in the PNA. Therefore, this command will affect both of these connectors in the same way.</td>
</tr>
</tbody>
</table>

**VB Syntax**

```
object.PassFailScope = value
```

<table>
<thead>
<tr>
<th><strong>Variable</strong> (Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>object</strong> (object) - An Aux I/O or Handler IO object</td>
</tr>
<tr>
<td><strong>value</strong> (enum NAPassFailScope) Choose from:</td>
</tr>
<tr>
<td>0 - naChannelScope - The PassFail line returns to its default state before sweeps on the next channel start. (A channel measurement may require several sweeps.)</td>
</tr>
<tr>
<td>1 - naGlobalScope - The PassFail line returns to its default state before the sweeps for the next triggerable channel start.</td>
</tr>
<tr>
<td>The default state of the PassFail line before a measurement occurs and after a failure occurs is set by the PassFailMode property.</td>
</tr>
</tbody>
</table>

**Return Type**

enum NAPassFailScope

**Default**

1 - naGlobalScope

**Examples**

```
HWAuxIO.PassFailScope = naGlobalScope  'Write
scope = HWAuxIO.PassFailScope  'Read
```

**C++ Syntax**

```
HRESULT put_PassFailScope ( tagNAPassFailScope Scope );
HRESULT get_PassFailScope ( tagNAPassFailScope* Scope );
```

**Interface**

IHWAuxIO

IHWMaterialHandlerIO
PassFailStatus Property

Read-Only

Description

Returns the most recent pass/fail status value. Use this command as follows:

1. Set the PNA trigger scope to GLOBAL.
2. Set the PNA trigger source to MANUAL or EXTERNAL.
3. Configure and enable Limit Testing.
4. Trigger the PNA.
5. Use the *OPC? (with SCPIStringParser object) to determine when the sweep is complete.
6. Use the PassFailStatus property to obtain the global pass/fail result.

Learn more about Global Pass/Fail.

VB Syntax

var = object.PassFailStatus

Variable (Type) - Description

var (enum as NAPassFailStatus) Variable to store returned status. One of the following will be returned:

0 - naStatusFail - all measurements not in HOLD mode have been swept, and one or more limit tests failed according to the specified Pass/Fail policy.
1 - naStatusPass - all measurements not in HOLD mode have been swept, and all associated limit tests have passed.
2 - naStatusNone - status cannot be determined because measurements are in progress.

object (object) - An Aux I/O or Handler I/O object

Return Type

Long Integer

Default

Not Applicable

Examples

status = aux.PassFailStatus 'Read

C++ Syntax

HRESULT get_PassFailPolicy ( tagNAPassFailStatus* status);

Interface

IHWAuxIO4
IHWMaterialHandlerIO2
### Path Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Specifies an interface to use for the power meter / sensor during a source power calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pwrMtrInterface.Path = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>pwrMtrInterface</code></td>
<td><em>(object)</em> - A <a href="#">PowerMeterInterface</a> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(enum as NACommunicationPath)</em> Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - <strong>naGPIB</strong> - GPIB interface</td>
</tr>
<tr>
<td></td>
<td>1 - <strong>naUSB</strong> - USB interface</td>
</tr>
<tr>
<td></td>
<td>2 - <strong>naLAN</strong> - LAN interface</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td><strong>naGPIB</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><a href="#">See example</a></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_Path(tagNACommunicationPath pNewVal);</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT get_Path(tagNACommunicationPath *pVal);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><a href="#">IPowerMeterInterface</a></td>
</tr>
</tbody>
</table>

---

Last Modified:

2-Aug-2007    MX New topic
About Cal Methods

PathCalMethod Property

**Description**

**Note:** This command replaces ThruCalMethod.

(Read-Write) Specifies the calibration method for each port pair.

**Note:** Before using this command, first do the following:

- Set the connector types: **ConnectorType**
- Set cal kit: **CalKitType**
- Set or query the thru path pairs: **ThruPortList**

After sending or querying this command, send the Thru method: **PathThruMethod**

See an example of a 4-port guided calibration using COM.

**VB Syntax**

```
guidedCal.PathCalMethod (port1, port2) = "caltype1[,caltype2]"
```

**Variable**  

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>guidedCal</code></td>
<td>GuidedCalibration (object)</td>
</tr>
<tr>
<td><code>port1</code></td>
<td>First port of the pair to be calibrated.</td>
</tr>
<tr>
<td><code>port2</code></td>
<td>Second port of the pair to be calibrated.</td>
</tr>
</tbody>
</table>

"caltype1[,caltype2]" (String) Cal types for 1st and 2nd ports of the pair, enclosed in a single pair of quotes. NOT case-sensitive.

*caltype1* Cal type for the pair if caltype2 is not specified. Otherwise, Cal type for port 1. Choose from:

- "TRL"
- "SOLT"
- "QSOLTN"
- "EnhRespN"

For the last two arguments, replace *N* with the port to be used as the source port, which MUST be one of the port pair.

*[caltype2]* Optional argument. Use only when performing an adapter removal cal on the pair. This argument specifies the Cal Type on the second port; *caltype1* then specifies the Cal Type of the first port.

Choose from the same arguments as *caltype1*.

**Return Type**  

**String** - Returns comma-separated cal types.
### Default
Not Applicable

### Example
```
guidedCal.PathCalMethod(2,3) = "TRL"  'Write trl for port pair

guidedCal.PathCalMethod(1,4) = "TRL,SOLT"  'Write adapter removal cal, consisting of trl on port 1 and solt on port 4
```

```
calmethod = guided.PathCalMethod(1,4)  'Read previous example, returns: "TRL,SOLT"
```

### C++ Syntax
```
HRESULT get_PathCalMethod(long firstport, long secondport, BSTR *calMethod);
HRESULT put_PathCalMethod(long firstport, long secondport, BSTR calMethod);
```

### Interface
IGuidedCalibration3

---

Last modified:

April 9, 2007  MX New topic
## PathConfiguration Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves or reads a path configuration file. This command is valid only on a PNA-X model.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.PathConfiguration = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>chan</code> (object)</td>
<td>A Channel object</td>
</tr>
<tr>
<td><code>value</code> (String)</td>
<td>Full path name of the configuration file to read or save.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.PathConfiguration = 'MyConfigFile' 'Write config = chan.PathConfiguration 'Read</code></td>
</tr>
</tbody>
</table>
| **C++ Syntax** | `HRESULT get_PathConfiguration(IPathConfiguration** ppOut );`  
`HRESULT put_PathConfiguration(BSTR pathConfigName );` |
| **Interface** | IChannel9 |

Last modified:  
10/18/06   MX New topic
PathThruMethod Property

**Description**

Note: This command replaces ThruCalMethod.

(Read-Write) Specifies the calibration THRU method for each port pair.

Before using this command, first do the following:

- Set the connector types: ConnectorType
- Set cal kit: CalKitType
- Set or query the thru path pairs: ThruPortList
- Set or query the Cal Type: PathCalMethod

See an example of a 4-port guided calibration using COM.

**VB Syntax**

```
guidedCal.PathThruMethod (port1, port2) = "ThruType1[,ThruType2]"
```

**Variable (Type) - Description**

- **guidedCal** GuidedCalibration (object)
  - **port1** First port of the pair to be calibrated.
  - **port2** Second port of the pair to be calibrated.

- **"ThruType1[,ThruType2]"** (String) Thru methods for 1st and 2nd ports of the pair, enclosed in a single pair of quotes. NOT case-sensitive.
  - **thruType1** Calibration thru method for the pair if thruType2 is not specified. Otherwise, thru method for port 1.
  - Choose from:
    - "Defined Thru" A thru type for which there is a stored definition in the Cal Kit.
    - "Zero Thru" Zero length thru, also known as flush-thru.
    - "Undefined Thru" A thru type for which there is NOT a stored definition in the Cal Kit. Also known as Unknown Thru. Valid ONLY for SOLT cal type.
    - "Undefined Thru using a Defined Thru" Using an ECal module, measure the internal thru using the "Undefined Thru" method.

- **ThruType2** (String) Optional argument. Use only when performing an adapter removal cal on the pair as determined by PathCalMethod. The only valid arguments for ThruType1&2 is "Defined Thru, Defined Thru".

**Return Type** String - Returns comma-separated ThruTypes.
<table>
<thead>
<tr>
<th>Default</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example</strong></td>
<td></td>
</tr>
<tr>
<td><code>guidedCal.PathThruMethod(2,3) = &quot;Zero Thru&quot; 'Write for port pair</code></td>
<td></td>
</tr>
<tr>
<td><code>guidedCal.PathThruMethod(1,4) = &quot;Defined Thru, Defined Thru&quot; 'Write for adapter removal cal.</code></td>
<td></td>
</tr>
<tr>
<td><code>calmethod = guided.PathThruMethod(1,4) 'Read previous example, return: &quot;Defined Thru, Defined Thru&quot;</code></td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td></td>
</tr>
<tr>
<td>HRESULT get_PathThruMethod(long firstport, long secondport, BSTR *thruMethod);</td>
<td></td>
</tr>
<tr>
<td>HRESULT put_PathThruMethod(long firstport, long secondport, BSTR thruMethod);</td>
<td></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGuidedCalibration3</td>
</tr>
</tbody>
</table>

Last modified:

April 9, 2007    MX New topic
### PeakExcursion Property

**Description**
Sets and reads the peak excursion value for the specified marker. The Excursion value determines what is considered a "peak".

**VB Syntax**
```
mark.PeakExcursion = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mark</td>
<td>A Marker (object)</td>
</tr>
<tr>
<td>value</td>
<td>(single) - Peak Excursion. Choose any number between -500 and 500</td>
</tr>
</tbody>
</table>

**Return Type**
Single

**Default**
3

**Examples**
```
mark.PeakExcursion = 1 'Write
PkExcur = mark.PeakExcursion 'Read
```

**C++ Syntax**
```
HRESULT get_PeakExcursion(float *pVal)
HRESULT put_PeakExcursion(float newVal)
```

**Interface**
IMarker
## PeakThreshold Property

**Description**
Sets peak threshold for the specified marker. If a peak (using the criteria set with \texttt{PeakExcursion}) is below this reference value, it will not be considered when searching for peaks.

**VB Syntax**
\[
mark.PeerThreshold = value
\]

**Variable**
\begin{itemize}
\item \texttt{mark} - A Marker (\texttt{object})
\end{itemize}

\begin{itemize}
\item \texttt{value} - Peak Threshold. Choose any number between:
\item \texttt{-500 and 500}
\end{itemize}

**Return Type**
Single

**Default**
-100db

**Examples**
\begin{itemize}
\item \texttt{mark.PeakThreshold = 1} \texttt{'}Write\texttt{'}
\item \texttt{PkThresh = mark.PeakThreshold \texttt{'}Read\texttt{'}
\end{itemize}

**C++ Syntax**
\begin{itemize}
\item HRESULT get_PeakThreshold(float *pVal)
\item HRESULT put_PeakThreshold(float newVal)
\end{itemize}

**Interface**
\texttt{IMarker}
### PeakToPeak Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Peak to Peak value of the measurement. To retrieve all 3 statistics value at the same time, use <code>meas.GetTraceStatistics</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>pp = meas.PeakToPeak</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>pp</code></td>
<td>(single) - Variable to store peak-to-peak value</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Single</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>pp = meas.PeakToPeak</code>  <code>Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_PeakToPeak(float* pp)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
### Period Property

**Description**  
Sets the pulse-period (1/PRF) for ALL PNA-X internal pulse generators.  
The resolution of the period is 16.667nS.

**VB Syntax**  
`pulse.Period = value`

**Variable**  
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pulse</code></td>
<td>A PulseGenerator (object)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>(Double) Pulse period in seconds. Choose a value from about 33ns to about 70 seconds.</td>
</tr>
</tbody>
</table>

**Return Type**  
Double

**Default**  
1e-3 sec

**Examples**  
`pulse.Period = 1ms 'Write`

`value = pulse.Period 'Read`

**C++ Syntax**  
```c++
HRESULT get_Period(double* period);
HRESULT put_Period(double period);
```

**Interface**  
IPulseGenerator

---

**Last Modified:**  
1-Jan-2007  
MX New topic
### PhaseOffset Property

**Description**
Sets the Phase Offset for the active channel.

**VB Syntax**
```vbnet
meas.PhaseOffset = value
```

**Variable (Type) - Description**
- `meas` A Measurement (object)
- `value` (double) - PhaseOffset in degrees. Choose any number between: -360 and +360

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
meas.PhaseOffset = 25 'Write
poffset = meas.PhaseOffset 'Read
```

**C++ Syntax**
```cpp
HRESULT get_PhaseOffset(double *pVal)
HRESULT put_PhaseOffset(double newVal)
```

**Interface**
IMeasurement
**Port1 Property  Superseded**

<table>
<thead>
<tr>
<th>Description</th>
<th>This command is replaced by PortDelay property. Sets a Port Extension value for Port 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>portExt.Port1 = value</td>
</tr>
<tr>
<td>Variable</td>
<td><strong>portExt</strong> - A Port Extension <em>(object)</em></td>
</tr>
<tr>
<td></td>
<td><strong>value</strong> - Port Extension value in seconds. Choose any number between -10 and 10</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>Examples</td>
<td>portExt.Port1 = 10e-6  'Write</td>
</tr>
<tr>
<td></td>
<td>prt1 = portExt.Port1  'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Port1(double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Port1(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IPortExtension</td>
</tr>
</tbody>
</table>
Port2 Property  Superseded

Description  This command is replaced by PortDelay property.

Sets a Port Extension value for Port 2

VB Syntax  

`portExt.Port2 = value`

Variable  

(Type) - Description

`portExt`  A Port Extension (object)

`value`  (double) - Port Extension value in seconds. Choose any number between -10 and 10

Return Type  Double

Default  0

Examples  

`portExt.Port2 = 10e-6`  'Write

`prt2 = portExt.Port2`  'Read

C++ Syntax  

HRESULT get_Port2(double *pVal)

HRESULT put_Port2(double newVal)

Interface  IPortExtension
### Port3 Property  Superseded

**Description**  
This command is replaced by PortDelay property. Sets a Port Extension value for Port 3.

**VB Syntax**  
`portExt.Port3 = value`

**Variable**  

<table>
<thead>
<tr>
<th><strong>Variable</strong></th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>portExt</code></td>
<td>A Port Extension</td>
<td><em>object</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double)</td>
<td>Port Extension value in seconds. Choose any number between -10 and 10</td>
</tr>
</tbody>
</table>

**Return Type**  
Double

**Default**  
0

**Examples**  

```vbnet
portExt.Port3 = 10e-6 'Write
```

```vbnet
prt3 = portExt.Port3 'Read
```

**C++ Syntax**  

```cpp
HRESULT get_Port3(double *pVal)
HRESULT put_Port3(double newVal)
```

**Interface**  
IPortExtension

---

1251
### Port2PdeembedCktModel Property

**Description**
Select whether or not to load a 2-port De-embedding circuit model for the specified port number. Circuit model is applied when both "USER" is selected and the filename is specified. To set the filename, use `strPort2Pdeembed_S2PFile Property`.

*Note:* This command affects ALL measurements on the channel.

**VB Syntax**
```vbnet
fixture.Port2PdeembedCktModel(port) = value
```

**Variable**
- **Type** - Description
  - `fixture` - A `Fixturing` (object)
  - `port` - (Integer) Port number to receive circuit model.
  - `value` - (Enum as NAFixturing2PdeembedCkt)
    - 0 `naFix2PD_USER` load a 2-port De-embedding circuit model
    - 1 `naFix2PD_NONE` no model

**Return Type**
Long Integer

**Default**
`naFix2PD_NONE`

**Examples**
```vbnet
fixture.Port2PdeembedCktModel(2) = naFix2PD_USER 'Write

value = fixture.Port2PdeembedCktModel(1) 'Read
```

**C++ Syntax**
```csharp
HRESULT get_Port2PdeembedCktModel(short port tagNAFixturing2PdeembedCkt *pVal)
HRESULT put_Port2PdeembedCktModel(short port tagNAFixturing2PdeembedCkt newVal)
```

**Interface**
`IFixturing`
## Port2PdeembedState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns de-embedding ON or OFF for all ports on the channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>fixture.Port2PdeembedState = value</code></td>
</tr>
<tr>
<td><strong>Variable (Type)</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td><code>fixture</code></td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Boolean)</td>
</tr>
<tr>
<td><code>False</code></td>
<td>Turns De-embedding OFF</td>
</tr>
<tr>
<td><code>True</code></td>
<td>Turns De-embedding ON</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>fixture.Port2PdeembedState = False</code> 'Write value = fixture.Port2PdeembedState 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT <code>get_Port2PdeembedState(VARIANT_BOOL *pVal)</code></td>
</tr>
<tr>
<td></td>
<td>HRESULT <code>put_Port2PdeembedState(VARIANT_BOOL newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IFixturing</td>
</tr>
</tbody>
</table>
### PortArbzImag Property

**Description**
Sets and returns the Imaginary portion of the impedance value for the specified single-ended port. Use **PortArbzReal** to set the real value. Or use **PortArbzZ0** to set both values together.

**VB Syntax**

```vbnet
fixture.PortArbzImag(portNum) = value
```

**Variable (Type) - Description**

- **fixture**
  - A **Fixturing** (object)
- **portNum**
  - **(Integer)** Single-ended port number to receive impedance value.
- **value**
  - **(Double)** Real Impedance value. Choose a value between -1E18 and 1E18

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
fixture.PortArbzImag(2) = 75 'Write
value = fixture.PortArbzImag(1) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PortArbzImag( short portNum, double *pVal)
HRESULT put_PortArbzImag( short portNum, double newVal)
```

**Interface**

IFixturing3
## PortArbzReal Property

### Description
Sets and returns the Real portion of the impedance value for the specified single-ended port. Use `PortArbzImag` to set the imaginary value. Or use `PortArbzZ0` to set both values together.

### VB Syntax
```
f如ure.PortArbzReal(portNum) = value
```

### Variable (Type) - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixture</td>
<td>Fixturing (object)</td>
<td></td>
</tr>
<tr>
<td>portNum</td>
<td>Integer</td>
<td>Single-ended port number to receive impedance value.</td>
</tr>
<tr>
<td>value</td>
<td>Double</td>
<td>Real Impedance value. Choose a value between 0 to 1E7</td>
</tr>
</tbody>
</table>

### Return Type
Double

### Default
50

### Examples
```
f如ure.PortArbzReal(2) = 75 'Write
value = f如ure.PortArbzReal(1) 'Read
```

### C++ Syntax
```
HRESULT get_PortArbzReal( short portNum, double *pVal)
HRESULT put_PortArbzReal( short portNum, double newVal)
```

### Interface
IFixturing3
PortArbzState Property

**Description**
Turns Port Impedance ON or OFF for all ports on the channel.

**VB Syntax**
```
fixture.PortArbzState = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixture</td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Boolean)</td>
</tr>
<tr>
<td>False</td>
<td>Turns Port Impedance OFF</td>
</tr>
<tr>
<td>True</td>
<td>Turns Port Impedance ON</td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**
False

**Examples**
```
fixture.PortArbzState = False 'Write
value = fixture.PortArbzState 'Read
```

**C++ Syntax**
```
HRESULT get_PortArbzState(VARIANT_BOOL *pVal)
HRESULT put_PortArbzState(VARIANT_BOOL newVal)
```

**Interface**
IFixturing
PortArbzZ0 Property

Description
Sets and returns the Real portion of the impedance value for the specified single-ended port. The imaginary portion is automatically set to 0.0.
To set both values separately, use PortArbzReal and PortArbzImag.

VB Syntax
`fixture.PortArbzZ0(portNum) = value`

Variable (Type) - Description
`fixture` A Fixturing (object)
`portNum` (Integer) Single-ended port number to receive impedance value.
`value` (Double) Impedance value. Choose a value between 0 to 1E7

Return Type
Double

Default
50

Examples
`fixture.PortArbzZ0(2) = 75` 'Write
`value = fixture.PortArbzZ0(1)` 'Read

C++ Syntax
`HRESULT get_PortArbzZ0( short portNum, double *pVal)`
`HRESULT put_PortArbzZ0( short portNum, double newVal)`

Interface
IFxturing3
## PortCatalog Property

**Description**
Returns a comma-separated list of the Output port selections that are available for a given logical input port.

Read the number of input ports for the test set using [NumberOfPorts Property](#).

**VB Syntax**
```vbnet
value = tset.PortCatalog(logPort)
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(String) Variable to store the returned information.</td>
</tr>
<tr>
<td>tset</td>
<td>A TestsetControl object.</td>
</tr>
<tr>
<td>logPort</td>
<td>(Long) Logical Input port number for which to return valid output ports.</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Not Applicable

**Examples**
```vbnet
value = testset1.PortCatalog 2
```

See External Testset Program

**C++ Syntax**
```cpp
HRESULT get_PortCatalog(long inputPort, BSTR *outPort);
```

**Interface**
ITestsetControl

---

**Last Modified:**
18-Jan-2007   
Fixed argument
**PortCLogic Property**

**Description**
Sets and reads the logic mode of Port C on the AUX IO connector and the Handler IO connector.

*Note*: Port C lines are connected to both the Handler IO and Aux IO in the PNA. Therefore, this command will affect both of these connectors in the same way.

**VB Syntax**

```
AuxIO.PortCLogic = value
```

**Variable**

**Type** - Description

- `AuxIO` *(object)* - A Hardware Aux I/O object
- `value` *(Enum as NaRearPanelIOLogic)* - Choose from:
  - `0 - naPositiveLogic` - The associated data line goes **HIGH** when writing a 1 to a PortC bit.
  - `1 - naNegativeLogic` - The associated data line goes **LOW** when writing a 1 to a PortC bit.

When Port C is in Output/Write mode, a change in logic causes the output lines to change state immediately. For example, Low levels change to High levels.

When Port C is in Input/Read mode, a change in logic will not cause the lines to change, but data read from Port C will reflect the change in logic.

**Return Type**

*Enum*

**Default**

`1 - naNegativeLogic`

**Examples**

```
auxIO.PortCLogic = value 'Write
value = auxIo.PortCLogic 'Read
```

**C++ Syntax**

```
HRESULT put_PortCLogic ( tagNARearPanelIOLogic Mode );
HRESULT get_PortCLogic ( tagNARearPanelIOLogic* Mode );
```

**Interface**

IHWAuxIO
Read/Write

PortCMode Property

**Description**
Sets and reads whether Port C is setup for writing or reading data on the AUX IO connector and the Handler IO connector.

**Note**: Port C lines are connected to both the Handler IO and Aux IO in the PNA. Therefore, this command will affect both of these connectors in the same way.

**VB Syntax**
```
AuxIO.PortCMode = value
```

**Variable**
- **(Type)** - Description
  - **AuxIO** *(object)* - A Hardware Aux I/O object
  - **value** *(enum as NaPortMode)* - Choose from:
    - 0 - naInput - set the port for reading
    - 1 - naOutput - set the port for writing

**Return Type**
Enum as NaPortMode

**Default**
1 - naInput

**Examples**
```
auxIo.get_PortCMode = naInput  'Write
value = auxIo.get_PortCMode  'Read
```

**C++ Syntax**
```
HRESULT get_PortCMode( tagNAPortMode* pMode );
HRESULT put_PortCMode( tagNAPortMode pMode );
```

**Interface**
IHWAuxIO
### PortDelay Property

**Description**
Sets and returns the Port Extensions Delay value for the specified port number.

*Note:* This command affects ALL measurements on the channel.
This command replaces Port 1 Port 2 Port 3 Properties.

**VB Syntax**

```vbnet
fixture.PortDelay(port) = value
```

**Variable**

*Type* - Description

- **fixture**
  A Fixturing (object)

- **port**
  (Integer) Port number to receive delay value.

- **value**
  (Double) Delay value in seconds. Choose a value between -1E18 and 1E18.

**Return Type**
Double

**Default**
0

**Examples**

```vbnet
fixture.PortDelay(2) = .002 'Write
value = fixture.PortDelay(1) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PortDelay(short port double *pVal)
HRESULT put_PortDelay(short port double newVal)
```

**Interface**
IFixturing
**PortExtState Property**

**Description**
Turns Port Extension ON or OFF for all ports on the channel.

**VB Syntax**
```
fixture.PortExtState = value
```

**Variable**
- **Type** - Description
  - `fixture` A `Fixturing` (object)
  - `value` (Boolean)
    - `False` - Turns Port Extensions OFF
    - `True` - Turns Port Extensions ON

**Return Type**
Boolean

**Default**
False

**Examples**
```
fixture.PortExtState = 0 'Write
value = fixture.PortExtState 'Read
```

**C++ Syntax**
```
HRESULT get_PortExtState(VARIANT_BOOL *pVal)
HRESULT put_PortExtState(VARIANT_BOOL newVal)
```

**Interface**
IFixturing
## PortExtUse1 Property

### Description
Sets and returns the Use1 ON/OFF state for the use of the PortLoss1 and PortFreq1 values for the specified port number.

**Note:** This command affects ALL measurements on the channel.

### VB Syntax
```
fixture.PortExtUse1(port) = value
```

### Variable (Type) - Description

- **fixture**  
  A Fixturing (object)

- **port**  
  (Integer) Port number to receive Use1 ON / OFF state.

- **value**  
  (Boolean)  
  False - Turns Use1 OFF  
  True - Turns Use1 ON

### Return Type
Boolean

### Default
False

### Examples
```
fixture.PortExtUse1(2) = False 'Write

value = fixture.PortExtUse1(1) 'Read
```

### C++ Syntax
```
HRESULT get_PortExtUse1(short port VARIANT_BOOL *pVal)
HRESULT put_PortExtUse1(short port VARIANT_BOOL newVal)
```

### Interface
IFixturing
PortExtUse2 Property

**Description**
Sets and returns the Use2 ON/OFF state for the use of the PortLoss2 and PortFreq2 values for the specified port number.

*Note:* This command affects ALL measurements on the channel.

**VB Syntax**
```
fixture.PortExtUse2(port) = value
```

**Variable**

- **fixture** (Type) - Description
  - A Fixturing (object)
- **port** (Integer) - Port number to receive Use2 ON / OFF state.
- **value** (Boolean)
  - False - Turns Use1 OFF
  - True - Turns Use1 ON

**Return Type**
Boolean

**Default**
False

**Examples**
```
fixture.PortExtUse2(2) = False  'Write
value = fixture.PortExtUse2(1)  'Read
```

**C++ Syntax**
```
HRESULT get_PortExtUse2(short port VARIANT_BOOL *pVal)
HRESULT put_PortExtUse2(short port VARIANT_BOOL newVal)
```

**Interface**
IFixturing
PortFreq1 Property

**Description**
Sets and returns Frequency1 value for the specified port number.

*Note:* This command affects ALL measurements on the channel.

**VB Syntax**
```
fixture.PortFreq1(port) = value
```

**Variable**
- *(fixture)* - Description
- *PortFreq1* (integer) Port number to receive extrapolated loss.
- *value* (Double) Frequency1 value. Choose a frequency within the frequency span of the PNA.

**Return Type**
Double

**Default**
1 GHz

**Examples**
```
fixture.PortFreq1(2) = naFix2PD_USER 'Write

value = fixture.PortFreq1(1) 'Read
```

**C++ Syntax**
```
HRESULT get_PortFreq1(short port double *pVal)
HRESULT put_PortFreq1(short port double newVal)
```

**Interface**
IFixturing
## PortFreq2 Property

### Description
Sets and returns Frequency2 value for the specified port number.  

**Note:** This command affects ALL measurements on the channel.

### VB Syntax
```vbnet
fixture.PortFreq2(port) = value
```

### Variable *(Type)* - Description

- **fixture** (A Fixturing *(object)*)
- **port** *(Integer)* Port number to receive extrapolated loss.
- **value** *(Double)* Frequency2 value. Choose a frequency within the frequency span of the PNA.

### Return Type
Double

### Default
1 GHz

### Examples
```vbnet
fixture.PortFreq2(2) = 10E9 'Write
d = fixture.PortFreq2(1) 'Read
```

### C++ Syntax
```cpp
HRESULT get_PortFreq2(short port double *pVal)
HRESULT put_PortFreq2(short port double newVal)
```

### Interface
IFixturing
## PortLabel Property

### Description
Sets and returns the label on the calibration kit Port for the calibration wizard.

### VB Syntax
```
calKit.PortLabel (portNum) = value
```

### Variable (Type) - Description

- **calKit**: A CalKit (object)
- **portNum**: (long integer) - number of the port to be labeled. Choose either 1 or 2
- **value**: (string) - Label that is visible in the calibration wizard.

### Return Type
String

### Default
Depends on the Cal Kit.

### Examples
```
calKit.PortLabel = "MyCalKit" 'Write

kitLabel = calKit.PortLabel 'Read
```

### C++ Syntax
```
HRESULT get_PortLabel(long port, BSTR *pVal)
HRESULT put_PortLabel(long port, BSTR newVal)
```

### Interface
ICalKit
### PortLogic Property

**Description**
Sets and returns the logic mode of data ports A-H on the Handler I/O connector. Port C of the Handler IO is connected internally to the Port C of the Aux IO connector. Therefore, it will have the same logic mode.

**VB Syntax**
```vbnet
handler.PortLogic = value
```

**Variable**
- **(Type) - Description**
  - `handler` *(object)* - A Handler I/O object
  - `value` *(enum as NaRearPanelIOLogic)* - Choose from:
    - **0 - naPositiveLogic** - When a value of one is written, the associated line goes High
    - **1 - naNegativeLogic** - When a value of one is written, the associated line goes Low
  - For ports that are in output (write) mode, a change in logic causes the output lines to change state immediately. For example, Low levels change immediately to High levels.
  - For ports that are in input (read) mode (C,D,E only), a change in logic will be reflected when data is read from that port. For example, if a line read 0, the next read after a logic change will read 1.

**Return Type**
Long Integer

**Default**
1 - naNegativeLogic

**Examples**
```vbnet
handler.PortLogic = value 'Write
value = handler.PortLogic 'Read
```

**C++ Syntax**
```c++
HRESULT put_PortLogic( tagNARearPanelIOLogic Mode );
HRESULT get_PortLogic( tagNARearPanelIOLogic* Mode );
```

**Interface**
IHWMaterialHandlerIO
## PortLoss1 Property

**Description**
Sets and returns the Loss1 value for the specified port number.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**

```vbnet
fixture.PortLoss1(port) = value
```

**Variable**

- **fixture** (*Type* - Description)
  A `Fixturing` *object*

- **port** (*Integer*) Port number to receive Loss value

- **value** (*Double*) Loss1 value in dB. Choose a value between -90 and 90.

**Return Type**

Double

**Default**

0

**Examples**

```vbnet
fixture.PortLoss1(2) = .002 'Write
value = fixture.PortLoss1(1) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PortLoss1(short port double *pVal)
HRESULT put_PortLoss1(short port double newVal)
```

**Interface**

`IFixturing`
## PortLoss2 Property

**Description**  
Sets and returns the Loss2 value for the specified port number. 

**Note**: This command affects ALL measurements on the channel.

### VB Syntax

```vbnet
fixture.PortLoss2(port) = value
```

### Variable

- **Type** - Description
  - `fixture`  
    - A `Fixturing` object
  - `port`  
    - A `Integer` Port number to receive Loss value
  - `value`  
    - A `Double` Loss2 value in dB. Choose a value between -90 and 90.

### Return Type

Double

### Default

0

### Examples

```vbnet
fixture.PortLoss2(2) = .002 'Write
```

```vbnet
value = fixture.PortLoss2(1) 'Read
```

### C++ Syntax

```cpp
HRESULT get_PortLoss2(short port double *pVal)
HRESULT put_PortLoss2(short port double newVal)
```

### Interface

`IFixturing`
### PortLossDC Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and returns the Loss value at DC for the specified port number. Learn about Loss compensation values.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note:</td>
<td>This command affects ALL measurements on the channel.</td>
</tr>
</tbody>
</table>

#### VB Syntax

```
fixture.PortLossDC(port) = value
```

#### Variable

- **fixture**  
  - Type: A Fixturing (object)
- **port**  
  - Type: Integer  
  - Description: Port number to receive Loss value at DC.
- **value**  
  - Type: Double  
  - Description: Loss value in ohms. Choose a value between -90 and 90

#### Return Type

- Type: Double

#### Default

- Value: 0

#### Examples

- Write: 
  ```
  fixture.PortLossDC(2) = .002 'Write
  ```
- Read: 
  ```
  value = fixture.PortLossDC(1) 'Read
  ```

#### C++ Syntax

- `HRESULT get_PortLossDC(short port double *pVal)`
- `HRESULT put_PortLossDC(short port double newVal)`

#### Interface

- IFixturing
### PortMatching_C Property

#### Description
Sets and returns the Capacitance value for the specified port number.

**Note:** This command affects ALL measurements on the channel.

#### VB Syntax
```vbnet
fixture.PortMatching_C(port) = value
```

**Variable** *(Type) - Description*
- **fixture** *(Fixed) - Porting (object)*
- **port** *(Integer)* - Port number to receive capacitance value
- **value** *(Double)* - Capacitance value in farads. Choose a value between -1E18 to 1E18.

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
fixture.PortMatching_C(2) = .00002 'Write
value = fixture.PortMatching_C(1) 'Read
```

**C++ Syntax**
```c++
HRESULT get_PortMatching_C(short port double *pVal)
HRESULT put_PortMatching_C(short port double newVal)
```

**Interface**
IFixturing
### PortMatching_G Property

**Description**
Sets and returns the Conductance value for the specified port number.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**

```vbnet
fixture.PortMatching_G(port) = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixture</td>
<td>A Fixturing (object)</td>
</tr>
<tr>
<td>port</td>
<td>(Integer) Port number to receive conductance value.</td>
</tr>
<tr>
<td>value</td>
<td>(Double) Conductance value in siemens. Choose a value between -1E18 and 1E18.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
0

**Examples**

```vbnet
fixture.PortMatching_G = .002 'Write
value = fixture.PortMatching_G 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PortMatching_G(short port double *pVal)
HRESULT put_PortMatching_G(short port double newVal)
```

**Interface**
IFixturing
### PortMatching_L Property

**Description**  
Sets and returns the Inductance value for the specified port number.  

*Note:* This command affects ALL measurements on the channel.

**VB Syntax**  
`fixture.PortMatching_L(port) = value`

**Variable**  
* (Type) - Description  

- **fixture**  
  A Fixturing (object)

- **port**  
  (Integer) Port number to receive inductance value

- **value**  
  (Double) Inductance value in henries. Choose a value between -1E18 and 1E18.

**Return Type**  
Double

**Default**  
0

**Examples**  
`fixture.PortMatching_L = .002 'Write`

`value = fixture.PortMatching_L 'Read`

**C++ Syntax**  
HRESULT get_PortMatching_L(short port double *pVal)  
HRESULT put_PortMatching_L(short port double newVal)

**Interface**  
IFixturing
PortMatching_R Property

Description
Sets and returns the Resistance value for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax
```
fixture.PortMatching_R(port) = value
```

Variable (Type) - Description

```
fixture A Fixturing (object)

port (Integer) Port number to receive resistance value.

value (Double) Resistance value in ohms. Choose a value between -1E18 and 1E18.
```

Return Type
Double

Default
0

Examples
```
fixture.PortMatching_R = .1 'Write

value = fixture.PortMatching_R 'Read
```

C++ Syntax
```
HRESULT get_PortMatching_R(short port double *pVal)
HRESULT put_PortMatching_R(short port double newVal)
```

Interface IFixturing
PortMatchingCktModel Property

Description
Sets and returns the Port Matching circuit model for the specified port number.

Note: This command affects ALL measurements on the channel.

VB Syntax

```vbnet
fixture.PortMatchingCktModel(port) = value
```

Variable

- **(Type)** - Description
  - `fixture` A Fixturing (object)
  - `port` (Integer) Port number to receive circuit model.
  - `value` (Enum as NAFixturingPortMatchCkt) Circuit model. Choose from
    0 naFixPMC_SLPC Series L - Parallel C
    1 naFixPMC_PCSL Parallel C - Series L
    2 naFixPMC_PLSC Parallel L - Series C
    3 naFixPMC_SCPL Series C - Parallel L
    4 naFixPMC_PLPC Parallel L - Parallel C
    5 naFixPMC_USER Load S2P file - also set filename to load with strPortMatch_S2PFile Property
    6 naFixPMC_NONE No circuit model

Return Type
Long Integer

Default
naFixPMC_NONE

Examples

```vbnet
fixture.PortMatchingCktModel(2) = naFixPMC_PLSC 'Write
value = fixture.PortMatchingCktModel(1) 'Read
```

C++ Syntax

```c
HRESULT get_PortMatchingCktModel(short port tagNAFixturingPortMatchCkt *pVal)
HRESULT put_PortMatchingCktModel(short port  tagNAFixturingPortMatchCkt newVal)
```

Interface

IFixturing
## PortMatchingState Property

**Description**  
Sets and returns the Port Matching State on the channel.

**VB Syntax**  
`fixture.PortMatchingState = value`

**Variable**  
- **Type** - Description
  - `fixture`  
    - A Fixturing (object)
  - `value`  
    - (boolean)
    - `True` - Turns Port Matching ON
    - `False` - Turns Port Matching OFF

**Return Type**  
Boolean

**Default**  
False

**Examples**  
- `fixture.PortMatchingState = True 'Write`
- `value = fixture.PortMatchingState 'Read`

**C++ Syntax**  
- `HRESULT get_PortMatchingState(VARIANT_BOOL *pVal)`
- `HRESULT put_PortMatchingState(VARIANT_BOOL newVal)`

**Interface**  
IFixturing
About the Handler I/O Connector

PortMode Property

Description
Sets and returns whether Port C or Port D is used for writing or reading data on the Handler IO connector. The Handler IO Port C is connected internally to the Port C of the Aux IO connector. Therefore, the Aux IO connector will have the same input/output mode.

VB Syntax
handler.PortMode (port) = value

Variable (Type) - Description
handler (object) - A Handler I/O object

Port (enum as NAMatHandlerPort) Port to be changed. Choose from:
2 - naPortC
3 - naPortD

value (enum as NaPortMode) - Choose from:
0 - naInput - set the port for reading
1 - naOutput - set the port for writing

Return Type
Long Integer

Default
1 - naInput

Examples
handler.PortMode(naPortC) = naInput 'Write
value = handler.PortMode(naPortD) 'Read

C++ Syntax
HRESULT put_PortMode ( tagNAMatHandlerPort Port, tagNAPortMode Mode );
HRESULT get_PortMode ( tagNAMatHandlerPort Port, tagNAPortMode* Mode );

Interface
IHWMaterialHandlerIO
### PortsNeedingDeltaMatch Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the port numbers for which delta match correction is required. 0 (zero) is returned if the Cal does NOT require Delta Match correction for one of the following reasons:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- The Cal does NOT involve Unknown Thru or TRL. You specify this using ThruCalMethod Property.</td>
</tr>
<tr>
<td></td>
<td>- The Cal DOES involve Unknown Thru or TRL, but the delta match data can be calculated by the Unknown Thru or TRL Cal. Learn how this is possible. However, you can force the Cal to use the Delta Match data from a Cal Set.</td>
</tr>
</tbody>
</table>

#### VB Syntax

```vbnet
value = guided.PortsNeedingDeltaMatch
```

#### Variable

**value** (Variant) Variable to store the returned list of port numbers.

**guided** GuidedCalibration (object)

#### Return Type

Variant

#### Default

Not Applicable

#### Examples

```vbnet
Dim ports As Variant
ports = guided.PortsNeedingDeltaMatch
```

#### C++ Syntax

```cpp
HRESULT get_PortsNeedingDeltaMatch (VARIANT* portList);
```

#### Interface

IGuidedCalibration2
**PowerSlope Property**

**Description**
Sets or returns the Power Slope value. Power Slope function increases or decreases the output power over frequency. Units are db/GHz. For example: PowerSlope = 2 will increase the power 2db/1GHz.

**VB Syntax**

```
object.PowerSlope = value
```

**Variable**

- **Object**
  - **Channel (object)**
  - **CalSet (object)** - Read-only property

- **Value**
  - **(double)** - Power Slope. Choose any number between -2 and 2.
  - **No slope = 0**

**Return Type**

Double

**Default**

0

**Examples**

```
chan.PowerSlope = 2 'Write
pwrslp = chan.PowerSlope 'Read
```

**C++ Syntax**

- HRESULT get_PowerSlope(double *pVal)
- HRESULT put_PowerSlope(double newVal)

**Interface**

- IChannel
- |CalSet3
Description
Returns the power sensor channel (A or B) that is currently selected for use at a specific frequency.

If UsePowerSensorFrequencyLimits is set to False, this property will return the sensor channel last used for a source power calibration. This setting corresponds to the Use this sensor only checkbox in the Power Sensor Settings dialog.

When performing an SMC calibration, use SetPowerAcquisitionDevice Method to set the power sensor channel.

VB Syntax
\[
\text{sensor} = \text{pwrCal.PowerAcquisitionDevice(}dFreq)\]

Variable (Type) - Description

\text{sensor} (enum NA\text{PowerAcquisitionDevice}) The currently selected sensor channel for the specified frequency. Choose from:

- 0 – \text{naPowerSensor\_A}
- 1 – \text{naPowerSensor\_B}

\text{pwrCal} A SourcePowerCalibrator (object)

\text{dFreq} (double) Frequency (Hz) for the power reading of interest.

Return Type enum NA\text{PowerAcquisitionDevice}

Default Not Applicable

Examples
\[
\text{selectedSensor} = \text{pwrCal.PowerAcquisitionDevice(1.E9) 'Write}\]

C++ Syntax
HRESULT get_PowerAcquisitionDevice(double dFreq, tagNA\text{PowerAcquisitionDevice}* enumAcqDevice);

Interface ISourcePowerCalibrator2
### PowerLevel Property

**Description**  
Set and read the power level at which to perform the Source Power Cal portion of a GainCompression Calibration.

**VB Syntax**  
`gcaCal.PowerLevel = value`

**Variable**  
**(Type) - Description**

- `gcaCal`  
  A `GainCompressionCal` *(object)*

- `value`  
  *(Double)* - Power level in dB. Choose a value from +30 to (-30). [Learn about choosing a power level](#).

**Return Type**  
Double

**Default**  
0

**Examples**

- `gca.PowerLevel = -5` *(Write)*

- `pLevel = gca.PowerLevel` *(Read)*

**C++ Syntax**

- `HRESULT get_PowerLevel(double* pVal)`
- `HRESULT put_PowerLevel(double newVal)`

**Interface**  
`IGainCompressionCal`

---

**Last Modified:**

10-Mar-2008  
MX New topic
PowerMeterChannel Property

**Description**
Identifies which channel of the power meter the power sensor is connected to.

**VB Syntax**

```vbnet
chan = powerSensor.PowerMeterChannel
```

**Variable**

**(Type)** - Description

- **chan** *(enum NAPowerAcquisitionDevice)* – Power meter channel identifier for sensor. Choose from:
  - 0 – naPowerSensor_A
  - 1 – naPowerSensor_B

- **pwrSensor** *(object)* - A PowerSensor (object)

**Return Type**
NAPowerAcquisitionDevice

**Default**
Not Applicable

**Examples**

```vbnet
Set pwrCal = pna.SourcePowerCalibrator
meterChannel = pwrCal.PowerSensors(1).PowerMeterChannel
```

**C++ Syntax**

```cpp
HRESULT PowerMeterChannel(tagNAPowerAcquisitionDevice *pSensor);
```

**Interface**
IPowerSensor
PowerMeterGPIBAddress Property  Superseded

Description
This command is replaced with PowerMeterInterface Object. Specifies the GPIB address of the power meter that will be referenced by the SourcePowerCalibrator object.
When performing a source power cal, the PNA will search VISA interfaces that are configured in the Agilent IO Libraries on the PNA.

VB Syntax
powerCalibrator.PowerMeterGPIBAddress = value

Variable (Type) - Description

powerCalibrator (object) - A SourcePowerCalibrator (object)
value (long integer) – Power meter GPIB address. Choose any number between 0 and 30.

Return Type
Long integer

Default
13

Examples
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.PowerMeterGPIBAddress = 13 'Write
pwrMtrAddress = powerCalibrator.PowerMeterGPIBAddress 'Read

C++ Syntax
HRESULT put_PowerMeterGPIBAddress(long newVal);
HRESULT get_PowerMeterGPIBAddress(long *pVal);

Interface
ISourcePowerCalibrator

Last Modified:
9-Jul-2007  Superseded
### PowerOnDuringRetraceMode Property

**Description**
For single-band frequency or segment sweeps ONLY, specify whether to turn RF power ON or OFF during a retrace.

This setting remains until changed using this command, or until the hard drive is changed or reformatted.

**VB Syntax**

```vbnet
pref.PowerOnDuringRetraceMode = value
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pref</code></td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum) - Choose from:</td>
</tr>
<tr>
<td>    0 - <code>naRetracePowerMode_Auto</code></td>
<td>Power is left ON during retrace of single-band frequency or segment sweeps ONLY.</td>
</tr>
<tr>
<td>    1 - <code>naRetracePowerMode_OFF</code></td>
<td>Power is turned OFF during retrace of single-band frequency or segment sweeps ONLY.</td>
</tr>
</tbody>
</table>

**Return Type**

Enum

**Default**

0 - `naRetracePowerMode_Auto`

**Examples**

```vbnet
pref.PowerOnDuringRetraceMode = naPowerSweepRetraceMode_Start 'Write
psMode = pref.naPowerOnDuringRetraceMode_Start 'Read
```

**C++ Syntax**

```cpp
HRESULT get_PowerOnDuringRetraceMode (tagNARetracePowerMode* preference);
HRESULT put_PowerOnDuringRetraceMode (tagNARetracePowerMode val)
```

**Interface**

IPreferences4
**PowerSweepRetracePowerMode Property**

**Description**
At the end of a power sweep, while waiting to trigger the next sweep, maintain source power at either the start power level or at the stop power level.

**VB Syntax**
```vbnet
pref.PowerSweepRetracePowerMode = value
```

**Variable (Type) - Description**
- **pref** A Preferences (object)
- **value** (Enum) - Choose from:
  - 0 - `naPowerSweepRetraceMode_Start` - maintain source at start power level.
  - 1 - `naPowerSweepRetraceMode_Stop` - maintain source at stop power level.

**Return Type**
Enum

**Default**
0 - `naPowerSweepRetraceMode_Start`

**Examples**
```vbnet
pref.PowerSweepRetracePowerMode = naPowerSweepRetraceMode_Start 'Write
psMode = pref.naPowerSweepRetraceMode_Start 'Read
```

**C++ Syntax**
```cpp
HRESULT get_PowerSweepRetracePowerMode (tagNAPowerSweepRetraceMode* preference);
HRESULT put_PowerSweepRetracePowerMode (tagNAPowerSweepRetraceMode val)
```

**Interface**
IPreferences3

---

Last modified:

Oct. 25, 2006  New command
### PreciseTuningTolerance Property

**Description**
Sets and returns the tuning tolerance for precise tuning.

**VB Syntax**
```
embedLO.PreciseTuningTolerance = value
```

**Variable** *(Type)* - Description

- `embedLO` An **EmbeddedLO** *(object)*
- `value` *(Double)* Tuning tolerance in Hz. Choose a number between and

**Return Type** *(Double)*

**Default**
1 Hz

**Examples**
```
embedLO.PreciseTuningTolerance = .5 'write

value = embedLO.PreciseTuningTolerance 'read
```

**C++ Syntax**
```
HRESULT get_PreciseTuningTolerance(double* tolerance);
HRESULT put_PreciseTuningTolerance(double tolerance);
```

**Interface**
IEmbededLO

---

Last Modified:
13-Apr-2007   MX New topic
PreferInternalTriggerOnChannelSingle Property

Description
Set and read the preference for the chan.Single trigger behavior. This setting persists until changed.

These preferences are important when performing a Guided calibration, as the PNA uses the chan.Single trigger command to measure standards.

- set PreferInternalTriggerOnChannelSingle = False to use an External trigger sweep to measure a cal standard.
- set PreferInternalTriggerOnChannelSingle = True to use an External sweep for the measurement, but rely on the PNA to send Internal trigger signals for calibrating.

To set this preference for an Unguided Calibration, use PreferInternalTriggerOnUnguidedCal Property

The chan.Single trigger command NEVER respects the Trigger Source = Manual setting. It always switches to Internal for one trigger, then back to Manual, regardless of this preference command.

VB Syntax
pref.PreferInternalTriggerOnChannelSingle = bool

Variable (Type) - Description
pref A Preferences (object)
bool (Boolean) - Choose from:
0 - False - the Single trigger property does respect the Trigger Source = External setting. For example, if Trigger source = External, the single trigger method will wait for the External trigger signal and then allow only one sweep.
1 - True - the Single trigger command does NOT respect the Trigger Source = External setting. For example, when the Single method is sent, the PNA immediately switches to Internal sweep, responds to one trigger signal, then switches back to External.

Return Type
Boolean

Default
0 - False

Examples
pref.PreferInternalTriggerOnChannelSingle = False 'Write
prefer = pref.PreferInternalTriggerOnChannelSingle 'Read

C++ Syntax
HRESULT put_PreferInternalTriggerOnChannelSingle( VARIANT_BOOL bprefSingle)
HRESULT get_PreferInternalTriggerOnChannelSingle( VARIANT_BOOL *bprefSingle)

Interface IPreferences2
PreferInternalTriggerOnUnguidedCal Property

**Description**
Set and read the preference for the trigger behavior when performing an Unguided calibration.

**VB Syntax**

```
pref.PreferInternalTriggerOnUnguidedCal = bool
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pref</code></td>
<td>A Preferences (object)</td>
</tr>
<tr>
<td><code>bool</code></td>
<td>(Boolean) - Choose from:</td>
</tr>
</tbody>
</table>

0 - **False** - The trigger behavior during an Unguided calibration DOES respect the setting of the Trigger source command. For example, during an Unguided Cal, when Trigger source = External, the PNA will wait for the External trigger signal and then allow only one sweep.

1 - **True** - The trigger behavior during an Unguided calibration does NOT respect the Trigger Source = External setting. For example, during an Unguided Cal, when Trigger source = External, the PNA immediately switches to Internal sweep, measures the standard with one trigger signal, then switches back to External trigger.

**Note:** When Trigger Source = Manual during a calibration, the PNA ALWAYS switches to Internal for one trigger to measure a standard, then back to Manual, regardless of this preference command.

**Return Type**
Boolean

**Default**
0 - False

**Examples**

```vbnet
pref.PreferInternalTriggerOnUnguidedCal = False 'Write
prefer = pref.PreferInternalTriggerOnUnguidedCal 'Read
```

**C++ Syntax**

```cpp
HRESULT put_PreferInternalTriggerOnUnguidedCal( VARIANT_BOOL bprefUnguided)
HRESULT get_PreferInternalTriggerOnUnguidedCal( VARIANT_BOOL *bprefUnguided)
```

**Interface**
`IPreferences2`
RangeCount Property

**Description**  Returns the number of ranges that are available in the PNA. To see the range names, query the Name property of each range in the FOM collection.

**VB Syntax**  value = FOM.RangeCount

**Variable**  
- **object**  An FOM (collection object)
- **value**  (long) - Variable to store the returned number of ranges.

**Return Type**  Long Integer

**Default**  Not Applicable

**Examples**  NumRanges = fom.RangeCount

**C++ Syntax**  HRESULT get_RangeCount(long *count)

**Interface**  IFOM

Last Modified:

8-Mar-2007  Added link to Name property
rangeNumber Property

**Description**
Returns the index number of the range within the FOM collection.

**VB Syntax**
```
value = FOMRange.rangeNumber
```

**Variable (Type) - Description**
- **value** (Long) - Variable to store the returned range number.
- **object** An FOMRange (object)

**Return Type**
Long

**Default**
Not Applicable

**Examples**
```
num = fomRange.rangeNumber 'Read
```

**C++ Syntax**
```
HRESULT get_rangeNumber(long *pVal)
```

**Interface**
IFOMRange
### R1inputPath Property

**Description**
PNA models with option 081 have a switch in the test set that allows access to the port 1 reference receiver through the front panel Reference 1 connectors. This command throws that switch between the internal path to the receiver, or through the external connectors. You could use this feature to make converter measurements relative to a reference ("golden") mixer.

See other Frequency Offset properties.

**VB Syntax**
```
chan.R1InputPath = value
```

**Variable**
- **chan** A Channel (object)
- **value** (Enum as naInputPath) - Choose from:
  - naPathInternal - (0) - internal path to the reference receiver
  - naPathExternal - (1) - path through external connectors

**Return Type**
Enum

**Default**
naPathInternal - (0)

**Examples**
```
chan.R1InputPath = naPathInternal 'Write
Inpath = chan.R1InputPath 'Read
```

**C++ Syntax**
```
HRESULT get_R1InputPath (tag NAInputPath *pPath);
HRESULT put_R1InputPath (tag NAInputPath newPath);
```

**Interface**
IChannel2
**ReadingsPerPoint Property**

**Description**  
This command, along with ReadingsTolerance, allows for settling of the power sensor READINGS.  
Specifies the maximum number of power readings that are taken at each stimulus point to allow for power meter settling. Each reading is averaged with the previous readings at that stimulus point. When this average meets the ReadingsTolerance value or this number of readings has been made, the average is returned as the valid reading.  
The following two commands deal with the source power ADJUSTMENTS:  
IterationsTolerance Property  
MaximumIterationsPerPoint Property

**VB Syntax**  
```vbnet  
pwrCal.ReadingsPerPoint = value  
```

**Variable (Type) - Description**

- **pwrCal** (object) - A SourcePowerCalibrator (object)
- **value** (long integer) – Number of power readings. Choose any number between 3 and 100.

**Return Type**  
Long Integer

**Default**  
3

**Examples**  
```
Set powerCalibrator = pna.SourcePowerCalibrator  
powerCalibrator.ReadingsPerPoint = 3 'Write  
numReadings = powerCalibrator.ReadingsPerPoint 'Read
```

**C++ Syntax**  
```cpp  
HRESULT put_ReadingsPerPoint(long newVal);  
HRESULT get_ReadingsPerPoint(long *pVal);  
```

**Interface**  
ISourcePowerCalibrator

---

Last Modified:  
17-Apr-2007  Clarified verbage
**ReadingsTolerance Property**

**Description**
This command, along with **ReadingsPerPoint Property** allows for settling of the power sensor READINGS.

Each power reading is averaged with the previous readings at each stimulus point. When the average meets this tolerance value or the maximum **ReadingsPerPoint** has been made, the average is returned as the valid reading.

The following two commands deal with the source power ADJUSTMENTS:

**IterationsTolerance Property**
**MaximumIterationsPerPoint Property**

**VB Syntax**
```
pwrCal.ReadingsTolerance = value
```

**Variable (Type) - Description**

- **pwrCal** (object) - A **SourcePowerCalibrator** (object)
- **value** (Double) – Power meter settling tolerance value in dB. Choose any number between 0 and 5.

**Return Type**
Double

**Default**
.05 dB

**Examples**
```
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.ReadingsTolerance = .1 'Write
ReadTol = powerCalibrator.ReadingsTolerance 'Read
```

**C++ Syntax**
```
HRESULT get_ReadingsTolerance( double *pVal);
HRESULT put_ReadingsTolerance( double newVal);
```

**Interface**
ISourcePowerCalibrator3

---

**Last Modified:**
17-Apr-2007 Clarified verbage
ReadyForTriggerState Property

**Description**
Determines the control of Material Handler connector Pin 21.

**VB Syntax**
```
handler.ReadyForTriggerState = value
```

**Variable (Type) - Description**
- `handler` (object) - A Handler I/O object
- `value` (boolean)
  - False - Pin 21 is controlled by Output Port B7
  - True - Pin 21 is controlled by the Ready for Trigger signal

**Return Type**
Boolean

**Default**
False

**Examples**
```
handler.ReadyForTriggerState = False 'Write
bState = handler.ReadyForTriggerState 'Read
```

**C++ Syntax**
```
HRESULT put_ReadyForTriggerState (BOOL *pVal);
HRESULT get_ReadyForTriggerState (BOOL newVal);
```

**Interface**
IHWMaterialHandlerIO2
ReceiverAttenuator Property

**Description**
Sets or returns the value of the specified receiver attenuator control.

**VB Syntax**
```
object.ReceiverAttenuator(rec) = value
```

**Variable (Type) - Description**

- **object**
  Channel (object)
  
  or
  CalSet (object) - Read-only property

- **rec** (long integer) - Receiver with attenuator control to be changed. Choose from any of the available receivers in your PNA
  
  1 - Receiver A
  2 - Receiver B

  Receiver attenuation can not be set using logical receiver notation.

- **value** (double) - Attenuator value in dB. Choose any Long Integer between 0 and 35 in 5dB steps:
  
  If an invalid value is entered, the analyzer will select the next lower valid value. For example, if 19.9 is entered the analyzer will select 15 dB attenuation.

**Return Type**
Double

**Default**
0 db

**Examples**
```
chan.ReceiverAttenuator(1) = 5 'Write

attn = chan.ReceiverAttenuator(2) 'Read
```

**C++ Syntax**
```
HRESULT get_ReceiverAttenuator(long lport, double *pVal)
HRESULT put_ReceiverAttenuator(long lport, double newVal)
```

**Interface**
IChannel
|CalSet3
Read only

ReceiverCount Property

**Description**  Returns the number of receivers in the remote PNA. The returned number includes both test port receivers and reference receivers. See the number of reference receivers in your PNA.

**VB Syntax**  
\[
\text{value} = \text{cap}.\text{ReceiverCount}
\]

**Variable**  
- **(Type)** - Description
- **value** (Long) - Variable to store the returned number of receivers.
- **cap** A Capabilities (object)

**Return Type**  Long

**Default**  Not Applicable

**Examples**  
\[
\text{value} = \text{cap}.\text{ReceiverCount} \quad \text{'Read'}
\]

**C++ Syntax**  

```
HRESULT get_ReceiverCount(long * receiverCount);
```

**Interface**  ICapabilities
### ReceiverStepAttenuatorStepSize Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns a value indicating the step size of the attenuator.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.ReceiverStepAttenuatorStepSize (n)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Double)</em> - Variable to store the returned value of the attenuator step size.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A <a href="#">Capabilities</a> <em>(object)</em></td>
</tr>
<tr>
<td><code>n</code></td>
<td><em>(Long)</em> - port number to query for the value of the attenuator step size.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.ReceiverStepAttenuatorStepSize 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_ReceiverStepAttenuatorStepSize(long portNumber, double * stepSize );</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities</td>
</tr>
</tbody>
</table>
Read-only

ReceivePort Property

**Description**
Returns the receiver (response) port number of measurement. To understand how this property is useful, see [IMeasurement2 Interface](#).

**Note:** Returning a receiver port is only supported for S-Parameter measurements. If the measurement is not an S-Parameter, then E_NA_BAD_PARAMETER is returned.

**VB Syntax**

```
value = meas.ReceivePort
```

**Variable (Type) - Description**

- `value` *(Long)* - Variable to store the returned value
- `meas` A [Measurement](#) *(object)*

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**

```
rp = meas.ReceivePort
```

**C++ Syntax**

```
HRESULT ReceivePort(Long* rcvPort);
```

**Interface**
IMeasurement2
### ReduceIFBandwidth Property

**Description**  
Sets or returns the state of the [Reduced IF Bandwidth at Low Frequencies](#) setting.

**VB Syntax**  
\[ chan.ReduceIFBandwidth = state \]

**Variable**  
(Type) - Description

- `chan`  
  A Channel (object)

- `state`  
  (boolean)  
  *False* - Turns Reduce IFBW OFF  
  *True* - Turns Reduce IFBW ON

**Return Type**  
Boolean

**Default**  
True

**Examples**  
\[ chan.ReduceIFBandwidth = False 'Write \]

\[ reduce = chan.ReduceIFBandwidth 'Read \]

**C++ Syntax**  
\[ HRESULT get_ReduceIFBandwidth(BOOL *pVal) \]
\[ HRESULT put_ReduceIFBandwidth(BOOL newVal) \]

**Interface**  
IChannel5

---

```
Last Modified:
```

16-Aug-2007  Corrected Interface number
ReferenceCalFactor Property

Description
Reference cal factor (%) associated with this power sensor. This property and the CalFactorSegments collection are used to perform source power calibration only if the power sensor does not contain cal factors in EPROM (for example, HP/Agilent 848x sensors).

VB Syntax
```vbnet
powerSensor.ReferenceCalFactor = value
```

Variable (Type) - Description

- **pwrSensor** (object) - A PowerSensor (object)
- **value** (double) – Cal factor in units of percent. This can be any value between 1 and 150.

Return Type
Double

Default
100

Examples
```vbnet
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.PowerSensors(1).ReferenceCalFactor = 99 'Write
ReffFact = powerCalibrator.PowerSensors(1).ReferenceCalFactor 'Read
```

C++ Syntax
```c++
HRESULT put_ReferenceCalFactor(double newVal);
HRESULT get_ReferenceCalFactor(double *pVal);
```

Interface
IPowerSensor
## ReferenceMarkerState Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turn ON or OFF the reference marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>meas.ReferenceMarkerState = state</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>app</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>state</td>
<td>(boolean) -</td>
</tr>
<tr>
<td>True</td>
<td>turns the reference marker ON</td>
</tr>
<tr>
<td>False</td>
<td>turns the reference marker OFF</td>
</tr>
<tr>
<td>Return Type</td>
<td>Boolean</td>
</tr>
<tr>
<td>Default</td>
<td>False</td>
</tr>
<tr>
<td>Examples</td>
<td><code>meas.ReferenceMarkerState = True</code></td>
</tr>
<tr>
<td></td>
<td><code>reference = meas.ReferenceMarkerState</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_ReferenceMarkerState(VARIANT_BOOL bState)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_ReferenceMarkerState(VARIANT_BOOL* bState)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
## ReferenceValue Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the value of the Y-axis Reference Level of the active trace.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>trce.ReferenceValue = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>trce</code></td>
<td>A Trace <em>(object)</em></td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(double)</em> - Reference Value. Units and range depend on the current data format.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>meas.ReferenceValue = 0</code> <em>(Write)</em></td>
</tr>
<tr>
<td></td>
<td><code>rlev = meas.ReferenceValue</code> <em>(Read)</em></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_ReferenceValue(double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_ReferenceValue(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>ITrace</td>
</tr>
</tbody>
</table>
Write/Read

ReferencePosition Property

Description
Sets or returns the Reference Position of the active trace.

VB Syntax
trce.ReferencePosition = value

Variable
(Type) - Description

trce  A Trace (object)

value  (double) - Reference position on the screen measured in horizontal graticules from the bottom of the screen. Choose from any number between: 0 and 10.

Return Type
Double

Default
0

Examples
meas.ReferencePosition = 5 'Middle of the screen -Write
rpos = meas.ReferencePosition -Read

C++ Syntax
HRESULT get_ReferencePosition(double *pVal)
HRESULT put_ReferencePosition(double newVal)

Interface
ITrace
RemoteCalStoragePreference Property

Description
Specifies the default manner in which calibrations performed using COM or SCPI are to be stored. Cal data is always stored to the channel’s Cal Register regardless of this setting. This setting survives instrument preset and reboot. It remains until changed by another invocation of this property.

VB Syntax

```vbnet
pref.RemoteCalStoragePreference = value
```

Variable (Type) - Description

- `cal` (A Preferences (object))
- `value` (Enum) - Choose from:
  - 0 - `naPreferCalRegister` - Cal is saved ONLY to the channel’s Cal Register.
  - 1 - `naPreferNewUserCalSet` - Cal is automatically saved to a new User Cal Set file when performing a calibration using COM. The Cal Set name is automatically generated. This corresponds to pre-6.0 behavior. Use the `Name` property to change the name after the cal is complete.
  - 2 - `naPreferReuseCurrentCalSet` - The cal is saved to the Cal Set is that is currently selected on the specific channel. This could be the channel’s Cal Register. If the channel does not yet have a selected Cal Set, the cal will be saved to a new User Cal Set with an automatically-generated name.

Return Type
Enum

Default
0 - `naPreferCalRegister`

Examples

```vbnet
pref.RemoteCalStoragePreference = naPreferNewUserCalSet 'Write

calStorageMode = pref.RemoteCalStoragePreference ' Read
```

C++ Syntax

```csharp
HRESULT get_RemoteCalStoragePreference(enum NARemoteCalStoragePreference* preference);
HRESULT put_RemoteCalStoragePreference(enum NARemoteCalStoragePreference val);
```

Interface
IPreferences7

Last Modified:
16-Apr-2007   MX New topic
**Write/Read**

**ReverseLinearPowerLevel Property**

**Description**  
Set and read the reverse power level to the DUT. This is applied to the DUT output port when making reverse measurements like S22.

**VB Syntax**  
`gca.ReverseLinearPowerLevel = value`

**Variable**  
*(Type) - Description*

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gca</td>
<td>GainCompression</td>
<td>A</td>
</tr>
<tr>
<td>value</td>
<td>double</td>
<td>Reverse power level in dBm. Choose a value from +30 to (-30).</td>
</tr>
</tbody>
</table>

**Return Type**  
Double

**Default**  
-5

**Examples**  
`gca.ReverseLinearPowerLevel = -10 'Write`

`LinPwr = gca.ReverseLinearPowerLevel 'Read`

**C++ Syntax**  
HRESULT get_ReverseLinearPowerLevel(double* pVal)

HRESULT put_ReverseLinearPowerLevel(double newVal)

**Interface**  
IGainCompression

---

Last Modified:

21-Nov-2007   MX New topic
## SafeSweepCoarsePowerAdjustment Property

### Description
Set and read the Safe Sweep COARSE power adjustment.

### VB Syntax
```vbnet
gca.SafeSweepCoarsePowerAdjustment = value
```

### Variable *(Type) - Description*

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gca</code></td>
<td>A <code>GainCompression</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) Coarse power adjustment setting in dBm. Choose a value from +30 to (-30).</td>
</tr>
</tbody>
</table>

### Return Type
Double

### Default
3.0

### Examples
```vbnet
gca.SafeSweepCoarsePowerAdjustment = 2.0 'Write
SSCourse = gca.SafeSweepCoarsePowerAdjustment 'Read
```

### C++ Syntax
```cpp
HRESULT get_SafeSweepCoarsePowerAdjustment(double* value)
HRESULT put_SafeSweepCoarsePowerAdjustment(double value)
```

### Interface
`IGainCompression`

---

Last Modified:

1-Dec-2007    MX New topic
SafeSweepEnable Property

Description
Set and read the (ON | OFF) state of Safe Sweep mode.

VB Syntax
```
gca.SafeSweepEnable = value
```

Variable
- **Type** - Description
  - `gca` - A `GainCompression` object
  - `value` - (Boolean) Safe Sweep state. Choose from:
    - **False** - Disable Safe Sweep
    - **True** - Enable Safe Sweep

Return Type
Boolean

Default
False

Examples
```
gca.SafeSweepEnable = True 'Write
SSEnable = gca.SafeSweepEnable 'Read
```

C++ Syntax
```
HRESULT get_SafeSweepEnable(VARIANT_BOOL* value)
HRESULT put_SafeSweepEnable(VARIANT_BOOL value)
```

Interface
IGainCompression

Last Modified:
1-Dec-2007   MX New topic
SafeSweepFinePowerAdjustment Property

Description
Set and read the Safe Sweep FINE power adjustment.

VB Syntax
	`gca.SafeSweepFinePowerAdjustment = value`

Variable (Type) - Description

gca A `GainCompression` (object)

value (Double) Fine power adjustment setting in dBm. Choose a value from +30 to (-30).

Return Type
Double

Default
1.0

Examples
	`gca.SafeSweepFinePowerAdjustment = 0.1 'Write`

	`SSfine = gca.SafeSweepFinePowerAdjustment 'Read`

C++ Syntax

HRESULT get_SafeSweepFinePowerAdjustment(double* value)
HRESULT put_SafeSweepFinePowerAdjustment(double value)

Interface
`IGainCompression`

Last Modified:

1-Dec-2007 MX New topic
### SafeSweepFineThreshold Property

**Description**  
Set and read the compression level at which Safe Sweep changes from the COARSE power adjustment to the FINE power adjustment.

**VB Syntax**  
`gca.SafeSweepFineThreshold = value`

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gca</td>
<td>GainCompression (object)</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>Double</td>
<td>Threshold setting in dBm. Choose a value from +30 to (-30).</td>
</tr>
</tbody>
</table>

**Return Type**  
Double

**Default**  
0.75 dBm

**Examples**

```vbnet
gca.SafeSweepFineThreshold = 0.1  'Write
SSThresh = gca.SafeSweepFineThreshold  'Read
```

**C++ Syntax**

```cpp
HRESULT get_SafeSweepFineThreshold(double* value)
HRESULT put_SafeSweepFineThreshold(double value)
```

**Interface**  
IGainCompression

---

**Last Modified:**

1-Dec-2007   MX New topic
### SB_BalPortNegative Property

**Description**  
With a Single-ended - Balanced topology, returns the PNA port number that is connected to the Negative side of the DUT’s Balanced Port. Use [SetSBPorts Method](#) to set the port mapping for a Single-Ended - Balanced topology.

**VB Syntax**  
```vbnet
var = balTopology.SB_BalPortNegative
```

**Variable**  
**(Type) - Description**

- `var` (Long Integer) Variable to store the returned value.

**balTopology**  
A [BalancedTopology](#) (object)

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
```vbnet
variable = balTop.SB_BalPortNegative  'Read
```

**C++ Syntax**  
```c++
HRESULT get_SB_BalPortNegative(long *bVal)
```

**Interface**  
IBalancedTopology
### Description
With a Single-ended - Balanced topology, returns the PNA port number that is connected to the Positive side of the DUT's Balanced Port.

Use SetSBPorts Method to set the port mapping for a Single-Ended - Balanced topology.

#### VB Syntax

```vbnet
var = balTopology.SB_BalPortPositive
```

#### Variable

**(Type)** - Description

- `var` (Long Integer) Variable to store the returned value.

#### balTopology

A BalancedTopology (object)

#### Return Type

Long Integer

#### Default

Not Applicable

#### Examples

```vbnet
variable = balTop.SB_BalPortPositive  'Read
```

#### C++ Syntax

```cpp
HRESULT get_SB_BalPortPositive(long *bVal)
```

#### Interface

IBalancedTopology
SB_SEPort Property

**Description**
With a Single-ended - Balanced topology, returns the PNA port number that is connected to the DUT's Single-ended port.

Use [SetSBPorts Method](#) to set the port mapping for a Single-Ended - Balanced topology.

**VB Syntax**

```
var = balTopology.SB_SEPort
```

**Variable**

*(Type) - Description*

- `var` (Long Integer) Variable to store the returned value.

**balTopology**
A [BalancedTopology](#) (object)

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**

```
variable = balTopology.SB_SEPort 'Read
```

**C++ Syntax**

```
HRESULT get_SB_SEPort(long *bVal)
```

**Interface**
IBalancedTopology
**SBalMeasurement Property**

**Description**
Sets and returns the measurement for the Single-Ended - Balanced topology.

**VB Syntax**

```
balMeas.SBalMeasurement = value
```

**Variable**

- **balMeas**
  A `BalancedMeasurement` (object)

- **value**
  (String) - Single-ended - Balanced Measurement parameter. Not case-sensitive. Choose from:

<table>
<thead>
<tr>
<th></th>
<th>Ssd12</th>
<th>Ssc12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sss11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sds21</td>
<td>Sdd22</td>
<td>Sdc22</td>
</tr>
<tr>
<td>Scs21</td>
<td>Scd22</td>
<td>Scc22</td>
</tr>
<tr>
<td>Imb</td>
<td>CMRR1</td>
<td>CMRR2</td>
</tr>
<tr>
<td></td>
<td>(Sds21/Scs21)</td>
<td>(Ssd12/Ssc12)</td>
</tr>
</tbody>
</table>

**Return Type**

Sss11

**Default**

Not Applicable

**Examples**

```
balMeas.SBalMeasurement = "Ssd12" 'Write
variable = balMeas.SBalMeasurement 'Read
```

**C++ Syntax**

```
HRESULT get_SBalMeasurement(BSTR *pVal)
HRESULT put_SBalMeasurement(BSTR newVal)
```

**Interface**

IBalancedMeasurement
### Scope Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the scope of a trigger signal. This determines whether a trigger signal affects a single channel or all channels in the PNA.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong> Trigger Modes</td>
<td><strong>Point and EverySweep</strong> require that <strong>Trigger.Scope</strong> be set to <strong>naChannelTrigger</strong>.</td>
</tr>
</tbody>
</table>

#### VB Syntax

```vbnet
trigsetup.Scope = value
```

#### Variable *(Type) - Description*

- **trigsetup** *(A TriggerSetup (object))*
- **value** *(enum NATriggerType)* - Trigger type. Choose from:
  - 0 - **naGlobalTrigger** - a trigger signal is applied to all triggerable channels
  - 1 - **naChannelTrigger** - a trigger signal is applied to the current channel. The next trigger signal will be applied to the next channel; not necessarily the next channel in numeric sequence (1-2-3-4 and so forth).

#### Return Type

- Long Integer

#### Default

- naGlobalTrigger

#### Examples

- ```vbnet
  trigsetup.Scope = naGlobalTrigger 'Write
  ```
- ```vbnet
  trigtyp = trigsetup.Scope 'Read
  ```

#### C++ Syntax

```c++
HRESULT get_Scope(tagNATriggerType *pTrigger)
HRESULT put_Scope(tagNATriggerType trigger)
```

#### Interface

- ITriggerSetup

---

**Last Modified:**

- 6-Nov-2007  Updated for new sweep mode
## SearchFunction Property

**Description**
Emulates the Tracking function in the marker search dialog box. The value you choose for SearchFunction will determine the type of search that takes place when the Tracking property is set true.

The tracking function finds the selected search function every sweep. In effect, turning Tracking ON is the same as executing one of the "Search..." methods (such as SearchMin, SearchMax) for every sweep.

**VB Syntax**
```vbnet
mark.SearchFunction = value
```

**Variable (Type) - Description**
- **mark**
  A Marker (object)
- **value**
  (enum NAMarkerFunction) - search function. Choose from:
  - 0 - naMarkerFunction_None
  - 1 - naMarkerFunction_Min
  - 2 - naMarkerFunction_Max
  - 3 - naMarkerFunction_Target
  - 4 - naMarkerFunction_NextPeak
  - 5 - naMarkerFunction_PeakRight
  - 6 - naMarkerFunction_PeakLeft

**Return Type**
Long Integer

**Default**
0 - naMarkerFunction_None

**Examples**
```vbnet
mark.SearchFunction = naMarkerFunction_Target 'When this marker is set to track, it will track the Target value.
```
```
searchfunction = mark.SearchFunction 'Read
```

**C++ Syntax**
```cpp
HRESULT get_SearchFunction(tagNAMarkerFunction *pVal)
HRESULT put_SearchFunction(tagNAMarkerFunction newVal)
```

**Interface**
IMarker
### SecurityLevel Property

**Description**
Controls the display of frequency information on the PNA screen and printouts.

**VB Syntax**
```
app.SecurityLevel = value
```

**Variable**
**Type** - Description

- **app** An [Application](#) (object)

- **value** (enum NASecurityLevel) - Choose from:
  - **0 - naNoSecurity** ALL frequency information is displayed.
  - **1 - naLowSecurity** NO frequency information is displayed. Frequency information can be redisplayed using the Security Setting dialog box or this command.
  - **2 - naHighSecurity** LOW setting plus [GPIB console](#) is disabled. Frequency information can be redisplayed ONLY by performing a Preset, recalling an instrument state with None or Low security settings, or using this command.
  - **3 - naExtraSecurity** HIGH setting plus [ASCII data saving](#) is disabled. Same method to redisplay frequency information as HIGH setting.

**Return Type**
Long Integer

**Default**
0 - None

**Examples**
```
app.SecurityLevel = naLowSecurity 'Write

level = app.SecurityLevel 'Read
```

**C++ Syntax**
```
HRESULT get_NASecurityLevel(tagNASecurityLevel *level);
HRESULT put_NASecurityLevel(tagNASecurityLevel level);
```

**Interface**
IApplication4

---

**Last Modified:** 17-Jul-2007  Add Extra level
**SegmentNumber Property**

**Description**
Returns the number of the current segment, PowerSensorCalFactorSegment or PowerLossSegment object.

**VB Syntax**
`seg.SegmentNumber`

**Variable (Type) - Description**
`seg (object)` - A Segment, PowerSensorCalFactorSegment or PowerLossSegment. Get a handle to the object by referring to the item in the appropriate collection (Segments, CalFactorSegments or PowerLossSegments).

**Return Type**
Long Integer

**Default**
Not Applicable

**Examples**
```
segNum = seg.SegmentNumber 'returns the segment number - Read
```

**C++ Syntax**
`HRESULT get_SegmentNumber(long *pVal)`

**Interface**
- ISegment
- IPowerSensorCalFactorSegment
- IPowerLossSegment
SelectPort Property

**Description**
Sets and returns a port mapping for a single port. If this command creates a conflict with an existing port, the PNA will resolve the conflict.

**Note:** This command is currently not supported for the Z5623AK44.

**VB Syntax**
`tset.SelectPort(chNum, portNum) = portValue`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tset</td>
<td>A <em>TestsetControl</em> object.</td>
</tr>
<tr>
<td><em>chNum</em></td>
<td>(Long) Channel number of the measurement.</td>
</tr>
<tr>
<td><em>portNum</em></td>
<td>(Long) Physical port number to map.</td>
</tr>
<tr>
<td><em>portValue</em></td>
<td>(Long) <em>Logical</em> port value to assign</td>
</tr>
</tbody>
</table>

**Return Type**
Long

**Default**
Not Applicable

**Examples**
See External Testset Program

**C++ Syntax**

```cpp
HRESULT get_SelectPort(long channelNum, long PortNum long *outPort);
HRESULT put_SelectPort(long channelNum, long PortNum long outPort);
```

**Interface**
`ITestsetControl`
ShowStatistics Property

Description: Displays and hides the measurement (Trace) statistics (peak-to-peak, mean, standard deviation) on the screen. To display measurement statistics for a narrower band of the X-axis, use StatisticsRange.

The analyzer will display either measurement statistics or Filter Bandwidth statistics; not both.

VB Syntax:  
```vbnet
meas.ShowStatistics = value
```

Variable: (Type) - Description

- **meas** A Measurement (object)
- **value** (boolean) - Boolean value:
  - **True** - Show statistics
  - **False** - Hide statistics

Return Type: Boolean

Default: False

Examples:
```vbnet
meas.ShowStatistics = True 'Write
showstats = meas.ShowStatistics 'Read
```

C++ Syntax: HRESULT put_ShowStatistics(VARIANT_BOOL bState)

Interface: IMeasurement
ShowProperties Property

**Description**
Turns ON and OFF the display of the test set control status bar. This status bar indicates the test set that is being controlled and the current port mappings. This setting is turned ON and OFF automatically when the test set is enabled or disabled.

**VB Syntax**
\[ tset.ShowProperties = value \]

**Variable (Type) - Description**
- \( tset \) A `TestsetControl` object.
  OR
  An `E5091Testset` object.
- \( value \) (Boolean)
  - `True` - Turns display of testset properties ON.
  - `False` - Turns display of testset properties OFF.

**Return Type**
Boolean

**Default**
`False` (True when test set control is enabled.)

**Examples**
See E5091A Example Program
See External Testset Program

**C++ Syntax**
```cpp
HRESULT get_ShowProperties(VARIANT_BOOL *state);
HRESULT put_ShowProperties(VARIANT_BOOL state);
```

**Interface**
IE5091Testsets
ITestsetControl
SICL Property

Description
Allows you to control the PNA via SICL (standard instrument control library). In this mode, the analyzer can receive SCPI commands from the LAN interface or from a program residing on the PNA itself. This command performs the same function as the SICL / GPIB dialog box - SICL Enabled checkbox. See Configuring the analyzer for SICL/VISA.

With this method you can augment a test program written using SICL that resides on the PNA so that it will run unattended. An automation script can be written to start the PNA, enable SICL (using the SICL property), and then start the SICL based program.

VB Syntax

app.SICL value

Variable (Type) - Description

app An Application (object)

value (Boolean) Choose from:

True - enable SICL
False - disable SICL

Return Type
Boolean

Default False

Examples
Dim Pna as AgilentPNA835x.Application
Dim siclState as Boolean
Set Pna = CreateObject("AgilentPNA835x.Application")
Pna.SICL = true  'write

siclState = Pna.SICL  'Read

C++ Syntax

HRESULT get_SICL(VARIANT_BOOL *pVal)
HRESULT put_SICL(VARIANT_BOOL newVal)

Interface IApplication5
SICLAddress Property

**Description**  Sets and returns the PNA SICL address. This is the address used for SICL over LAN.

**VB Syntax**  
```
app.SICLAddress = value
```

**Variable (Type) - Description**

- **app**  An Application (object)
- **value**  (Integer) SICL Address of the PNA. Choose a value between 0 and 30.

**Return Type**  Short Integer

**Default**  16

**Examples**

```
address=app.SICLAddress 'Read
app.SICLAddress=16 'Write
```

**C++ Syntax**  
```
HRESULT get_SICLAddress(short busIndex, short* address);
HRESULT put_SICLAddress(short busIndex,short address);
```

**Interface**  IApplication8
Simultaneous2PortAcquisition Property

**Description**
Specifies whether a 2-port calibration will be done with a single set of standards (one port at a time) or with two sets of standards (simultaneously).

The `AcquireCalStandard2` command uses the same standard index for each calibration class. To specify the calibration standard gender for each port, you must first ensure that the order of calibration class accurately reflects the configuration of your DUT. For example, for a DUT with a male connector on port 1 and a female connector on port 2, order the devices within the S11 classes (A, B, and C) such that the MALE standards are first in the list. Then order the S22 classes specifying the FEMALE standards as the first in the list.

**VB Syntax**

```vbnet
cal.Simultaneous2PortAcquisition = state
```

**Variable**

- **(Type)** - Description
  - **cal** A Calibrator *(object)*
  - **state** *(boolean)* - Choose from:
    - **True** - measures 2 ports simultaneously
    - **False** - measures 1 port at a time

**Return Type**
Boolean

**Default**
True

**Examples**

```vbnet
cal.Simultaneous2PortAcquisition = True
```

**C++ Syntax**

```c++
HRESULT put_Simultaneous2PortAcquisition( VARIANT_BOOL bTwoSetsOfStandards)
HRESULT Simultaneous2PortAcquisition( VARIANT_BOOL *bTwoSetsOfStandards)
```

**Interface**
ICalibrator

---

Last modified:

- **9/20/06**  Changed default to True
- **9/12/06**  Modified for cross-browser
**SmartSweepMaximumIterations Property**

**Description**
Set and read the maximum permitted number of iterations which SMART Sweep may utilize to find the desired compression level, to within the specified tolerance.

**VB Syntax**
```
gca.SmartSweepMaximumIterations = value
```

**Variable (Type) - Description**
- **gca** A *GainCompression* (object)
- **value** (integer) - Maximum number of iterations. Choose a value between 1 and 50.

**Return Type**
Integer

**Default**
20

**Examples**
```
gca.SmartSweepMaximumIterations = 10 'Write

iters = gca.SmartSweepMaximumIterations 'Read
```

**C++ Syntax**
```
HRESULT get_SmartSweepMaximumIterations(int* pVal)
HRESULT put_SmartSweepMaximumIterations(int newVal)
```

**Interface**
IGainCompression

**Last Modified:**
11-Sep-2007   MX New topic
SmartSweepSettlingTime Property

Description
Set and read the amount of time SMART Sweep will dwell at the first point where the input power changes by the Backoff or X level.

Learn more.

VB Syntax
`gca.SmartSweepSettlingTime = value`

Variable (Type) - Description

`gca` A `GainCompression` (object)

`value` (double) - Settling time in seconds. Choose any positive value.

Default 0

Examples
`gca.SmartSweepSettlingTime = .01` 'Write

`sTime = gca.SmartSweepSettlingTime` 'Read

C++ Syntax

```cpp
HRESULT get_SmartSweepSettlingTime(double* pVal)
HRESULT put_SmartSweepSettlingTime(double newVal)
```

Interface
`IGainCompression`

Last Modified:
21-Nov-2007 MX New topic
### SmartSweepShowIterations Property

**Description**
Set and read whether to show intermediate results for each iteration in SMART sweep.

**VB Syntax**
`gca.SmartSweepShowIterations = value`

**Variable**
- **Type**: Description
- `gca` A GainCompression (object)
- `value` (Boolean) Choose from:
  - **True**: Compression traces are updated after each iteration.
  - **False**: Compression traces are updated after ALL iterations are complete.

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
gca.SmartSweepShowIterations = True 'Write
SShow = gca.SmartSweepShowIterations 'Read
```

**C++ Syntax**
```cpp
HRESULT get_SmartSweepShowIterations(VARIANT_BOOL *pVal)
HRESULT put_SmartSweepShowIterations(VARIANT_BOOL newVal)
```

**Interface**
IGainCompression

---

Last Modified:
21-Nov-2007   MX New topic
## SmartSweepTolerance Property

**Description**
Set and read the acceptable range SMART Sweep will allow for the measured compression level.

**VB Syntax**
```vbnet
gca.SmartSweepTolerance = value
```

**Variable**
- **Type**: 
  - `gca` (GainCompression) - Object
  - `value` (double) - Tolerance level in dB. Choose a value between .01 and 10

**Return Type**
Double

**Default**
.05

**Examples**
- `gca.SmartSweepTolerance = .01 'Write`
- `tol = gca.SmartSweepTolerance 'Read`

**C++ Syntax**
- HRESULT get_SmartSweepTolerance(double* pVal)
- HRESULT put_SmartSweepTolerance(double newVal)

**Interface**
IGainCompression

---

**Last Modified:**
11-Sep-2007  MX New topic
About Smoothing

SmoothingAperture Property

Description
Specifies or returns the amount of smoothing as a ratio of the number of data points in the measurement trace.
There is no COM command for specifying smoothing by number of aperture points.

VB Syntax
meas.SmoothingAperture = value

Variable
(meas) A Measurement (object)

value (double) - Smoothing Aperture. A ratio of (aperture points / trace points). Choose any number between .01 and .25.

Return Type
Double

Default
.25

Examples
meas.SmoothingAperture = .10 'Write
saperture = meas.SmoothingAperture 'Read

C++ Syntax
HRESULT get_SmoothingAperture(double *pVal)
HRESULT put_SmoothingAperture(double newVal)

Interface
IMeasurement

Last modified:
Oct. 25, 2006 Fixed formula for smoothing
Smooth Property

Description
 Turns ON and OFF data smoothing.

VB Syntax
 meas.Smoothing = state

Variable
 (Type) - Description
 meas A Measurement (object)
 state (boolean)
 True - Turns smoothing ON
 False - Turns smoothing OFF

Return Type
 Boolean

Default
 False

Examples
 meas.Smoothing = False 'Write
 smooth = meas.Smoothing 'Read

C++ Syntax
 HRESULT get_Smoothing(VARIANT_BOOL *pVal)
 HRESULT put_Smoothing(VARIANT_BOOL newVal)

Interface
 IMeasurement
Write/Read

SnPFormat Property

Description
Specifies the format of .SnP files.
Use either app.Save (saves data to file) or meas.GetSnPData (reads data into variant array).

VB Syntax
```
pref.SnPFormat = value
```

Variable (Type) - Description

```vbnet
pref
```
A Preferences (object)

```vbnet
value
```
(string) - Format of the .S1P, .S2P, .S3P, .S4P data. Choose from:
- "MA" - Linear Magnitude / degrees
- "DB" - Log Mag / degrees
- "RI" - Real / Imaginary
- "Auto" - Format in which the trace is already displayed. If other than Log Mag, Linear Magnitude, or Real/Imag, then the format will be in Real/Imag.

Return Type
String

Default
"Auto"

Examples
```
pref.SnPFormat = "MA" 'Write

format = pref.SnPFormat 'Read
```

C++ Syntax
```
HRESULT get_SnPFormat(BSTR *Format)
HRESULT put_SnPFormat(BSTR Format)
```

Interface
IPreferences
### SoundOnFail Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns ON or OFF the audio indicator for limit failures.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>limitst.SoundOnFail = state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>limitst</code></td>
<td>A LimitTest (object)</td>
</tr>
<tr>
<td><code>state</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td><code>False</code></td>
<td>Turns the sound OFF</td>
</tr>
<tr>
<td><code>True</code></td>
<td>Turns the sound ON</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>Limittest.SoundOnFail = False</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>sound = Limittest.SoundOnFail</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_SoundOnFail(VARIANT_BOOL *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_SoundOnFail(VARIANT_BOOL newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ILimitTest</td>
</tr>
</tbody>
</table>
## SourceCount Property

**Description**
Returns the number of sources in the remote PNA.

**VB Syntax**
```
value = cap.SourceCount
```

**Variable**
- **value** *(Long)* - Variable to store the returned number of sources.
- **cap** A [Capabilities](#) *(object)*

**Return Type**
Long

**Default**
Not Applicable

**Examples**
```
value = cap.SourceCount 'Read
```

**C++ Syntax**
```
HRESULT get_SourceCount(long * sourceCount);
```

**Interface**
ICapabilities
### SourcePort Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the source port of measurement. To understand how this property is useful, see IMeasurement2 Interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = meas.SourcePort</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) - Variable to store the returned value</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long Integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>sp = meas.SourcePort</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SourcePort( [out, retval] Long* srcPort);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
## SourcePortCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of ports that can output a signal. To learn more, see <a href="#">Remotely Specifying a Source Port</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = cap.SourcePortCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) - Variable to store the returned integer value of the number of source ports.</td>
</tr>
<tr>
<td><code>cap</code></td>
<td>A <a href="#">Capabilities</a> (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>value = cap.SourcePortCount 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_SourcePortCount(long * count );</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICapabilities4</td>
</tr>
</tbody>
</table>

---

**Last Modified:**

14-Jan-2007  MX New topic
SourcePortMode Property

**Description**
Sets the state of the PNA source for the specified port.

**VB Syntax**
`chan.SourcePortMode (sourcePort) = value`

**Variable** *(Type) - Description*
- `chan` *(object)* - A `Channel` object
- `sourcePort` *(long integer)* - The source port for which to make this setting.
  Use `GetPortNumber` to return the port number of a source that only has a string name, such as an `External Source`.
- `value` *(enum)* - State of the source. Choose from:
  - 0 - `naSourcePortAuto` Port power is turned on when required for a measurement.
  - 1 - `naSourcePortOn` Port power is always ON, regardless of the measurement.
  - 2 - `naSourcePortOff` Port power is always OFF, regardless of the measurement.
  **Note:** ON and OFF are valid only on PNA models with two sources.

**Return Type**
Enum

**Default**
0 - `naSourcePortAuto`

**Examples**
```
chan.SourcePortMode(1) = naSourcePortOn  'Write
state = chan.SourcePortMode(4)  'Read
```

**C++ Syntax**
```
HRESULT get_SourcePortMode(long sourcePort, enum NASourcePortMode*);
HRESULT put_SourcePortMode(long sourcePort, enum NASourcePortMode);
```

**Interface**
`IChannel9`

---

Last modified:

April 30, 2007     Edited for src strings

10/18/06     MQQ New topic
SourcePortNames Property

**Description**
Returns the string names of ports that can output a signal.

The following is a list of string names for the PNA-X. Your PNA will NOT have all of these ports. Use GetPortNumber Method to return the correct port number for the specified port name.

- "Port 1"
- "Port 2"
- "Port 3"
- "Port 4"
- "Src2 Out1"
- "Src2 Out2"
- "Port 1 Src2"

This command also lists the External Sources that are currently configured and selected.

To learn more, see Remotely Specifying a Source Port.

**VB Syntax**

```
value = cap.SourcePortNames
```

**Variable** *(Type)* - Description

- **value** *(Variant array)* - Variable to store the returned integer value of the number of source ports.

- **cap** A Capabilities *(object)*

**Return Type**
Variant array of string names.

**Default**
Not Applicable

**Examples**

```
value = cap.SourcePortNames 'Read
```

**C++ Syntax**

```
HRESULT get_SourcePortNames(VARIANT *names);
```

**Interface**
ICapabilities4

---

Last Modified:

- 23-Jul-2007   Clarification
- 14-Jan-2007   MX New topic

1338
## Source Property

### Description
Sets or returns the source of triggering in the PNA.

### VB Syntax
```
trigSetup.Source = value
```

### Variable
- **(Type)** - Description
- **trigSetup** - A `TriggerSetup (object)`

- **value** - (enum `NATriggerSource`) - Choose from:
  - 0 - `naTriggerSourceInternal` - free run
  - 2 - `naTriggerSourceExternal` - a trigger signal is generated when a trigger signal is sensed on the external trigger pin of the Aux IO connector. Use `ExternalTriggerConnectionBehavior` to configure the characteristics of the external trigger signal.

This setting has implications on Calibration. [Learn more](#).

### Return Type
Long Integer

### Default
`naTriggerSourceInternal`

### Examples
```
trigSetup.Source = naTriggerSourceInternal 'Write
trigsource = trigSetup.Source 'Read
```

### C++ Syntax
```
HRESULT get_Source(tagNATriggerSource *pTrigger);
HRESULT put_Source(tagNATriggerSource trigger);
```

### Interface
- `ITriggerSetup`
SourcePowerCalPowerOffset Property

**Description**
Sets or returns a power level offset from the PNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT.

Cal power is the sum of the test port power setting and this offset value. Following the calibration, the PNA power readouts are adjusted to the cal power.

This property performs the same function as the power offset argument on SetCalInfoEx Method, except that this property can read the offset value.

**VB Syntax**
`chan.SourcePowerCalPowerOffset(sourcePort) = value`

**Variable (Type) - Description**
- `chan` (object) - A Channel object
- `sourcePort` (long integer) - The source port for which to set this power offset value.
- `value` (double) - Gain or loss value in dB. Choose a value between -200 and 200.

**Return Type** Double

**Default** 0 dB

**Examples**
```vba
chan.SourcePowerCalPowerOffset(1) = 10 'Write
offset = chan.SourcePowerCalPowerOffset(2) 'Read
```

**C++ Syntax**
```c++
HRESULT get_SourcePowerCalPowerOffset(long sourcePort, double *pVal);
HRESULT put_SourcePowerCalPowerOffset(long sourcePort, double newVal);
```

**Interface** IChannel4

Last Modified:
1-May-2007 Modified link to EX method.
SourcePowerCorrection Property

**Description**
Sets source power correction ON or OFF for a specific source port on this channel, or returns the current ON or OFF state of correction for that source port.

**VB Syntax**
`chan.SourcePowerCorrection (srcPort) = value`

**Variable**
- **Type** - Description
  - `chan` *(object)* – A Channel object
  - `srcPort` *(long integer)* – Source port for which to set or return the ON or OFF state of source power correction.
    Use `GetPortNumber` to return the port number of a source that only has a string name, such as an `External Source`.
  - `value` *(boolean)*
    - False – Turns source power correction OFF for the source port.
    - True – Turns source power correction ON for the source port.

**Return Type**
Boolean

**Default**
False - Source power correction will turn correction ON

**Examples**
```
chan.SourcePowerCorrection(1) = False 'Write
calOnPort2 = chan.SourcePowerCorrection(2) 'Read
```

**C++ Syntax**
```
HRESULT put_SourcePowerCorrection(VARIANT_BOOL bState);
HRESULT get_SourcePowerCorrection(VARIANT_BOOL *bState);
```

**Interface**
IChannel

---

Last Modified:

30-Apr-2007  Modified for src strings
SourcePowerOption Property

**Description**
Enables the source power to be set on individual sweep segments. This property must be set True before seg.TestPortPower = value is sent. Otherwise, the test port power command will be ignored.

**VB Syntax**
```vbnet
segs.SourcePowerOption = state
```

**Variable**
- **Type**: Boolean
- **Description**
  - `segs` (object): A Segments collection
  - `state` (boolean):
    - True: Enables variable TestPortPower to be set segment sweep
    - False: Disables variable TestPortPower to be set segment sweep

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
segs.SourcePowerOption = True 'Write
powerOption = SourcePowerOption 'Read
```

**C++ Syntax**
```cpp
HRESULT get_SourcePowerOption(VARIANT_BOOL *pVal)
HRESULT put_SourcePowerOption(VARIANT_BOOL newVal)
```

**Interface**
ISegments
### SourcePowerState Property

**Description**

Turns Source Power ON and OFF.

*See note about source power state with instrument state save and recall.*

**VB Syntax**

```vbnet
app.SourcePowerState = state
```

**Variable**

**(Type)** - Description

- **app** An Application (object)
- **state** (boolean)
  - *False* - Turns Source Power OFF
  - *True* - Turns Source Power ON

**Return Type**

Boolean

**Default**

True

**Examples**

```vbnet
app.SourcePowerState = True 'Write
pwr = app.SourcePowerState 'Read
```

**C++ Syntax**

```c++
HRESULT get_SourcePowerState(VARIANT_BOOL *pVal)
HRESULT put_SourcePowerState(VARIANT_BOOL newVal)
```

**Interface**

IApplication
### Span Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the Span time of either Gating or Time Domain transform windows</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>object.Span = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td>object</td>
<td><em>(object)</em> As Gating or <em>(object)</em> As Transform</td>
</tr>
<tr>
<td>value</td>
<td><em>(double)</em> - Span time in seconds. Choose any number between: (2\times[\frac{\text{number of points}-1}{\text{frequency span}}]) and 0</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>20ns</td>
</tr>
<tr>
<td>Examples</td>
<td><code>Trans.Span = 4.5e-9</code> 'sets the time span of a transform window'</td>
</tr>
<tr>
<td></td>
<td><code>Gate.Span = 4.5e-9</code> 'sets the Span time of a gating window'</td>
</tr>
<tr>
<td></td>
<td><code>span = Trans.Span</code> 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_Span(double *pVal)</code></td>
</tr>
<tr>
<td></td>
<td><code>HRESULT put_Span(double newVal)</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>ITransform</code></td>
</tr>
<tr>
<td></td>
<td><code>IGating</code></td>
</tr>
</tbody>
</table>
Span Property

**Description**
Returns the stimulus span of the measurement (stop-start data points). To understand how this property is useful, see [IMeasurement2 Interface](#).

**VB Syntax**

```vbnet
value = meas.Span
```

**Variable**

- **value** *(Double)* - Variable to store the returned value.
- **meas** A Measurement *(object)*

**Return Type**
Double

**Default**
Not Applicable

**Examples**

```vbnet
Print meas.Span 'prints the span of the measurement
```

**C++ Syntax**

```cpp
HRESULT get_Span(double * Val);
```

**Interface**
IMeasurement2
**SSB_BalPortNegative Property**

**Description**
With a Single-ended - Single-ended - Balanced topology, returns the PNA port number that is connected to the Negative side of the DUT’s Balanced Port.


**VB Syntax**
```
var = balTopology.SSB_BalPortNegative
```

**Variable**
- **(Type)** - Description
  - `var` (Long Integer) Variable to store the returned value.

**balTopology**
- A [BalancedTopology](#) object

**Return Type**
- Long Integer

**Default**
- Not Applicable

**Examples**
```
variable = balTopology.SSB_BalPortNegative  'Read
```

**C++ Syntax**
```
HRESULT get_SSB_BalPortNegative(long *bVal)
```

**Interface**
- IBalancedTopology
SSB_BalPortPositive Property

**Description**  
With a Single-ended - Single-ended - Balanced topology, returns the PNA port number that is connected to the Positive side of the DUT's Balanced Port. Use [SetSSBPorts Method](#) to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.

**VB Syntax**  
`var = balTopology.SSB_BalPortPositive`

**Variable**  
- **Type**: Long Integer  
- **Description**: Variable to store the returned value.

**balTopology**  
A [BalancedTopology](#) (object)

**Return Type**  
Long Integer

**Default**  
Not Applicable

**Examples**  
`variable = balTopology.SSB_BalPortPositive  ' Read`

**C++ Syntax**  
`HRESULT get_SSB_BalPortPositive(long *bVal)`

**Interface**  
IBalancedTopology
SSB_SEPort1 Property

<table>
<thead>
<tr>
<th>Description</th>
<th>With a Single-ended - Single-ended - Balanced topology, returns the PNA port number that is connected to the DUT's Logical Port 1. Use SetSSBPorts Method to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>var = balTopology.SSB_SEPort1</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>var</td>
<td>(Long Integer) Variable to store the returned value.</td>
</tr>
<tr>
<td>balTopology</td>
<td>A BalancedTopology (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>variable = balTopology.SSB_SEPort1 'Read'</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_SSB_SEPort1(long *bVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IBalancedTopology</td>
</tr>
</tbody>
</table>
SSB_SEPort2 Property

**Description**  With a Single-ended - Single-ended - Balanced topology, returns the PNA port number that is connected to the DUT's Logical Port 2.

Use **SetSSBPorts Method** to set the port mapping for a Single-Ended - Single-Ended - Balanced topology.

**VB Syntax**  
```
var = balTopology.SSB_SEPort2
```

**Variable**  (Type) - Description

- **var**  (Long Integer) Variable to store the returned value.

**balTopology**  A **BalancedTopology** (object)

**Return Type**  Long Integer

**Default**  Not Applicable

**Examples**  
```
variable = balTopology.SSB_SEPort2  'Read
```

**C++ Syntax**  
```
HRESULT get_SSB_SEPort2(long *bVal)
```

**Interface**  IBalancedTopology
**SSBMeasurement Property**

**Description**

**VB Syntax**
`balMeas.SSBMeasurement = value`

**Variable (Type) - Description**

- `balMeas` A `BalancedMeasurement` (object)
- `value` (String) - Single-ended - Single-ended - Balanced Measurement parameter. Not case sensitive. Choose from:

<table>
<thead>
<tr>
<th>Sss11</th>
<th>Sss12</th>
<th>Ssd13</th>
<th>Ssc13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sss21</td>
<td>Sss22</td>
<td>Ssd23</td>
<td>Ssc23</td>
</tr>
<tr>
<td>Sds31</td>
<td>Sds32</td>
<td>Sdd33</td>
<td>Sdc33</td>
</tr>
<tr>
<td>Scs31</td>
<td>Scs32</td>
<td>Scd33</td>
<td>Scc33</td>
</tr>
<tr>
<td>Imb1</td>
<td>Imb2</td>
<td>CMRR1</td>
<td>CMRR2</td>
</tr>
</tbody>
</table>

(Sds31/Scs31) (Sds32/Scs32)

**Return Type** String

**Default** Sss11

**Examples**
`balMeas.SSBMeasurement = "Sss11"` 'Write
`variable = balMeas.SSBMeasurement` 'Read

**C++ Syntax**

- HRESULT get_SSBMeasurement(BSTR *pVal)
- HRESULT put_SSBMeasurement(BSTR p newVal)

**Interface**
IBalancedMeasurement
### Stage1Coefficients Property

**Description**
Sets and returns the digital filter coefficients of stage1.

**VB Syntax**

```vbnet
spm4.Stage1Coefficients = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>spm4</code></td>
<td>A <code>SignalProcessingModuleFour</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Variant Array) Coefficients. An array of real values.</td>
</tr>
</tbody>
</table>

**Return Type**

Variant

**Default**

Stage dependent.

**Examples**

```vbnet
spm4.Stage1Coefficients = 0,0.1,0.7,0.7,0.1 'Write
mode = spm4.Stage1Coefficients 'Read
```

**C++ Syntax**

```c++
HRESULT get_Stage1Coefficients(VARIANT* pCoefs);
HRESULT put_Stage1Coefficients(VARIANT pCoefs);
```

**Interface**

`ISignalProcessingModuleFour`

---

Last Modified:

18-Jun-2007  MX New topic
Stage1Frequency Property

Description
Sets and returns the Numerically Controlled Oscillator (NCO) frequency of the Stage 1 filter. This command is only used when FilterMode Property is set to Manual.

VB Syntax

```
spm4.Stage1Frequency = value
```

Variable (Type) - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spm4</td>
<td>A SignalProcessingModuleFour (object)</td>
</tr>
<tr>
<td>value</td>
<td>(Double) Stage 1 Frequency. Min value= 0 Hz, Max value= 15 MHz. Or programmatically use MinimumIFFrequency Property and MaximumIFFrequency Property to determine the range of settable values.</td>
</tr>
</tbody>
</table>

Return Type
Double

Default
9 MHz

Examples
```
spm4.Stage1Frequency = 9E6 'Write
mode = spm4.Stage1Frequency 'Read
```

C++ Syntax

```
HRESULT getStage1Frequency(double *val);
HRESULT putStage1Frequency(double val);
```

Interface
ISignalProcessingModuleFour

Last Modified:
18-Jun-2007    MX New topic
Description

Returns the maximum value of any single coefficient.

VB Syntax

```
value = spm4.Stage1MaximumCoefficient
```

Variable  (Type) - Description

<table>
<thead>
<tr>
<th>value</th>
<th>(Long) Variable to store the returned Max coefficient.</th>
</tr>
</thead>
<tbody>
<tr>
<td>spm4</td>
<td>A <code>SignalProcessingModuleFour</code> (object)</td>
</tr>
</tbody>
</table>

Default  Not Applicable

Examples

```
mode = spm4.Stage1MaximumCoefficient
```

C++ Syntax

```
HRESULT get_Stage1MaximumCoefficient(long* val);
```

Interface  ISignalProcessingModuleFour

Last Modified:

1-Jan-2007  MX New topic
### Stage1MaximumCoefficientCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the maximum number of coefficients for Stage1.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>Value = spm4.Stage1MaximumCoefficientCount</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>Type</strong> - Description</td>
</tr>
<tr>
<td>Value</td>
<td>(Long) Variable to store the returned Max coefficient count.</td>
</tr>
<tr>
<td>spm4</td>
<td>A <a href="#">SignalProcessingModuleFour</a> object</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td></td>
</tr>
<tr>
<td>mode = spm4.Stage1MaximumCoefficientCount</td>
<td></td>
</tr>
</tbody>
</table>

**C++ Syntax**

```
HRESULT get_Stage1MaximumCoefficientCount(long* val);
```

**Interface**

ISignalProcessingModuleFour

---

Last Modified:

1-Jan-2007   MX New topic
# Stage1MaximumCoefficientSum Property

**Description**  
Returns the maximum sum of all Stage1 coefficients.

**VB Syntax**  
`value = spm4.Stage1MaximumCoefficientSum`

**Variable**  
**Type** - Description

- `value` (`__int64* val`)  
  Variable to store the returned Max sum of all coefficients.

- `spm4`  
  A `SignalProcessingModuleFour` (object)

**Default**  
Not Applicable

**Examples**  
`mode = spm4.Stage1MaximumCoefficientSum 'Read`

**C++ Syntax**  
`HRESULT get_Stage1MaximumCoefficientSum(__int64* val);`

**Interface**  
`ISignalProcessingModuleFour`

---

Last Modified:  
1-Jan-2007  
MX New topic
## Stage1MinimumCoefficientCount Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the minimum number of coefficients for Stage1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = spm4.Stage1MinimumCoefficientCount</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long) Variable to store the returned Min coefficient count.</td>
</tr>
<tr>
<td><code>spm4</code></td>
<td>A SignalProcessingModuleFour (object)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mode = spm4.Stage1MinimumCoefficientCount</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Stage1MinimumCoefficientCount(long* val);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISignalProcessingModuleFour</td>
</tr>
</tbody>
</table>

---

Last Modified:

1-Jan-2007 MX New topic
Stage2Coefficients Property

**Description**
Sets and returns Stage2Coefficients.

**VB Syntax**
```
spm4.Stage2Coefficients = value
```

**Variable**
- **Type**: Description
  - `spm4`: A `SignalProcessingModuleFour` (object)
  - `value`: (Variant) An array of real numbers. Filter coefficients

**Return Type**
Variant

**Default**
Not Applicable

**Examples**
```
spm4.Stage2Coefficients = "Write"
mode = spm4.Stage2Coefficients "Read"
```

**C++ Syntax**
```
HRESULT get_Stage2Coefficients(VARIANT* pCoefs);
HRESULT put_Stage2Coefficients(VARIANT pCoefs);
```

**Interface**
`ISignalProcessingModuleFour`

---

Last Modified:

1-Jan-2007   MX New topic
### Stage2MaximumCoefficient Property

**Description**  
Returns the maximum value of any single coefficient.

**VB Syntax**  
`value = spm4.Stage2MaximumCoefficient`

**Variable**  
**Type** - **Description**

- `value`  
  **Type**: Long  
  Variable to store the returned Max coefficient.

- `spm4`  
  A SignalProcessingModuleFour (object)

**Default**  
Not Applicable

**Examples**  
`mode = spm4.Stage2MaximumCoefficient`

**C++ Syntax**  
`HRESULT get_Stage2MaximumCoefficient(long* val);`

**Interface**  
ISignalProcessingModuleFour

---

**Last Modified:**

- 1-Jan-2007  
  MX New topic
Stage2MaximumCoefficientCount Property

**Description**
Returns the maximum number of coefficients for Stage2

**VB Syntax**
```
value = spm4.Stage2MaximumCoefficientCount
```

**Variable (Type) - Description**
- **value** (Long) Variable to store the returned Max coefficient count.
- **spm4** A SignalProcessingModuleFour (object)

**Default**
Not Applicable

**Examples**
```
mode = spm4.Stage2MaximumCoefficientCount
' Read
```

**C++ Syntax**
```
HRESULT get_Stage2MaximumCoefficientCount(long* val);
```

**Interface**
ISignalProcessingModuleFour

---

**Last Modified:**
1-Jan-2007 MX New topic
### Stage2MaximumCoefficientSum Property

**Description**
Returns the maximum sum of all Stage2 coefficients.

**VB Syntax**

```vbnet
value = spm4.Stage2MaximumCoefficientSum
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td><code>__int64* val</code> Variable to store the returned Max sum of all coefficients.</td>
</tr>
</tbody>
</table>

**Default**
Not Applicable

**Examples**

```csharp
mode = spm4.Stage2MaximumCoefficientSum 'Read
```

**C++ Syntax**

```cpp
HRESULT get_Stage2MaximumCoefficientSum(__int64* val);
```

**Interface**
ISignalProcessingModuleFour

---

**Last Modified:**
1-Jan-2007 MX New topic
## Stage2MinimumCoefficientCount Property

**Description**  
Returns the minimum number of coefficients for Stage2

**VB Syntax**  
value = spm4.Stage2MinimumCoefficientCount

**Variable (Type) - Description**

- **value** *(Long)* Variable to store the returned Min coefficient count.

- **spm4** A *SignalProcessingModuleFour* *(object)*

**Default**  
Not Applicable

**Examples**

```vbnet
mode = spm4.Stage2MinimumCoefficientCount
'Read
```

**C++ Syntax**

```cpp
HRESULT get_Stage2MinimumCoefficientCount(long* val);
```

**Interface**

*ISignalProcessingModuleFour*

---

**Last Modified:**

1-Jan-2007  
MX New topic
Stage3FilterType Property

**Description**
Sets and returns the Stage 3 filter type. This command is only used when FilterMode is set to Manual.

**VB Syntax**

```vbnet
spm4.Stage3FilterType = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>spm4</code></td>
<td>A SignalProcessingModuleFour (object)</td>
</tr>
</tbody>
</table>

`value` (String) Filter type. Chose from:

- "RECT" Rectangular Window Filter
- "TUKEY" Tukey Filter
- "PWIN" Pulse window filter

**Default**
TUKEY

**Examples**

```vbnet
spm4.Stage3FilterType = "PWIN"
```

**C++ Syntax**

```cpp
HRESULT get_Stage3FilterType(BSTR* pFType);
HRESULT put_Stage3FilterType(BSTR GType);
```

**Interface**
ISignalProcessingModuleFour

---

Last Modified:

1-Jan-2007    MX New topic
Stage3FilterTypes Property

**Description**
Returns a list of strings for the currently supported filter types that can be used for the stage 3 filter. This command is only used when FilterMode is set to False (Manual). See Stage3FilterType for a list of currently supported filter types.

**VB Syntax**
```vbnet
values = spm4.Stage3FilterTypes
```

**Variable (Type) - Description**
- `value` (Variant) Variable to store the returned filter types.
- `spm4` A SignalProcessingModuleFour (object)

**Return Type**
Variant Array

**Default**
Not Applicable

**Examples**
```vbnet
mode = spm4.Stage3FilterTypes 'Read
```

**C++ Syntax**
```cpp
HRESULT getStage3FilterTypes(VARIANT* pTypes);
```

**Interface**
ISignalProcessingModuleFour

Last Modified:
1-Jan-2007   MX New topic
# Stage3Parameter Property

**Description**
Sets and returns the Stage 3 filter parameters. Must first select the filter type using `Stage3FilterType` before setting these parameters.

Use `Stage3Parameters` to return a list of the available parameters for the currently selected filter type.

**VB Syntax**

```
spm4.Stage3Parameter(param) = value
```

**Variable**

- **(Type)** - **Description**

  - `spm4` A `SignalProcessingModuleFour` (object)
  - `param` (String) Filter parameter. Choose from:
    - "C" - Tap count (Tukey, RECT, PWIN)
    - "P" - Period (PWIN ONLY)
    - "D" - Delay (PWIN ONLY)
    - "W" - Width (PWIN ONLY)
    - "R" - Ramp Count (PWIN ONLY)
  - `value` (String) Parameter Value for the specified stage 3 parameter. Use `Stage3ParameterMaximum` and `Stage3ParameterMinimum` to return a range of values for the specified parameter.

**Default**

- **RECT**: C = 1
- **PWIN**: C=1E6, P=10ms, D=50us, W=50us, R=7
- **TUKEY**: C=1

**Examples**

```vbnet
spm4.Stage3Parameter("C") = 2

mode = spm4.Stage3Parameter("pwin") 'Read
```

**C++ Syntax**

- `HRESULT get_Stage3Parameter(BSTR pName, double* pVal);`
- `HRESULT put_Stage3Parameter(BSTR pName, double pVal);`

**Interface**

- `ISignalProcessingModuleFour`

---

*Last Modified: 1-Jan-2007*  
MX New topic
Stage3ParameterMaximum Property

**Description**  Returns maximum parameter value for the current filter type.

**VB Syntax**  
\[
\text{values} = \text{spm4.Stage3ParameterMaximum (parameter)}
\]

**Variable**  
(Type) - Description

- **value** (Variant) Variable to store the maximum parameter value.
- **spm4** A **SignalProcessingModuleFour** (object)
- **parameter** (String) Parameter name. See **Stage3Parameter Property** for a list of parameters.

**Return Type**  Double

**Default**  Not Applicable

**Examples**  
\[
\text{mode} = \text{spm4.Stage3ParameterMaximum ("c")}
\]

**C++ Syntax**  
```c++
HRESULT get_Stage3ParameterMaximum(BSTR pName, double* pVal);
```

**Interface**  ISignalProcessingModuleFour

---

Last Modified:  
1-Jan-2007  MX New topic
**Stage3ParameterMinimum Property**

**Description**
Returns minimum parameter value for the current filter type.

**VB Syntax**
```vbnet
values = spm4.Stage3ParameterMinimum (parameter)
```

**Variable**
- **Type**
  - `value` (Variant) Variable to store the minimum parameter value.
  - `spm4` A `SignalProcessingModuleFour` object

**Parameter**
- **Type**
  - `parameter` (String) Parameter name. See `Stage3Parameter Property` for a list of parameters.

**Return Type**
Double

**Default**
Not applicable

**Examples**
```vbnet
mode = spm4.Stage3ParameterMinimum ("c")  'Read
```

**C++ Syntax**
```c++
HRESULT get_Stage3ParameterMinimum(BSTR pName, double* pVal);
```

**Interface**
`ISignalProcessingModuleFour`

---

Last Modified:

1-Jan-2007  MX New topic
### Stage3Parameters Property

**Description**  
Returns the names of parameters for the current filter type. Use [Stage3FilterType Property](#) to set the filter type.

**VB Syntax**  
\[ values = spm4.Stage3Parameters \]

**Variable**  
**Type** - **Description**

- **value**  
  **Variant** Variable to store the returned parameter names.

- **spm4**  
  A [SignalProcessingModuleFour](#) object

**Return Type**  
Variant

**Default**  
Not Applicable

**Examples**  
mode = spm4.Stage3Parameters 'Read

**C++ Syntax**  

\[ HRESULT get_Stage3Parameters(VARIANT* pNames); \]

**Interface**  
ISignalProcessingModuleFour

---

Last Modified:

18-Jan-2007  
MX New topic
### StandardDeviation Property

**Description**
Returns the standard deviation of the measurement. To retrieve all 3 statistics value at the same time, use `meas.GetTraceStatistics`.

**VB Syntax**
```vbnet
stdev = meas.StandardDeviation
```

**Variable**
- **Type**: (single)
- **Description**: `stdev` - Variable to store standard deviation value
- `meas` - A Measurement *(object)*

**Return Type**
Single

**Default**
Not applicable

**Examples**
```vbnet
stdev = meas.StandardDeviation 'Read
```

**C++ Syntax**
```cpp
HRESULT get_StandardDeviation(float* stdDeviation)
```

**Interface**
IMeasurement
**StandardForClass Property - Superseded**

**Description**  
Superseded  
This command sets a **single** standard to a calibration class. Does NOT set or dictate the order for measuring the standards.  
Use GetStandardForClass and SetStandardForClass. These commands allow up to seven standards to be assigned to a cal class.

**VB Syntax**  
`calKit.StandardForClass(class, portNum) = value`

**Variable**  
(Type) - Description

- **calKit**  
  A CalKit (object). Use calKit.GetCalStandard to get a handle to the standard.

- **class**  
  (enum NACalClass) Standard. Choose from:

  1 - naClassA  
  2 - naClassB  
  3 - naClassC  
  4 - naClassD  
  5 - naClassE  
  6 - naReferenceRatioLine  
  7 - naReferenceRatioThru

**SOLT Standards**

1 - naSOLT_Open  
2 - naSOLT_Short  
3 - naSOLT_Load  
4 - naSOLT_Thru  
5 - naSOLT_Isolation

**TRL Standards**

1 - naTRL_Reflection  
2 - naTRL_Line_Reflection  
3 - naTRL_Line_Tracking  
4 - naTRL_Thru  
5 - naTRL_Isolation
**portNum** *(long)* - The port number the standard will be connected to. For example, you may have a 3.5mm connector designated for port 1, and Type N designated for port 2.

**value** *(long)* - Calibration class number. Choose a number between 1 and 8. The *<value>* numbers are associated with the following calibration classes:

<table>
<thead>
<tr>
<th>&lt;value&gt;</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S11A</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>2</td>
<td>S11B</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>3</td>
<td>S11C</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>4</td>
<td>S21T</td>
<td>Thru standard</td>
</tr>
<tr>
<td>5</td>
<td>S22A</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>6</td>
<td>S22B</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>7</td>
<td>S22C</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>8</td>
<td>S21T</td>
<td>Thru standard</td>
</tr>
</tbody>
</table>

**Return Type** Long Integer

**Default** Not Applicable

**Examples**

```
calKit.StandardForClass(naSOLT_Short, 1) = 1
```

```
Kclass = calKit.StandardForClass(naSOLT_Short, 1)
```

**C++ Syntax**

```c
HRESULT put_StandardForClass (NACalClass item, long pNum);
HRESULT get_StandardForClass (NACalClass* item, long *pNum);
```

**Interface** ICalKit
StartFrequency (Cal Set) Property

Description
Returns the start frequency that is stored in the Cal Set.

VB Syntax
value = CalSet.StartFrequency (range)

Variable
(value) - Description

CalSet: CalSet (object)

range (Long) - Choose from:
0 - Source and receiver frequency

Return Type
Double

Default
Not Applicable

Examples
start = calset.StartFrequency(1) 'Reads the start frequency stored in the cal set.

C++ Syntax
HRESULT get_StartFrequency(double *pVal)

Interface
|CalSet3
Write/Read

About Linear Frequency Sweep

StartFrequency Property

Description
Sets or returns the start frequency of the channel.
or
Sets or returns the start frequency of the segment.
or
Sets or returns the start frequency of the FOM Range.
See also Measurement2 interface

VB Syntax

\[ \text{object} \cdot \text{StartFrequency} = \text{value} \]

Variable (Type) - Description

\textit{object} \hspace{1em} \text{Any of the following:}

\textbf{Channel (object)}
\textbf{Segment (object)}
\textbf{FOMRange (object)}

\textit{value} \hspace{1em} \text{(double)} \hspace{1em} \text{- Start frequency in Hertz. Choose any number between the minimum and maximum frequencies of the analyzer.}

Return Type
Double

Default
Channel - Minimum frequency of the analyzer
Segment - 0
FOMRange - Minimum frequency of the analyzer

Examples

\texttt{chan.StartFrequency = 4.5e9 'sets the start frequency of a linear sweep for the channel object -Write}

\texttt{startfreq = Chan.StartFrequency 'Read}

C++ Syntax

\texttt{HRESULT get_StartFrequency(double *pVal)}
\texttt{HRESULT put_StartFrequency(double newVal)}

Interface
IChannel
ISegment
IFOMRange

Last modified:

1372
8-Mar-2007  Added FOMRange

Nov. 1, 2006  Removed Cal Set object. There is now a new cs.StartFreq
# StartPower Property

**Description** Sets the start power of the analyzer when sweep type is set to Power Sweep. Frequency of the measurement is set with chan.CWFrequency.

**VB Syntax**

```
object.StartPower = value
```

**Variable (Type) - Description**

- **object**
  One of the following:
  - **Channel** (object)
  - **CalSet** (object) - Read-only property

- **value** (double) - Start Power in dBm.

**Note:** The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, use cap.MaximumSourceALCPower and cap.MinimumSourceALCPower

Auto attenuation is not allowed in Power Sweep.

**Return Type** Double

**Default** 0

**Examples**

```
Chan.StartPower = -10 'Write

strtpwr = Chan.StartPower 'Read
```

**C++ Syntax**

HRESULT get_StartPower(double *pVal)
HRESULT put_StartPower(double newVal)

**Interface**

IChannel | CalSet3

---

**Last Modified:**

- 7-Jan-2008  Removed FOMRange
- 8-Mar-2007  Added FOMRange
**Start Property**

**Description**
Sets or returns the start time of either Gating or Time Domain transform windows

**VB Syntax**
```
object.Start = value
```

**Variable**

<table>
<thead>
<tr>
<th><strong>(Type) - Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code> <em>(object)</em> As Gating</td>
</tr>
<tr>
<td><em>(object)</em> As Transform</td>
</tr>
</tbody>
</table>

| **value** *(double)* - Start time in seconds. Choose any number between: ± (number of points-1) / frequency span |

**Return Type**
Double

**Default**
-10ns

**Examples**
```
Trans.Start = 4.5e-9 'sets the start time of a transform window

Gate.Start = 4.5e-9 'sets the start time of a gating window

strt = Trans.Start 'Read
```

**C++ Syntax**
```
HRESULT get_Start(double *pVal)
HRESULT put_Start(double newVal)
```

**Interface**
ITransform
IGating
## Start Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the stimulus value of the first data point for the measurement. To understand how this property is useful, see <a href="#">IMeasurement2 Interface</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = meas.Start</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Double) - Variable to store the returned value</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>Print meas.Start 'prints the stimulus value of the first data point</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT get_Start (double * Val);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
Write/Read
State Property

**Description**
Turns an Object ON and OFF.

**VB Syntax**
`object.State = value`

**Variable** *(Type) - Description*

- **object**
  Applies to any of the following objects:
  - FOM
  - Gating
  - InterfaceControl
  - LimitTest
  - Port Extension- Superseded *(See Fixturing Object)*
  - Segment
  - Transform
  - Equation

**Notes:**

- **LimitTest.State** - If using Global Pass/Fail status, trigger the PNA AFTER turning Limit testing ON.
- **Segment.State** - At least ONE segment must be ON or **Sweep Type** is automatically set to **Linear**.

**value** *(boolean) -*

- **False** - Turns `obj` OFF
- **True** - Turns `obj` ON

**Return Type**
Boolean

**Default**
Depends on the object:

- 0 - FOM
- 0 - Gating
- 0 - InterfaceControl
- 0 - LimitTest
- 0 - Port Extension
- 1 - Segment
- 0 - Transform
Examples

Seg.State = 1  'Turns the segment object ON  -Write

tran = Trans.State  'returns the state of Transform  -Read

C++ Syntax

HRESULT get_State(VARIANT_BOOL *pVal)
HRESULT put_State(VARIANT_BOOL newVal)

Interface

ISegment
IInterfaceControl
ITransform
IGating
ILimitTest
IPortExtension
IFOM
IEquation
IEmbeddedLO

Last Modified:

13-Apr-2007  Added EmbeddedLO
**Write/Read**

**State Property**

**Description**
Turns the specified pulse generator ON and OFF.

**VB Syntax**
`pulse.State (n) = value`

**Variable**

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pulse</code></td>
<td>A <code>PulseGenerator</code> (object)</td>
</tr>
<tr>
<td><code>n</code></td>
<td>(Integer) Pulse generator number. Choose from 0 to 4. 0 is the generator that pulses the ADC.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>ON (or 1) - turns pulse output ON. OFF (or 0) - turns pulse output OFF.</td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**
OFF (0)

**Examples**

```vbnet
pulse.State(1) = True 'Write
value = pulse.State(4) 'Read
```

**C++ Syntax**

```cpp
HRESULT get_State(integer pulse, bool* on_off);
HRESULT put_State(integer pulse, bool on_off);
```

**Interface**
`IPulseGenerator`

---

Last Modified:

2-Jan-2007  MX New topic
**Statistics Range Property**

**Description**
Sets the User Range number for calculating measurement statistics. Set the start and stop values for a User Range with `chan.UserRangeMin` and `chan.UserRangeMax`.

There are 16 User Ranges per channel. User ranges are applied independently to any measurement.

**VB Syntax**

```
meas.StatisticsRange = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td>value</td>
<td>(long integer) - Range Number. Choose any number between 0 and 16</td>
</tr>
<tr>
<td></td>
<td>0 is Full Span</td>
</tr>
<tr>
<td></td>
<td>1 - 16 are user-defined ranges</td>
</tr>
</tbody>
</table>

**Return Type**
Long Integer

**Default**
0

**Examples**

```vbnet
meas.StatisticsRange = 2 'Write
statrange = meas.StatisticsRange 'Read
```

**C++ Syntax**

- HRESULT get_StatisticsRange(long* rangeNumber)
- HRESULT put_StatisticsRange(long rangeNumber)

**Interface**
IMeasurement
StatusAsString Property

**Description**
Returns a string that describes the result of the last tuning sweeps.

**VB Syntax**
```
value = embedLODiag.StatusAsString
```

**Variable**
- **Type**: String
- **Description**: Variable to store the returned data.

**embedLODiag**
An EmbeddedLODiagnostic (object)

**Return Type**
(String)

**Default**
Not Applicable

**Examples**
```
data= embedLO.StatusAsString 'read
```

---

**C++ Syntax**
```
HRESULT get_StatusAsString(BSTR* status);
```

**Interface**
IEembedLODiagnostic

---

Last Modified:
12-Apr-2007    MX New topic
## StepRiseTime Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the Rise time of the stimulus in Low Pass Step Mode.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>trans.StepRiseTime = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td></td>
<td><code>trans</code> A Transform <em>(object)</em></td>
</tr>
<tr>
<td></td>
<td><code>value</code> <em>(double)</em> - Rise time in seconds. Choose any number between <em>5.0e-13</em> and <em>1.63e-14</em>.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Double</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>Examples</td>
<td><code>trans.StepRiseTime = 1.0e-14 'sets the step rise time to 100 psec. -Write</code></td>
</tr>
<tr>
<td></td>
<td><code>rt = trans.StepRiseTime 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_StepRiseTime(double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_StepRiseTime(double newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>ITransform</td>
</tr>
</tbody>
</table>
StepData Property

Description
Returns an array of data from the specified tuning sweep.

VB Syntax
value = embedLODiag.StepData(n)

Variable (Type) - Description
value (Variant Array) Variable to store the returned data.

embedLODiag An EmbeddedLODiagnostic (object)
n (Long) Tuning sweep number. Use NumberOfSweeps to find the number of sweeps taken.

Default Not Applicable

Examples
data = embedLO.StepData 3 'read

C++ Syntax
HRESULT StepData(long sweep,VARIANT* pArray);

Interface IEmbeddedLODiagnostic

Last Modified:
12-Apr-2007 MX New topic
### StepTitle Property

**Description**

Returns the title of the specified tuning sweep.

**VB Syntax**

```
value = embedLODiag.StepTitle (n)
```

**Variable (Type) - Description**

- `value` (String) Variable to store the returned data.
- `n` (Long) Tuning sweep number. Use `NumberOfSweeps` to find the number of sweeps taken.

**Default**

Not Applicable

**Examples**

```
data = embedLO.StepTitle 3 'read
```

**C++ Syntax**

```
HRESULT StepTitle (long sweep, BSTR * title);
```

**Interface**

IEembedLODiagnostic

---

_Last Modified:  13-Apr-2007  MX New topic_
### StimulusValues Property

**Description**
Returns multiple X-axis frequency arrays (source and response) needed by frequency offset measurements. The arrays contain one frequency value for each data point.

**VB Syntax**
```vbnet
value = calSet.StimulusValues (range)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calSet</code></td>
<td><code>CalSet (object)</code></td>
<td>Read-only property</td>
</tr>
<tr>
<td><code>range</code></td>
<td><code>Long</code></td>
<td>Specify 0 to return source frequencies.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify 1 to return response frequencies.</td>
</tr>
</tbody>
</table>

**Return Type**
1-dimensional variant array

**Default**
Not Applicable

**Examples**
```vbnet
array = CalSet.StimulusValues 'Read
```

**C++ Syntax**
```cpp
HRESULT get_StimulusValues (long range, VARIANT* vals)
```

**Interface**
ICalSet3

---

Last Modified:

19-Oct-2007 Fixed title and syntax
StopFrequency Property

**Description**
Sets or returns the stop frequency of the channel
or
Sets or returns the stop frequency of the segment.
or
Sets or returns the stop frequency of the FOMRange.

see also Measurement2 interface

**VB Syntax**

```
object.StopFrequency = value
```

**Variable (Type) - Description**

- **object**
  - Any of the following:
    - Channel (object)
    - Segment (object)
    - FOMRange (object)

- **value** (double) - Stop frequency in Hertz. Choose any number between the minimum and maximum frequencies of the analyzer.

**Return Type**
Double

**Default**
Channel - Maximum frequency of the analyzer.
Segment - 0
FOMRange - Maximum frequency of the analyzer.

**Examples**

```
chan.StopFrequency = 4.5e9 'sets the stop frequency for the channel
object.Write

stopfreq = Chan.StopFrequency 'Read
```

**C++ Syntax**

```
HRESULT get_StopFrequency(double *pVal)
HRESULT put_StopFrequency(double newVal)
```

**Interface**
IChannel
ISegment
IFOMRange

Last modified:
8-Mar-2007     Added FOMRange

Nov. 1, 2006     Removed Cal Set object - created cs.stopfreq
Read-only

StopFrequency (Cal Set) Property

**Description**
Returns the stop frequency that is stored in the Cal Set.

**VB Syntax**
```
value = CalSet.StopFrequency(range)
```

**Variable**
- **value** *(double)* - returned Stop frequency in Hertz.
- **CalSet** *(object)*
- **range** *(Long)* Choose from:
  0 - Source and receiver frequency

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```
stop = calset.StopFrequency(1)'Reads the stop frequency stored in the cal set.
```

**C++ Syntax**
```
HRESULT get_StopFrequency(double *pVal)
```

**Interface**
-CalSet3

---

**Last modified:**
Nov. 1, 2006  New command - split from ch.StopFreq
**StopPower Property**

**Description**
Sets the Stop Power of the analyzer when sweep type is set to Power Sweep. Frequency of the measurement is set with chan.CWFrequency.

**VB Syntax**
`object.StopPower = value`

**Variable (Type) - Description**
- **object**
  - One of the following:
    - `Channel` (object)
    - `CalSet` (object) - Read-only property
- **value** (double) - Stop Power in dB. Start Power in dB.

**Note:** The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, use `cap.MaximumSourceALCPower` and `cap.MinimumSourceALCPower`.

Auto attenuation is not allowed in Power Sweep.

**Return Type**
Double

**Default**
0

**Examples**
```vbnet
Chan.StopPower = -10 'Write

stppwr = Chan.StopPower 'Read
```

**C++ Syntax**
```c++
HRESULT get_StopPower(double *pVal)
HRESULT put_StopPower(double newVal)
```

**Interface**
- IChannel
- |CalSet3

**Last Modified:**
- 7-Jan-2008 Removed FOMRange
- 8-Mar-2007 Added FOMRange
### Stop Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the Stop time of either Gating or Time Domain transform windows</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>object.Stop = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
</tbody>
</table>
| `object` | *(object)* As Gating  

*or*  

*(object)* As Transform |
| `value` | *(double)* - Start time in seconds. Choose any number between:  

± (number of points-1) / frequency span |
| **Return Type** | Double |
| **Default** | 10 ns |
| **Examples** | `Trans.Stop = 4.5e-9 'sets the stop time of a transform window - Write`  

`Gate.Stop = 4.5e-9 'sets the stop time of a gating window - Write`  

`stp = Trans.Stop 'Read` |
| **C++ Syntax** | HRESULT get_Sop(double *pVal)  

HRESULT put_Sop(double newVal) |
| **Interface** | ITransform  

IGating |
**Stop Property**

**Description**  Returns the stimulus value of the last data point for the measurement. To understand how this property is useful, see [IMeasurement2 Interface](#).

**VB Syntax**  

```vb
value = meas.Stop
```

**Variable (Type) - Description**

- `value` (Double) Variable to store the returned value
- `meas` A Measurement (object)

**Return Type**  Double

**Default**  Not Applicable

**Examples**  

```vb
Print meas.Stop 'prints the stimulus value of the last data point
```

**C++ Syntax**  

```cpp
HRESULT get_Stop(double * Val);
```

**Interface**  IMeasurement2
**strPort2Pdeembed_S2PFile Property**

**Description**
Sets and returns the 2 port De-embedding .S2P file name for the specified port number. Model is applied when both the file name is specified and **User** is specified using **Port2PdeembedCktModel Property**.

Learn more about S2P files.

**Note:** This command affects ALL measurements on the channel.

**VB Syntax**
```vbnet
fixture.strPort2Pdeembed_S2PFile(port) = value
```

**Variable** *(Type) - Description*

- **fixture** *(A Fixturing (object))*
- **port** *(Integer)* Port number to receive circuit model.
- **value** *(String)* Full path, file name, and extension (.s2P) of the de-embedding circuit.

Files are typically stored in "C:\Program Files\Agilent\Network Analyzer\Documents"

**Return Type**
String

**Default**
Not Applicable

**Examples**
```vbnet
fixture.strPort2Pdeembed_S2PFile(2) = "C:\Program Files\Agilent\Network Analyzer\Documents\myFile.s2p" 'Write

value = fixture.strPort2Pdeembed_S2PFile(1) 'Read
```

**C++ Syntax**
```cpp
HRESULT get_strPort2Pdeembed_S2PFile(short port BSTR *bstrFile)
HRESULT put_strPort2Pdeembed_S2PFile(short port BSTR bstrFile)
```

**Interface**
IFixturing
### strPortMatch_S2PFile Property

#### Description
Sets and returns the Port Matching 'S2P' file name for the specified port number. Model is applied when both the file name is specified and **User** is specified using PortMatchingCktModel Property.

Learn more about S2P files.

**Note:** This command affects ALL measurements on the channel.

#### VB Syntax
```
fixture.strPort2PMatch_S2PFile(port) = value
```

#### Variable (Type) - Description

- **fixture** (A Fixturing (object))
- **port** (Integer) Port number to receive circuit model.
- **value** (String) Full path, file name, and extension (.s2P) of the matching circuit.

Files are typically stored in "C:\Program Files\Agilent\Network Analyzer\Documents".

#### Return Type
String

#### Default
Not Applicable

#### Examples
```
fixture.strPort2PMatch_S2PFile(2) = "C:\Program Files\Agilent\Network Analyzer\Documents\myFile.s2p" 'Write

value = fixture.strPort2PMatch_S2PFile(1) 'Read
```

#### C++ Syntax
```
HRESULT get_strPort2PMatch_S2PFile(short port BSTR *bstrFile)
HRESULT put_strPort2PMatch_S2PFile(short port BSTR bstrFile)
```

#### Interface
IFixturing
Read/Write
SweepEndMode Property

**Description**
Sets and reads the event that will cause the Sweep End line to go to a low state. The line will return to a high state after the appropriate calculations are complete.

*Note:* This line is connected to the following pins on the HANDLER IO connector and AUX IO connector in the PNA. Therefore, this command will affect both of these connectors in the same way.

**VB Syntax**

```
object.SweepEndMode = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code></td>
<td>(object) - A HandlerIO or AuxIO object</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum as NASweepEndMode) Choose from:</td>
</tr>
</tbody>
</table>

- 0 - naSweep - the line goes low when each sweep is complete
- 1 - naChannelSweep - the line goes low when all the sweeps for each channel is complete.
- 2 - naGlobalSweep - the line goes low when all sweeps for all triggerable channels are complete.

**Return Type**
Long Integer

**Default**
0 - naSweep

**Examples**

```
HWAuxIO.PassFailMode = naSweep 'Write
value = HWAuxIO.PassFailMode 'Read
```

**C++ Syntax**

```
HRESULT put_SweepEndMode ( tagNASweepEndMode Mode );
HRESULT get_SweepEndMode ( tagNASweepEndMode* Mode );
```

**Interface**

IHWAuxIO
IHWMaterialHandlerIO
**SweepHoldOff Property**

**Description**
Returns a boolean that represents the state of SweepHoldoff line (pin2) of the External Test Set connector.

**VB Syntax**
```
value = ExtIO.SweepHoldOff
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(boolean) - Variable to store the returned data</td>
</tr>
<tr>
<td>ExtIO</td>
<td>(object) - An External IO object</td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

- **False** - indicates the line is being held at a TTL Low
- **True** - indicates the line is being held at a TTL High

**Default**
Not Applicable

**Examples**
```
value = ExtIO.SweepHoldOff
```

**C++ Syntax**
```
HRESULT get_SweepHoldOff( VARIANT_BOOL* bValue);
```

**Interface**
IHWEexternaTestSetIO
**SweepGenerationMode Property**

**Description**
Sets the method used to generate a sweep: continuous ramp (analog) or discrete steps (stepped).

**VB Syntax**

```
object.SweepGenerationMode = value
```

**Variable (Type) - Description**

- `object` Channel (object)
- or
- CalSet (object) - Read-only property

**value (enum NASweepGenerationModes) - Choose either:**

- **0 - naSteppedSweep** - source frequency is CONSTANT during measurement of each displayed point. More accurate than Analog. Dwell time can be set in this mode.
- **1 - naAnalogSweep** - source frequency is continuously RAMPING during measurement of each displayed point. Faster than Stepped. Sweep time (not dwell time) can be set in this mode.

**Return Type**
Long Integer

**Default**
Analog

**Examples**

```vbnet
Chan.SweepGenerationMode = naAnalogSweep 'Write
swpgen = Chan.SweepGenerationMode 'Read
```

**C++ Syntax**

```cpp
HRESULT get_SweepGenerationMode(tagNASweepGenerationModes* pVal)
HRESULT put_SweepGenerationMode(tagNASweepGenerationModes newVal)
```

**Interface**
IChannel, ICalSet3
### Description
Sets the Sweep time of the analyzer. If sweep time accuracy is critical, use ONLY the values that are attained using the up and down arrows next to the sweep time entry box. See Sweep Time.

### VB Syntax
```
object.SweepTime = value
```

### Variable (Type) - Description

<table>
<thead>
<tr>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>object</strong></td>
<td><strong>Channel (object)</strong></td>
</tr>
<tr>
<td>or</td>
<td><strong>Segment (object)</strong> first set <strong>SweepTimeOption</strong> to true.</td>
</tr>
<tr>
<td>or</td>
<td><strong>CalSet (object)</strong> - Read-only property</td>
</tr>
</tbody>
</table>

### Value (double) - Sweep time in seconds. The maximum sweep time of the PNA is 86400 seconds (1 day). To set the fastest sweep speed possible, set this value to 0.

### Return Type
Double

### Default
0

### Examples
```
chan.SweepTime = 3e-3 'Write
swptme = chan.SweepTime 'Read
```

### C++ Syntax
```
HRESULT get_SweepTime(double *pVal)
HRESULT put_SweepTime(double newVal)
```

### Interface
- IChannel
- CalSet3
- ISegment2
SweepTimeOption Property

Description
Enables the Sweep time or Dwell time to be set on individual sweep segments. This property must be set True before the sweep or dwell time commands are sent. Otherwise, those commands will be ignored.

VB Syntax
```
segs.SweepTimeOption = state
```

Variable
- **segs** (Type) - Description
  - A Segments collection (object)
- **state** (boolean)
  - **True** - Enables Sweep or Dwell time to be set independently.
  - **False** - Disables Sweep or Dwell time from being set independently.

Return Type
Boolean

Default
False

Examples
```
segs.SweepTimeOption = True 'Write
timeOption = SweepTimeOption 'Read
```

C++ Syntax
```
HRESULT get_SweepTimeOption(VARIANT_BOOL *pVal)
HRESULT put_SweepTimeOption(VARIANT_BOOL newVal)
```

Interface
ISegments3

Last modified:
9/29/06 MQQ New command
SweepType Property

Description
Sets and returns the type of sweep.

VB Syntax

\[ \text{object}.\text{SweepType} = \text{value} \]

Variable

\((\text{Type}) - \text{Description}\)

- \text{object} One of the following:
  - \text{Channel} (object)
  - \text{FOMRange} (object) Must be an \text{UNCOPLED} range.
  - \text{CalSet} (object) - Read-only property

\(\text{value} (\text{enum NAoSweepTypes}) - \text{Choose from:}\)

- 0 - naLinearSweep
- 1 - naLogSweep
- 2 - naPowerSweep
- 3 - naCWT imeSweep
- 4 - naSegmentSweep

\text{Note}: Sweep type cannot be set to Segment sweep if there are no segments turned ON. A segment is automatically turned ON when an application is created.

Return Type
Long Integer

Default
naLinearSweep

Examples

\[ \text{chan.SweepType} = \text{naPowerSweep} \text{'Write} \]

\[ \text{swptyp} = \text{chan.SweepType} \text{'Read} \]

C++ Syntax

\text{HRESULT get}_\text{SweepType(tagNASweepTypes* pVal)}
\text{HRESULT put}_\text{SweepType(tagNASweepTypes newVal)}

Interface

IChannel
|CalSet3
IFOMRange

Last Modified:

8-Mar-2007 Added FOMRange
### SystemImpedanceZ0 Property

**Description**
Sets and returns the impedance for the analyzer.

**VB Syntax**

```vbnet
app.SystemImpedanceZ0 = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) Analyzer Impedance. Choose any number between 0 and 1000 ohms.</td>
</tr>
</tbody>
</table>

**Return Type**
Double

**Default**
50

**Examples**

```vbnet
app.SystemImpedanceZ0 = 75 'Write

z0 = app.SystemImpedanceZ0 'Read
```

**C++ Syntax**

```cpp
HRESULT get_SystemImpedanceZ0(double dSystemZ0)
HRESULT put_SystemImpedanceZ0(double *pdSystemZ0)
```

**Interface**
IApplication
## SystemName Property

**Description**  
Returns the computer name of the PNA.

**VB Syntax**  
```vbnet
name = app.SystemName
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>(String) Variable to store the returned computer name.</td>
</tr>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
</tbody>
</table>

**Return Type**  
String

**Default**  
Not Applicable

**Examples**  
```vbnet
name = app.SystemName
```

**C++ Syntax**  
```c++
HRESULT SystemName(BSTR* computerName)
```

**Interface**  
IApplication
## TargetValue Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets the target value for the marker when doing Target Searches (<a href="#">SearchTargetLeft</a>, <a href="#">SearchTarget</a>, <a href="#">SearchTargetRight</a>).</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.TargetValue = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>mark</strong> (Type) - Description</td>
</tr>
<tr>
<td></td>
<td><code>mark</code> A Marker (object)</td>
</tr>
<tr>
<td></td>
<td><code>value</code> (single) - Target value. Choose any number between: <strong>-500</strong> and <strong>500</strong></td>
</tr>
<tr>
<td>Return Type</td>
<td>Single</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark.TargetValue = 10.5</code> 'Write</td>
</tr>
<tr>
<td></td>
<td><code>target = mark.TargetValue</code> 'Read</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_TargetValue(float *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_TargetValue(float newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
### TestPortPower Property

#### Description
Sets or returns the RF power level for the channel or
Sets or returns the RF power level of the segment.

#### VB Syntax
```
object.TestPortPower(srcPort) = value
```

#### Variable *(Type) - Description*

- **object**
  - A Channel *(object)* - to set coupled power, use chan.CouplePorts. If CouplePorts = False, then each port power can be set independently. Otherwise, chanTestPortPower (1) = value sets power level at both ports.
  - or
  - A CalSet *(object)*
  - or
  - A Segment *(object)*

- **srcPort** *(long integer) - Source Port number.*
  - Use GetPortNumber to return the port number of a source that only has a string name, such as an External Source.

- **value** *(double) - RF Power in dBm.*
  - Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, use cap.MaximumSourceALCPower and cap.MinimumSourceALCPower
  - Actual achievable leveled power depends on frequency.

- **Return Type** Double

- **Default** 0

#### Examples
```
chan.TestPortPower(1) = 5 'sets the port 1 RF power level for the channel
object -Write

powerlev = Chan.TestPortPower(1) 'Read
```

#### C++ Syntax
```
HRESULT get_TestPortPower(long port, double *pVal)
HRESULT put_TestPortPower(long port, double newVal)
```

#### Interface
IChannel
ICalSet3
ISegment

Last Modified: 1403
### TestSetType Property

**Description**
Returns the Test Set Type (model) that was used for the Cal Set.

**VB Syntax**
```vbnet
TSType = calset.TestSetType
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSType</td>
<td>String</td>
<td>Variable to store the returned test set model.</td>
</tr>
<tr>
<td>calset</td>
<td>Cal Set object.</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Depends on the test set.

**Example**
```vbnet
TSType = calset.TestSetType
```

**C++ Syntax**
```cpp
HRESULT get_OutputPorts(BSTR *mapping);
```

**Interface**
ICalset5

---

Last modified:

9/18/06  MQ Added for multiport
## Text Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies an equation or expression to be used on the measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>eq.Text = eqText</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>eq</code></td>
<td>Equation (object)</td>
</tr>
<tr>
<td><code>eqText</code></td>
<td>(String) - Any valid equation or expression.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>eq.Text = &quot;foo=S11/S21&quot;</code></td>
</tr>
<tr>
<td><code>equation = eq.Text</code></td>
<td><em>Read</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_Text(BSTR *equation)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Text(BSTR  equation)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEquation</td>
</tr>
</tbody>
</table>
## ThruCalMethod Property

**Description**
Sets and returns the method for performing the thru portion of the calibration.

**VB Syntax**
```
obj.ThruCalMethod = value
```

**Variable**
- **obj** (Type) - Description
  - **SMCType** (object)
  - or
    - **VMCType** (object)

- **value** (String) Specifies the Thru method. Case insensitive - include spaces. Choose from:
  - "Default"
  - "Flush Thru" or "FLUSH"
  - "Unknown Thru" or "UNKN"
  - "Adapter Removal" or "ADAP"

**Return Type**
String

**Default**
Default

**Examples**
```
SMC.ThruCalMethod = "UNKN"
```

**C++ Syntax**
```
HRESULT put_ThruCalMethod(enum NAThruCalMethod thruMethod);
HRESULT get_ThruCalMethod(enum NAThruCalMethod *thruMethod);
```

**Interface**
- SMCType
- VMCType
**Read/Write**

**ThruCalMethod Property**  **Superseded**

**Description**

This command is replaced by PathThruMethod Property.

Sets and returns the method for performing the Cal Method and the THRU portion of the calibration.

**VB Syntax**

```vb
guidedCal .ThruCalMethod = value
```

**Variable (Type) - Description**

`guidedCal` GuidedCalibration (object)

`value` (Enum as NAThruCalMethod) Choose from:

- 0 - `naDefaultCalMethod` - allow the PNA to choose the best possible method (from the following) depending on whether the device or ECal module is insertable or non-insertable and given the model number of the PNA. (default selection if omitted.)
- 1 - `naAdapterRemoval` - Perform Adapter removal calibration.
- 2 - `naFlushThru` - Perform Flush Thru calibration.
- 3 - `naDefinedThru` - Perform Defined Thru calibration. If performing an ECal, this is the Thru standard in the ECal module.
- 4 - `naUnknownThru` - Perform Unknown Thru calibration.
- 5 - `naSOLT` - Perform SOLT calibration
- 6 - `naTRL` - Perform TRL calibration
- 7 - `naQSOLT` - Perform QSOLT calibration.

Learn more about Cal Methods

**Return Type**

Enum

**Default**

0 - `naDefaultCalMethod`

**Examples**

```vb
guided.ThruCalMethod = naDefinedThru
```

**C++ Syntax**

```c++
HRESULT get_ThruCalMethod(enum NAThruCalMethod *thruMethod);
HRESULT put_ThruCalMethod(enum NAThruCalMethod thruMethod);
```

**Interface**

`IGuidedCalibration`

---

Last Modified:

30-Apr-2007   MX Superseded
**ThruPortList Property**

**Description**

**Note:** Available only on PNA releases 5.0 and greater.

Sets and returns the thru connection port pairs for the calibration.

- For 3-port cals, specify at least two pairs.
- For 4-port cals, specify at least three pairs.
- For highest accuracy, specify more than the minimum pairs.
- For a 2-port cal, there is only one port pair. It is the only pair in the list and it is required.

Learn more about Thru method and port pairings.
See an example of a 4-port guided calibration using COM.

**VB Syntax**

```
guidedCal.ThruPortList = t1a, t1b, t2a, t2b, t3a, t3b
```

**Variable** (Type) - Description

- **guidedCal** (GuidedCalibration) (object)
- **t1a, t1b...** (Variant) Port numbers in pairs - a one-dimensional array of Long integers.
  - t1a, t1b  (Thru1 - port A and port B)
  - t2a, t2b  (Thru2 - port A and port B)
  - t3a, t3b  (Thru3 - port A and port B)

**Return Type** Variant - a one-dimensional array of Long integers.

**Default** Not Applicable

**Example**

```
thruList = Array(1,2,1,3,1,4)
guided.ThruPortList = thruList
```

'Sets the following three thru connections for a 4-port calibration:
Thru 1 - ports 1 and 2
Thru 2 - ports 1 and 3
Thru 3 - ports 1 and 4

**C++ Syntax**

```
HRESULT get_ThruPortList(VARIANT* portList);
HRESULT put_ThruPortList(VARIANT portList);
```

**Interface** IGuidedCalibration
### Title Property

**Description**
 Writes or reads a custom title for the window. Newer entries replace (not append) older entries. Turn the title ON and OFF with TitleState.

**VB Syntax**

```vbnet
win.Title = string
```

**Variable**

- **(Type)** - Description
  - `win` A NaWindow (object)
  - `string` (long) - Title limited to 50 characters.

**Return Type**

String

**Default**

Null

**Examples**

```vbnet
win.Title = "Hello World" 'Write

titl = win.Title 'Read
```

**C++ Syntax**

```c++
HRESULT get_Title(BSTR *title)
HRESULT put_Title(BSTR title)
```

**Interface**

INAWindow
TitleState Property

**Description**
Turns ON and OFF the window title. Write a window title with `Title`.

**VB Syntax**
```
win.TitleState = state
```

**Variable**
- **(Type)** - Description
  - `win` A NaWindow *(object)*
  - `state` *(boolean)*
    - `True` - Title ON
    - `False` - Title OFF

**Return Type**
Boolean

**Default**
False

**Examples**
```
win.TitleState = True 'Write

titlestate = win.TitleState 'Read
```

**C++ Syntax**
- HRESULT get_TitleState(VARIANT_BOOL* bState)
- HRESULT put_TitleState(VARIANT_BOOL bState)

**Interface**
INAWindow
TotalNumberOfPoints Property

**Description**
Read the total number of points a complete GCA measurement will generate.

- For 2D modes, this is Frequency * Power points
- For SMART Sweep, this is Frequency points.

The total can NOT exceed the PNA maximum.
See Frequency and Power points.

**VB Syntax**

```vbnet
value = gca.TotalNumberOfPoints
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>(integer) Variable to store the returned total number of points</td>
</tr>
<tr>
<td><code>gca</code></td>
<td>A GainCompression (object)</td>
</tr>
</tbody>
</table>

**Return Type**
Integer

**Default**
5226 (201 * 26)

**Example**

```vbnet
totPoints = gca.TotalNumberOfPoints 'Read
```

**C++ Syntax**

```cpp
HRESULT get_TotalNumberOfPoints(int* pVal)
```

**Interface**
IGainCompression

---

**Last Modified:**
11-Sep-2007  MX New topic
**Touchscreen Property**

**Description**
Sets and reads the state of the PNA-X Touchscreen (ON and OFF).
This setting remains until changed again from the front-panel or remote command.

**VB Syntax**
```vbnet
app.Touchscreen = state
```

**Variable** *(Type)* - Description

- `app` An Application *(object)*
- `state` *(boolean)*
  - `False (0)` - Disables use of Touchscreen
  - `True (1)` - Enables use of Touchscreen

**Return Type**
Boolean

- `False` - OFF
- `True` - ON

**Default**
TRUE when shipped from factory.

**Examples**
```vbnet
app.Touchscreen = True 'Write
```
```vbnet
coupl = app.Touchscreen 'Read
```

**C++ Syntax**
```cpp
HRRESULT put_Touchscreen(VARIANT_BOOL bState)
HRRESULT get_Touchscreen(VARIANT_BOOL *bState)
```

**Interface**
IApplication12

---

Last Modified:
23-Feb-2007   MX New topic
### TraceMath Property

**Description**
Performs math operations on the measurement object and the trace stored in memory. (There MUST be a trace stored in Memory to perform math. See Meas.DataToMemory method.)

**VB Syntax**
```vbnet
meas.TraceMath = value
```

**Variable (Type) - Description**
- **meas** A measurement (object)
- **value** (enum NAMathOperation) - Choose from:
  - 0 - naDataNormal
  - 1 - naDataMinusMemory
  - 2 - naDataPlusMemory
  - 3 - naDataDivMemory
  - 4 - naDataTimesMemory

**Return Type**
NAMathOperation

**Default**
Normal (0)

**Examples**
```vbnet
meas.TraceMath = naDataMinusMemory 'Write
mathOperation = meas.TraceMath 'Read
```

**C++ Syntax**
```cpp
HRESULT get_TraceMath(tagNAMathOperation* pMathOp)
HRESULT put_TraceMath(tagNAMathOperation mathOp)
```

**Interface**
IMeasurement
TraceTitle Property

**Description**
Writes and reads data for the trace title area.

The trace title is embedded in the [trace status field](#).
The title is turned ON and OFF using [TraceTitleState](#).

**VB Syntax**
```vb
meas.TraceTitle = value
```

**Variable**

<table>
<thead>
<tr>
<th><strong>Type</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(string) - Title to be displayed. Any characters (no spaces), enclosed with quotes.</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Not Applicable

**Examples**
```vb
meas.TraceTitle = "My new s11 measurement"
```
```vb
title = TraceTitle 'Read
```

**C++ Syntax**
```cpp
HRESULT get_TraceTitle(BSTR *title);
HRESULT put_TraceTitle(BSTR title);
```

**Interface**
IMeasurement8

---

Last Modified:
16-Jan-2007
MX New topic
TraceTitleState Property

**Description**
Turns display of the Trace Title ON or OFF. When turned OFF, the previous trace title returns. Create a trace title using TraceTitle Property

**VB Syntax**
```vbnet
meas.TraceTitleState = value
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
<td></td>
</tr>
<tr>
<td>value</td>
<td>(boolean) - Choose from:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>True - Turns the trace title ON</td>
<td></td>
</tr>
<tr>
<td></td>
<td>False - Turns the trace title OFF</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**
False

**Examples**
```vbnet
meas.TraceTitleState = False

title = TraceTitleState 'Read
```

**C++ Syntax**
```cpp
HRESULT get_TraceTitleState(VARIANT_BOOL *isTitleON);
HRESULT put_TraceTitleState(VARIANT_BOOL isTitleON);
```

**Interface**
IMeasurement8

---

Last Modified:
18-Jun-2007   MX New topic
### Tracking Property

**Description**
This property, when on, executes the search function `(marker.SearchFunction)` every sweep. In effect, turning Tracking ON is the same as executing one of the immediate, one-time, "Search..." methods (such as `SearchMin`, `SearchMax`) for every sweep.

**VB Syntax**

```vbnet
mark.Tracking = state
```

**Variable**

<table>
<thead>
<tr>
<th><strong>(Type)</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mark</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td><code>state</code></td>
<td><em>(boolean)</em> - Tracking state. Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - Tracking OFF</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - Tracking ON</td>
</tr>
</tbody>
</table>

**Return Type**

Boolean

**Default**

False

**Examples**

```vbnet
mark.Tracking = False 'Write
markTracking = mark.Type 'Read
```

**C++ Syntax**

```c++
HRESULT put_Tracking(VARIANT_BOOL bOn)
HRESULT get_Tracking(VARIANT_BOOL * pbOn)
```

**Interface**

`IMarker`
**TriggerDelay Property**

**Description**
Sets and reads the trigger delay for all measurements (GLOBAL). This delay is only applied while in `app.Source = naTriggerSourceExternal` and `trigsetup.Scope = naGlobalTrigger`. After an external trigger is applied, the start of the sweep is delayed for the specified delay value plus any inherent latency.

To apply a trigger delay for a channel only, use `ExternalTriggerDelay Property`.

**VB Syntax**

```vba
app.TriggerDelay = value
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>value</td>
<td>Double- Trigger delay value in seconds. Range is from 0 to 107</td>
</tr>
</tbody>
</table>

**Return Type**

Double

**Default**

0

**Examples**

```vba
app.TriggerDelay = .003  'Write
delay = app.TriggerDelay 'Read
```

**C++ Syntax**

```cpp
HRESULT get_TriggerDelay(double *delay);
HRESULT put_TriggerDelay(double delay)
```

**Interface**

IAplication
**TriggerInPolarity Property**

**Description**  
Specifies the polarity of the trigger IN signal to which the PNA will respond.

**VB Syntax**  
`auxTrig.TriggerInPolarity = value`

**Variable**  
(Type) - Description

- **auxTrig**  
  An **AuxTrigger** (object)

- **value**  
  (enum NATriggerPolarity) - Choose from:
  - **naTriggerPositive**  
    PNA responds to leading edge or HIGH level
  - **naTriggerNegative**  
    PNA responds to trailing edge or LOW level.

Set Edge or Level triggering using **TriggerInType Property**

**Return Type**  
Enum

**Default**  
NEGative

**Examples**

- `auxTrig.TriggerInPolarity = naTriggerPositive` 'Write
- `value = auxTrig.TriggerInPolarity` 'Read the value

**C++ Syntax**

- `HRESULT get_TriggerInPolarity(enum NATriggerPolarity *val);`
- `HRESULT put_TriggerInPolarity(enum NATriggerPolarity val);`

**Interface**  
IAuxTrigger

---

Last Modified:

14-Dec-2006  
MX New topic
# TriggerlnType Property

**Description**
Specifies the type of aux trigger input being supplied to the PNA.

**VB Syntax**

```vbnet
auxTrig.TriggerInType = value
```

**Variable**

- **(Type) - Description**
  - `auxTrig` An [AuxTrigger](object) object
  - `value` (enum NATriggerSignalType) Choose from:
    - `naTriggerEdge` PNA responds to the leading edge of a signal
    - `naTriggerLevel` PNA responds to the level (HIGH or LOW) of a signal

**Return Type**
Enum

**Default**

**Examples**

```vbnet
auxTrig.TriggerInType = naTriggerEdge 'Write

value = auxTrig.TriggerInType 'Read the value
```

**C++ Syntax**

```cpp
HRESULT get_TriggerInType(enum NATriggerSignalType *val);
HRESULT put_TriggerInType(enum NATriggerSignalType val);
```

**Interface**
IAuxTrigger

---

Last Modified: 14-Dec-2006
MX New topic
Write/Read About Auxiliary Triggering

**TriggerOutDuration Property**

**Description**
Specifies the width of the output pulse, which is the time that the Aux trigger output will be asserted.

**VB Syntax**

```vb
auxTrig.TriggerOutDuration = value
```

**Variable**

- **(Type)**: Description
- **auxTrig**: An **AuxTrigger** *(object)*
- **value**: *(single)* - Duration value in seconds. Choose a value between 1E-6 and 1.

**Return Type**
Double

**Default**
1E-6 sec

**Examples**

```vb
auxTrig.TriggerOutDuration = 1e-3 'Write
value = auxTrig.TriggerOutDuration 'Read the value
```

**C++ Syntax**

```cpp
HRESULT get_TriggerOutDuration(double *val);
HRESULT put_TriggerOutDuration(double val);
```

**Interface**
IAuxTrigger

---

Last Modified:

14-Dec-2006 MX New topic
**TriggerOutInterval Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifies how often a trigger output signal is sent.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>auxTrig.TriggerOutInterval = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>auxTrig</code></td>
<td>An <code>AuxTrigger</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(single) - Choose from:</td>
</tr>
<tr>
<td>0 - <code>naTriggerModePoint</code></td>
<td>a single data point is measured with each trigger signal the channel receives. Subsequent trigger signals continue to go to the channel in Point mode until the channel measurements are complete. This is effectively the same as trigger point mode.</td>
</tr>
<tr>
<td>1 - <code>naTriggerModeMeasurement</code></td>
<td>entire traces are swept with a trigger signal. which and how many traces depends on the Scope setting.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Enum</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>1 - <code>naTriggerModeMeasurement</code></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>auxTrig.TriggerOutInterval = naTriggerModeMeasurement 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>value = auxTrig.TriggerOutInterval 'Read the value</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_TriggerOutInterval(enum NATriggerMode *val);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_TriggerOutInterval(enum NATriggerMode val);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IAuxTrigger</td>
</tr>
</tbody>
</table>

Last Modified:

14-Dec-2006     MX New topic
### TriggerOutPolarity Property

**Description**
Specifies the polarity of the trigger output signal being supplied by the PNA.

**VB Syntax**
```vb
auxTrig.TriggerOutPolarity = value
```

**Variable** *(Type) - Description*

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>auxTrig</td>
<td>An <strong>AuxTrigger</strong> (object)</td>
</tr>
<tr>
<td>value</td>
<td>(enum NATriggerPolarity) - Choose from:</td>
</tr>
<tr>
<td></td>
<td><strong>naTriggerPositive</strong> PNA sends positive going (active HIGH) pulse.</td>
</tr>
<tr>
<td></td>
<td><strong>naTriggerNegative</strong> PNA sends negative going (active LOW) pulse.</td>
</tr>
</tbody>
</table>

**Return Type**
Enum

**Default**

**Examples**
```vb
auxTrig.TriggerOutPolarity = naTriggerPositive 'Write
value = auxTrig.TriggerOutPolarity 'Read the value
```

**C++ Syntax**
```cpp
HRESULT get_TriggerOutPolarity(enum NATriggerPolarity *val);
HRESULT put_TriggerOutPolarity(enum NATriggerPolarity val);
```

**Interface**
IAuxTrigger

---

**Last Modified:**
14-Dec-2006    MX New topic
## TriggerOutPosition Property

**Description**  
Specifies whether the Aux trigger out signal is sent Before or After the acquisition.

**VB Syntax**  
`auxTrig.TriggerOutPosition = value`

**Variable**  
*(Type) - Description*

- **auxTrig**  
  An [AuxTrigger](object) (object)

- **value**  
  (enum NATriggerPosition) Choose from:
  - **naTriggerOutBeforeAcquire**  
    Use if the external device needs to be triggered before the data is acquired, such as a power meter.
  - **naTriggerOutAfterAcquire**  
    Use if the external device needs to be triggered just after data has been acquired, such as an external source. This could be more efficient since it allows the external device to get ready for the next acquisition at the same time as the PNA.

**Return Type**  
Enum

**Default**  
naTriggerOutAfterAcquire

**Examples**  
`auxTrig.TriggerOutPosition = naTriggerOutAfterAcquire 'Write`

`value = auxTrig.TriggerOutPosition 'Read the value`

**C++ Syntax**  
`HRESULT get_TriggerOutPosition(enum NATriggerPosition *val);`

`HRESULT put_TriggerOutPosition(NATriggerPosition val);`

**Interface**  
IAuxTrigger

---

Last Modified:

14-Dec-2006  
MX New topic
### TriggerOutputEnabled Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Enables the PNA to send trigger signals out the rear-panel TRIGGER OUT connector. For more information, see External triggering.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>trigsetup.TriggerOutputEnabled = boolean</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>trigsetup</code></td>
<td>A TriggerSetup2 (object)</td>
</tr>
<tr>
<td><code>boolean</code></td>
<td>Choose from:</td>
</tr>
<tr>
<td><code>False</code></td>
<td>PNA does NOT send output trigger signals.</td>
</tr>
<tr>
<td><code>True</code></td>
<td>PNA sends output trigger signals.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>False</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>trigsetup.TriggerOutputEnabled = True 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>atba = trigsetup.TriggerOutputEnabled 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_TriggerOutputEnabled( BOOL *pVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_TriggerOutputEnabled( BOOL newVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ITriggerSetup2</td>
</tr>
</tbody>
</table>
TuningIFBW Property

**Description** Set the IF Bandwidth for Broadband and Precise tuning sweeps.

**VB Syntax**
```
embedLO.TuningIFBW = value
```

**Variable**

- **embedLO** An **EmbeddedLO** (object)
- **value** (Double) IF Bandwidth

**Return Type** (Double)

**Default** 30 kHz

**Examples**
```
embedLO.TuningIFBW = 10e3 'write

value = embedLO.TuningIFBW 'read
```

**C++ Syntax**
```
HRESULT get_TuningIFBW(double* ifbw);
HRESULT put_TuningIFBW(double ifbw);
```

**Interface** IEmbededLO

---

Last Modified:

18-Apr-2007  MX New topic
## TuningMode Property

**Description**
Sets and returns the method used to determine the embedded LO Frequency.

**VB Syntax**
```vbnet
embedLO.TuningMode = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>embedLO</code></td>
<td>An <code>EmbeddedLO</code> (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Enum as <code>NAEmbeddedLOTuningMode</code>)</td>
</tr>
</tbody>
</table>

Tuning mode. Choose from:
- 0 - `naEmbeddedLOTuningMode_Broadband_And_Precise`
- 1 - `naEmbeddedLOTuningMode_Precise_Only`
- 2 - `naEmbeddedLOTuningMode_None`

**Return Type**
(Enum)

**Default**
0 - `naEmbeddedLOTuningMode_Broadband_And_Precise`

**Examples**
```vbnet
embedLO.TuningMode = naEmbeddedLOTuningMode_None 'write
value = embedLO.TuningMode 'read
```

**C++ Syntax**
```cpp
HRESULT get_TuningMode(enum NAEmbeddedLOTuningMode* mode);
HRESULT put_TuningMode(enum NAEmbeddedLOTuningMode mode);
```

**Interface**
`IEmbededLO`

---

**Last Modified:**
13-Apr-2007   MX New topic
**Description**  
Set how often a tuning sweep is performed.

**VB Syntax**  
`embedLO.TuningSweepInterval = value`

**Variable (Type)**  
An `EmbeddedLO` (object)

- `embedLO` (Long) Tuning sweep interval.
- `value` (Long) Tuning sweep interval.

**Return Type**  
(Long)

**Default**  
1

**Examples**  
```vbnet
embedLO.TuningSweepInterval = 3 'write .. tuning is performed every third measurement sweep
value = embedLO.TuningSweepInterval 'read
```

**C++ Syntax**  
```cpp
HRESULT get_TuningSweepInterval(long* interval);
HRESULT put_TuningSweepInterval(long interval);
```

**Interface**  
IEembededLO
## TriggerMode Property

**Description**
These settings determine what EACH signal will trigger.

**Note:** Setting Point and EverySweep mode forces `Trigger.Scope = naChannelTrigger`.

### VB Syntax

```
chan.TriggerMode = value
```

### Variable (Type) - Description

<table>
<thead>
<tr>
<th>chan</th>
<th>A <code>Channel</code> (object)</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(enum NATriggerMode) - Choose from:</td>
</tr>
<tr>
<td>0 - naTriggerModePoint</td>
<td>Each Manual or External trigger signal causes one data point to be measured.</td>
</tr>
<tr>
<td>1 - naTriggerModeMeasurement</td>
<td>(superseded - still works but replaced with a more descriptive enum)</td>
</tr>
<tr>
<td>1 - naTriggerModeChannel</td>
<td>Each trigger signal causes <strong>ALL traces</strong> in that channel to be swept.</td>
</tr>
<tr>
<td>2 - naTriggerModeEverySweep</td>
<td>Each Manual or External trigger signal causes <strong>ALL traces that share a source port</strong> to be swept.</td>
</tr>
</tbody>
</table>

### Return Type
Long Integer

### Default
1 - naTriggerModeChannel

### Examples

```
chan.TriggerMode = naTriggerModePoint 'Write
```

```
trigtyp = chan.TriggerMode 'Read
```

### C++ Syntax

```
HRESULT get_TriggerMode (tagNATriggerMode *pMode)
HRESULT put_TriggerMode (tagNATriggerMode newMode)
```

### Interface
ICHannel

---

**Last Modified:**

6-Nov-2007  Added new sweep mode
### TriggerSignal Property - Superseded

**Description**

_Note: This command has been replaced by Source Property_

Sets or returns the trigger source.

**VB Syntax**

```vbs
app.TriggerSignal = value
```

**Variable**

**(Type) - Description**

- **app** An Application *(object)*
- **value** *(enum NATriggerSignal)* - Choose from:
  - 0 - naTriggerInternal - free run
  - 1 - naTriggerExternalPositive - a trigger signal is generated when a TTL high is sensed on the external trigger pin of the Aux IO connector
  - 2 - naTriggerExternalNegative - a trigger signal is generated when a TTL low is sensed on the external trigger pin of the Aux IO connector.
  - 4 - naTriggerExternalHigh - a trigger signal is generated when a TTL high is sensed on the external trigger pin of the Aux IO connector
  - 5 - naTriggerExternalLow - a trigger signal is generated when a TTL low is sensed on the external trigger pin of the Aux IO connector.

**Return Type**

Long Integer

**Default**

`naTriggerInternal`

**Examples**

```vbs
app.TriggerSignal = naTriggerExternalPositive 'Write
```

```vbs
trigsign = app.TriggerSignal 'Read
```

**C++ Syntax**

```c++
HRESULT get_TriggerSignal(tagNATriggerSignal *pSignal)
HRESULT put_TriggerSignal(tagNATriggerSignal signal)
```

**Interface**

IApplication
### TriggerType Property - Superseded

**Description**

Note: This property has been replaced with **Scope** Property.

Sets or returns the trigger type which determines the scope of a trigger signal.

**VB Syntax**

```vbnet
app.TriggerType = value
```

**Variable (Type) - Description**

- **app** - An Application *(object)*

- **value** *(enum NATriggerType)* - Trigger type. Choose from:
  - 0 - **naGlobalTrigger** - a trigger signal is applied to all triggerable channels
  - 1 - **naChannelTrigger** - a trigger signal is applied to the current channel. The next trigger signal will be applied to the next channel; not necessarily channel 1-2-3-4.

**Return Type**

Long Integer

**Default**

naGlobalTrigger

**Examples**

```vbnet
app.TriggerType = naGlobalTrigger 'Write
trigtyp = app.TriggerType 'Read
```

**C++ Syntax**

```c++
HRESULT get_TriggerType(tagNATriggerType *pTrigger)
HRESULT put_TriggerType(tagNATriggerType trigger)
```

**Interface**

IAplication
### Type (calstd) Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets and Returns the type of calibration standard.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calstd.Type = value</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>calstd</code></td>
<td>A CalStandard <em>(object)</em>. Use <code>calKit.GetCalStandard</code> to get a handle to the standard.</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(enum NACalStandardType)</em> - Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naOpen</td>
</tr>
<tr>
<td></td>
<td>1 - naShort</td>
</tr>
<tr>
<td></td>
<td>2 - naLoad</td>
</tr>
<tr>
<td></td>
<td>3 - naThru</td>
</tr>
<tr>
<td>Return Type</td>
<td>Long Integer</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>calstd.Type = naOpen 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>standardtype = calstd.Type 'Read</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT get_Type(tagNACalStandardType *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_Type(tagNACalStandardType newVal)</td>
</tr>
<tr>
<td>Interface</td>
<td>ICalStandard</td>
</tr>
</tbody>
</table>
**Read-only**

**About External Testset Control**

**Type (testset) Property**

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Returns the testset model number.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>tset.Type model</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>model</code></td>
<td>(String) Variable to return the Test set model</td>
</tr>
<tr>
<td><code>tset</code></td>
<td>A <code>TestsetControl</code> object.</td>
</tr>
<tr>
<td></td>
<td>Obtained from the <code>ExternalTestsets</code> collection.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>None</td>
</tr>
</tbody>
</table>

**Examples**

```
testset.type model
```

*See External Testset Program*

**C++ Syntax**

```
HRESULT get_Type(BSTR *pType);
```

**Interface**

`ITestsetControl`
### TZImag Property

**Description**
Sets and Returns the TZImag value (the Imaginary Terminal Impedance value) for the calibration standard. Only applicable when "Type" is set to `naArbitraryImpedance`.

To set the other resistance values, use `TZReal`.

**VB Syntax**
```vbnet
calstd.TZImag = value
```

**Variable** *(Type)* - Description

`calstd` A CalStandard *(object)*. Use `calKit.GetCalStandard` to get a handle to the standard.

`value` *(single)* - Value for TZImag in Ohms

**Return Type**
Single

**Default**
Not Applicable

**Examples**
```vbnet
calstd.TZImag = 15 'Write the value of TZImag to 15 Ohms
imp0 = calstd.TZImag 'Read the value of TZImag
```

**C++ Syntax**
```c++
HRESULT get_TZImag(float *pVal);
HRESULT put_TZImag(float newVal);
```

**Interface**
ICalStandard2
### TZReal Property

**Description**
Sets and Returns the TZReal value (the real Terminal Impedance value) for the calibration standard. Only applicable when "Type" is set to `naArbitraryImpedance`.

To set the other resistance values, use `TZImag`.

**VB Syntax**
```vbnet
calstd.TZReal = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calstd</code></td>
<td>A CalStandard <em>(object)</em>. Use <code>calKit.GetCalStandard</code> to get a handle to the standard.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Value for TZReal in Ohms</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**
Single

**Default**
Not Applicable

**Examples**
```vbnet
calstd.TZReal = 15 'Write the value of TZReal to 15 Ohms
imp0 = calstd.TZReal 'Read the value of TZReal
```

**C++ Syntax**
```cpp
HRESULT get_TZReal(float *pVal);
HRESULT put_TZReal(float newVal);
```

**Interface**
`ICalStandard2`
**UnusedChannelNumbers Property**

**Description**
Returns an array of channel numbers that are NOT in use. An unused channel has NO measurements subscribed to it.

**VB Syntax**
```
chanNumbers = chans.UnusedChannelNumbers(NumberOfChannels)
```

**Variable**
<table>
<thead>
<tr>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chanNumbers</td>
</tr>
<tr>
<td>chans</td>
</tr>
</tbody>
</table>

**NumberOfChannels**
(Long Integer) Number of channels that you are requesting.

**Return Type**
One-dimensional array of long integers. The size of the array is specified by the NumberOfChannels parameter.

**Default**
Not Applicable

**Examples**
```
chanNumbers = chans.UnusedChannelNumbers(5)
```

**C++ Syntax**
```
HRESULT get_UnusedChannelNumbers(long numberRequested,VARIANT* channelNumbers);
```

**Interface**
IChannels2
### USBPowerMeterCatalog Property

**Description**
Returns the ID string of power meters / sensors that are connected to the PNA USB. Use the list to select a power sensor for a source power cal.

**VB Syntax**

```vbnet
list = pwrCal.USBPowerMeterCatalog
```

**Variable**

- **list** *(String)* Variable to store the returned list of USB power meters.
- **pwrCal** *(object)* – A `SourcePowerCalibrator` object

**Return Type**
Comma-delimited strings

**Default**
Not Applicable

**Examples**

```vbnet
Set pwrCal = pna.SourcePowerCalibrator
list = pwrCal.USBPowerMeterCatalog'
```

**C++ Syntax**

```cpp
HRESULT get_USBPowerMeterCatalog(BSTR *pUSBList);
```

**Interface**

`ISourcePowerCalibrator6`

---

**Last Modified:**

11-Jul-2007  MX New topic
**Read/Write**

**UseCalWindow Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Turns Calibration window ON or OFF during a calibration. Learn more about this preference.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>$\text{guidedCal.UseCalWindow} = \text{value}$</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>$(\text{Type})$ - <strong>Description</strong></td>
</tr>
<tr>
<td>$\text{guidedCal}$</td>
<td>GuidedCalibration (object)</td>
</tr>
<tr>
<td>$\text{value}$</td>
<td>$(\text{Boolean})$</td>
</tr>
<tr>
<td>$\text{True}$</td>
<td>Show calibration window</td>
</tr>
<tr>
<td>$\text{False}$</td>
<td>Hide calibration window</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Boolean</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>True</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>$\text{guided.UseCalWindow} = \text{True}$</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_UseCalWindow(VARIANT_BOOL* val);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_UseCalWindow(VARIANT_BOOL newVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IGuidedCalibration</td>
</tr>
</tbody>
</table>

Last Modified: 12-Sep-2007  MX New topic
Read-only

**UsedChannelNumbers Property**

**Description**
Returns an array of channel numbers that are in use. A used channel has at least one measurement subscribed to it.

**VB Syntax**

```vbnet
chanNumbers = chans.UsedChannelNumbers
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chanNumbers</td>
<td>Variable array to store the returned channel numbers</td>
</tr>
<tr>
<td>chans</td>
<td>A Channel collection (object)</td>
</tr>
</tbody>
</table>

**Return Type**
One-dimensional array of long integers

**Default**
Not Applicable

**Examples**

```vbnet
chanNumbers = chans.UsedChannelNumbers
```

**C++ Syntax**

```cpp
HRESULT get_UsedChannelNumbers(VARIANT* channelNumbers);
```

**Interface**
IChannels2
UsePowerLossSegments Property

**Description**
Specifies if subsequent calls to the [AcquirePowerReadings](#) method will make use of the loss table (PowerLossSegments).

**VB Syntax**
```
pwrCal.UsePowerLossSegments = value
```

**Variable (Type) - Description**
- `pwrCal` *(object)* – A SourcePowerCalibrator (object)
- `value` *(boolean)*
  - `False` – Do not use loss table
  - `True` – Use loss table

**Return Type**
Boolean

**Default**
False

**Examples**
```
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.UsePowerLossSegments = 1 'Write
lossTableState = powerCalibrator.UsePowerLossSegments 'Read
```

**C++ Syntax**
```cpp
HRESULT put_UsePowerLossSegments(VARIANT_BOOL bState);
HRESULT get_UsePowerLossSegments(VARIANT_BOOL *bState);
```

**Interface**
ISourcePowerCalibrator
**UsePowerSensorFrequencyLimits Property**

**Description**
Specifies if subsequent calls to the `AcquirePowerReadings` method will observe frequency values of the `MinimumFrequency` and `MaximumFrequency` properties.

**VB Syntax**
```
pwrCal.UsePowerSensorFrequencyLimits = value
```

**Variable (Type) - Description**

- `pwrCal` *(object)* – A `SourcePowerCalibrator` (object)
- `value` *(boolean)* -
  - **False** – Do not use power sensor frequency limits. An acquisition will use just one power sensor for the entire sweep, regardless of frequency.
  - **True** – Use power sensor frequency limits. A requested acquisition will only succeed for those frequency points which fall between the `MinimumFrequency` and `MaximumFrequency` values of that `PowerSensor`. An acquisition will pause in mid-sweep if the frequency is about to exceed the `MaximumFrequency` value. When the sweep is paused in this manner, a sensor connected to the other channel input of the power meter can be connected to the measurement port in place of the previous sensor, and then the sweep completed by another call to `AcquirePowerReadings`. However, the `MaximumFrequency` specified for the second sensor would need to be sufficient for the sweep to complete.

**Return Type**
Boolean

**Default**
False

**Examples**
```
Set powerCalibrator = pna.SourcePowerCalibrator
powerCalibrator.UsePowerSensorFrequencyLimits = True
Write
FreqCheck = powerCalibrator.UsePowerSensorFrequencyLimits
Read
```

**C++ Syntax**
```
HRESULT put_UsePowerSensorFrequencyLimits(VARIANT_BOOL bState);
HRESULT get_UsePowerSensorFrequencyLimits(VARIANT_BOOL *bState);
```

**Interface**
`ISourcePowerCalibrator`
UserRange Property

Description
Assigns the marker to the specified User Range. This restricts the marker's x-axis travel to the User Range span, specified with Start and Stop values.

- Each channel has 16 user ranges.
- Markers and trace statistics can be restricted to any user range.
- More than one marker can occupy a user range.
- User ranges can overlap. For example:
  - User range 1 - 3GHz to 5GHz
  - User range 2 - 4GHz to 6GHz

Note: User ranges are especially useful in restricting marker searches to specific areas of the measurement.

VB Syntax

```
mark.UserRange = value
```

Variable (Type) - Description

- **mark** (A Marker (object))
- **value** (long integer) - User Range. Choose any number between 0 and 16 (0=Full Span)

Return Type
Long Integer

Default
0 - Full Span

Examples
```
mark.UserRange = 1 'Write
UseRange = mark.UserRange 'Read
```

C++ Syntax

- HRESULT get_UserRange(long *pRangeNumber)
- HRESULT put_UserRange(long lRangeNumber)

Interface
IMarker
**UserRangeMax Property**

**Description**
Sets the stimulus stop value for the specified User Range. This property uses different arguments for the channel and marker objects.

**VB Syntax**
```vbnet
chan.UserRangeMax(domainType,Mnum) = value
or
mark.UserRangeMax(mnum) = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chan</td>
<td>A Channel (object)</td>
</tr>
<tr>
<td>mark</td>
<td>A Marker (object)</td>
</tr>
</tbody>
</table>

To assign a marker to a User Range, use the [UserRange Property](#).

**Note:** The Marker object does not require the "domainType" argument.

**domainType**

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - naDomainFrequency</td>
<td></td>
</tr>
<tr>
<td>1 - naDomainTime</td>
<td></td>
</tr>
<tr>
<td>2 - naDomainPower</td>
<td></td>
</tr>
</tbody>
</table>

**Mnum**

<table>
<thead>
<tr>
<th>Long Integer</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Range number. Choose any number between 1 and 16 (0=Full Span)</td>
<td></td>
</tr>
</tbody>
</table>

**value**

<table>
<thead>
<tr>
<th>Double</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop value. Choose any number within the full span of the channel</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**

Double

**Default**
The current stimulus setting for the channel

**Examples**

```vbnet
mark.UserRangeMax(1) = 3e9 'Write
chan.UserRangeMax(naDomainFrequency,1) = 3e9 'Write
```

```csharp
UserRngeMax = mark.UserRangeMax 'Read
UserRngeMax = chan.UserRangeMax 'Read
```

**C++ Syntax**

```c
HRESULT put_UserRangeMax(tagNADomainType domain, long rangeNumber, double maxValue)
HRESULT get_UserRangeMax(tagNADomainType domain, long rangeNumber, double *maxValue)
```

**Interface**

IChannel
### UserRangeMin Property

**Description**
Sets the stimulus start value for the specified User Range. This property uses different arguments for the channel and marker objects.

**VB Syntax**

```vbnet
chan.UserRangeMin(domainType, range) = value
or
mark.UserRangeMin(range) = value
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chan</td>
<td>A Channel (object)</td>
<td></td>
</tr>
<tr>
<td>mark</td>
<td>A Marker (object)</td>
<td></td>
</tr>
</tbody>
</table>

To assign a marker to a User Range, use the `UserRange Property`.

**Note:** The Marker object does not require the `domainType` argument.

<table>
<thead>
<tr>
<th>domainType</th>
<th>(enum NADomainType)</th>
<th>Type of sweep currently implemented on the channel - Choose from:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>naDomainFrequency</td>
<td>0 - naDomainFrequency</td>
</tr>
<tr>
<td>1</td>
<td>naDomainTime</td>
<td>1 - naDomainTime</td>
</tr>
<tr>
<td>2</td>
<td>naDomainPower</td>
<td>2 - naDomainPower</td>
</tr>
</tbody>
</table>

| range      | (long) - User Range number. Choose any number between 1 and 16 (0=Full Span) |
| value      | (double) - Start value. Choose any number within the full span of the analyzer |

**Return Type**
Double

**Default**
The current stimulus setting for the channel

**Examples**

```vbnet
mark.UserRangeMin(1) = 3e9 'Write
chan.UserRangeMin(naDomainFrequency, 1) = 3e9 'Write
```

```plaintext
UseRngeMin = mark.UserRangeMin 'Read
UseRngeMin = chan.UserRangeMin 'Read
```

**C++ Syntax**

```cpp
HRESULT put_UserRangeMin(tagNADomainType domain, long rangeNumber, double minValue)
HRESULT get_UserRangeMin(tagNADomainType domain, long rangeNumber, double *minValue)
```

**Interface**
IChannel
UserPresetEnable Property

Description
'Checks' and 'clears' the enable box on the User Preset dialog box. This only affects subsequent Presets from the front panel user interface.

Regardless of the state of the User Preset Enable checkbox, the app.Preset command will always preset the PNA to the factory preset settings, and app.UserPreset will always perform a User Preset.

VB Syntax
app.UserPresetEnable = state

Variable (Type) - Description

app An Application (object)

state (boolean) Front Panel User Preset State. Choose from:
False – User Preset OFF
True – User Preset ON

Return Type Boolean

Default False

Examples
app.UserPresetEnable = True 'Write
upreset = app.UserPresetEnable 'Read

C++ Syntax
HRESULT get_UserPresetEnable(VARIANT_BOOL *pVal)
HRESULT put_UserPresetEnable(VARIANT_BOOL newVal)

Interface IApplication6
Valid Property

**Description**
Returns a boolean value to indicate if the current equation on the measurement is valid. For equation processing to occur, the equation must be valid and ON.

**VB Syntax**
IsValid = eq.Valid

**Variable**
*(Type) - Description*

*good* *(Boolean)* Variable to store the returned value.

- **True** (1) - equation is valid
- **False** (0) - equation is NOT valid

*eq* *(Equation object)*

**Return Type**
Boolean

**Default**
Not Applicable

**Examples**
IsValid = eq.Valid 'Read

**C++ Syntax**
HRESULT get_Valid(Boolean *equation)

**Interface**
IEquation

---

Last Modified:

18-Jun-2007  New topic
ValidConnectorTypes Property

**Description**
Returns a list of connector types for which there are calibration kits.

**VB Syntax**
```
value = obj.ValidConnectorTypes
```

**Variable** *(Type)* - Description

- **value** *(Variant)* List of connector types
- **obj** Any of the following:
  - `GuidedCalibration` (object)
  - `SMCType` (object)
  - `VMCType` (object)

**Return Type**
Variant

**Default**
Not Applicable

**Examples**
```
value = SMC.ValidConnectorTypes
```

**C++ Syntax**
```
HRESULT get_ValidConnectorTypes(VARIANT* connectorTypes);
```

**Interface**
`IGuidedCalibration`
- `SMCType`
- `VMCType`
## Value Property

**Description**  
Write or read a value (setting) for the current element.  
See a list of configurable elements and settings for various PNA models.

**VB Syntax**  
`pathElement.Value = value`

**Variable**  
(Type) - Description

- **pathElement**  
  A `PathElement` (object)

- **value**  
  (String) Value for the element. Use `pathElement.Values` to return a list of valid settings for this element.

**Return Type**  
String

**Default**  
Not Applicable

**Examples**  
- Set the "Combiner" element to value "Reversed"
  
  ```
  chan.PathConfiguration.Element("Combiner").Value = "Reversed"
  ```

  ```
  setting=pathElement.Value
  ```

**C++ Syntax**  
HRESULT get_Value( BSTR* pValue );
HRESULT put_Value( BSTR value );

**Interface**  
IPathElement

---

Last Modified:

14-Dec-2006   MX New topic
## Values Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Returns an array of valid settings that can be used with the element object. See a list of configurable elements and settings for various PNA models.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>values = pathElement.Values</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>values</code></td>
<td>(Variant array) Variable to store the array of valid settings for the element.</td>
</tr>
<tr>
<td><code>pathElement</code></td>
<td>A <code>PathElement</code> (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant array</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>settings=pathElement.Values</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Values(VARIANT* pValues);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IPathElement</td>
</tr>
</tbody>
</table>

Last Modified:

- 14-Dec-2006  MX New topic
### VelocityFactor Property

**Description**
Sets the velocity factor to be used with Electrical Delay, Port Extensions, and Time Domain marker distance calculations.

**VB Syntax**
```
app.VelocityFactor = value
```

**Variable**

<table>
<thead>
<tr>
<th><strong>(Type)</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><em>app</em> An <em>Application</em> <em>(object)</em></td>
<td></td>
</tr>
</tbody>
</table>

| **(double)** | Velocity factor. Choose a number between: 0 and 10 (.66 polyethylene dielectric; .7 teflon dielectric) |

**Return Type**
Double

**Default**
1

**Examples**
```
app.VelocityFactor = .66 'Write
RelVel = app.VelocityFactor 'Read
```

**C++ Syntax**
```
HRESULT get_VelocityFactor(double *pVal)
HRESULT put_VelocityFactor(double newVal)
```

**Interface**
IApplication
## View Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets (or returns) the type of trace displayed on the screen.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.View = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A measurement (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(enum NAView) - Type of trace. Choose from:</td>
</tr>
<tr>
<td>0 - naData</td>
<td></td>
</tr>
<tr>
<td>1 - naDataAndMemory</td>
<td></td>
</tr>
<tr>
<td>2 - naMemory</td>
<td></td>
</tr>
<tr>
<td>3 - naNoTrace</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The naData trace may reflect the result of a TraceMath operation.

<table>
<thead>
<tr>
<th>Return Type</th>
<th>NAView</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>naData</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.View = naData 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>trceview = meas.View 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_View(tagNAView* pView)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_View(tagNAView newView)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
Write/Read  
Visible Property

Description  Makes the Network Analyzer application visible or not visible. In the Not Visible state, the analyzer cycle time for making measurements can be significantly faster because the display does not process data.

VB Syntax  
app.Visible = state

Variable  
app  (Type) - Description

app  An Application (object)

state  (boolean)

False  - Network Analyzer application NOT visible
True  - Network Analyzer application IS visible

Return Type  Boolean

Default  True

Examples  
app.Visible = False 'Write

vis = app.Visible 'Read

C++ Syntax  
HRESULT get_Visible(VARIANT_BOOL * bVisible)
HRESULT put_Visible(VARIANT_BOOL bVisible)

Interface  IApplication
**WGCutoffFreq Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the value of the waveguide cut off frequency.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.WGCutoffFreq = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) - Frequency in Hertz.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>Print <code>meas.WGCutoffFreq</code> 'prints the value of the waveguide cut off frequency'</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_WGCutoffFreq(double *pVal);</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_WGCutoffFreq(double newVal);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement2</td>
</tr>
</tbody>
</table>
## Width Property

**Description**  
Sets the pulse width - the amount of time that the pulse is ON.

**VB Syntax**  
`pulse.Width (n) = value`

**Variable**  
**Type** - Description

- **pulse**  
  A `PulseGenerator` **object**

- **n**  
  **Integer**  
  Pulse generator number. Choose from 0 to 4.  
  0 is the generator that pulses the ADC.

- **value**  
  **Double**  
  Pulse width in seconds. Choose a value from about 33ns to about 70 seconds.

**Return Type**  
Double

**Default**  
1e-4 sec

**Examples**  
```
pulse.Width = 1ms 'Write
value = pulse.Width 'Read
```

**C++ Syntax**  
```
HRESULT get_Width (integer pulse, double* width);
HRESULT put_Width (integer pulse, double width);
```

**Interface**  
`IPulseGenerator`

---

Last Modified:  
2-Jan-2007  
MX New topic
WindowNumber Property

**Description**
Returns the window number. You might use this property to identify a particular window so that you can create a new Measurement in that window.

**VB Syntax**

```
value = win.WindowNumber
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>win</code></td>
<td>A NAWindow (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(long integer) - Variable to store the returned window number</td>
</tr>
</tbody>
</table>

**Return Type**

Long Integer

**Default**

Not Applicable

**Examples**

```
value = app.ActiveNAWindow.WindowNumber
```

**C++ Syntax**

```
HRESULT (long* windowNumber);
```

**Interface**

INAWindow
About Arranging Windows

WindowState Property

Description
Sets or returns the window setting of Maximized, Minimized, or Normal. To arrange all of the windows, use `app.ArrangeWindows`.

VB Syntax
`object.WindowState = value`

Variable (Type) - Description

- **object**  
  - An `Application (object)` - main window  
  - An `NaWindow (object)` - data windows

- **value**  
  - `(enum NAWindowStates)` - The window state. Choose from:  
    - `0 - naMinimized` - Minimizes the window to an Icon on the lower toolbar  
    - `1 - naMaximized` - Maximizes the window  
    - `2 - naNormal` - changes the window size to the user defined setting (between Max and Min).

Return Type
Long Integer

Default
naMaximized

Examples

- `app.WindowState = naMinimized` 'changes the Network Analyzer application window to an icon. -Write
- `win.WindowState = naNormal` 'changes the window defined by the win object variable to user defined settings. -Write

C++ Syntax

- `HRESULT get_WindowState(tagNAWindowStates *pVal)`
- `HRESULT put_WindowState(tagNAWindowStates newVal)`

Interface
`INAWindow`  
`IApplication`
## XAxisAnnotation Property

**Description**

Returns the X-Axis annotation of the specified tuning sweep.

**VB Syntax**

```vbnet
value = embedLODiag.XAxisAnnotation (n)
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String</td>
<td>Variable to store the returned data.</td>
</tr>
</tbody>
</table>

**embedLODiag**

An `EmbeddedLODiagnostic` object.

**n**

(Long) Tuning sweep number. Use `NumberOfSweeps` to find the number of sweeps taken.

**Default**

Not Applicable

**Examples**

```vbnet
data= embedLO.XAxisAnnotation 3 'read
```

**C++ Syntax**

```cpp
HRESULT XAxisAnnotation (long sweep, BSTR* annotation);
```

**Interface**

`IEembedLODiagnostic`

---

Last Modified:

13-Apr-2007  MX New topic
**XAxisPointSpacing Property**

**Description**
Sets X-axis Point Spacing for the displaytraces measured with segment sweeps on the active channel.

**VB Syntax**
```
chan.XAxisPointSpacing = value
```

**Variable**
*(Type)* - Description

- `chan` A Channel *(object)*

- `value` *(Enum as naStates)* - Choose from:
  - 0 - `naOFF` - Turns X-axis Point Spacing OFF
  - 1 - `naON` - Turns X-axis Point Spacing ON

**Return Type**
Enum

**Default**
0 - `naOFF`

**Examples**
```
chan.XAxisPointSpacing = naOFF 'Write

xspac = chan.XAxisPointSpacing 'Read
```

**C++ Syntax**
```
HRESULT get_XAxisPointSpacing (tagNAStates *pState);
HRESULT put_XAxisPointSpacing (tagNAStates newState);
```

**Interface**
IChannel2
### XAxisStart Property

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Returns the X-Axis start value of the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = embedLODiag.XAxisStart (n)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td><em>(Double)</em> Variable to store the returned data.</td>
</tr>
<tr>
<td><code>embedLODiag</code></td>
<td>An EmbeddedLODiagnostic <em>(object)</em></td>
</tr>
<tr>
<td><code>n</code></td>
<td><em>(Long)</em> Tuning sweep number. Use NumberOfSweeps to find the number of sweeps taken.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>data = embedLO.XAxisStart 3 'read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT XAxisStart (long sweep, double* start);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEmbeddedLODiagnostic</td>
</tr>
</tbody>
</table>

**Last Modified:**

13-Apr-2007  MX New topic
**Read-only**

**XAxisStop Property**

**Description**
Returns the X-Axis stop value of the specified tuning sweep.

**VB Syntax**
```
value = embedLODiag.XAxisStop (n)
```

**Variable**
- **(Type)**: Description
- **value** (Double) Variable to store the returned data.
- **embedLODiag** An EmbeddedLODiagnostic (object)

- **n** (Long) Tuning sweep number. Use **NumberOfSweeps** to find the number of sweeps taken.

**Default**
Not Applicable

**Examples**
```
data= embedLO.XAxisStop 3 'read
```

**C++ Syntax**
```
HRESULT XAxisStop (long sweep, double* start);
```

**Interface**
IEmbeddedLODiagnostic

---

Last Modified:

13-Apr-2007  MX New topic
**YAxisAnnotation Property**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the Y-Axis annotation of the specified tuning sweep.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>( value = embedLODiag.YAxisAnnotation(n) )</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td>value</td>
<td><strong>(String) Variable to store the returned data.</strong></td>
</tr>
<tr>
<td>embedLODiag</td>
<td>An <strong>EmbeddedLODiagnostic (object)</strong></td>
</tr>
<tr>
<td>( n )</td>
<td>(Long) Tuning sweep number. Use <strong>NumberOfSweeps</strong> to find the number of sweeps taken.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
| **Examples** | data = embedLO.YAxisAnnotation 3  
| **C++ Syntax** | HRESULT YAxisAnnotation (long sweep, BSTR* annotation); |
| **Interface** | IEmbeddedLODiagnostic |

Last Modified:  
13-Apr-2007  MX New topic
### YScale Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets or returns the Y-axis Per-Division value of the active trace.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>trace.YScale = value</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>trace</code></td>
<td>A Trace (object)</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(double) - Scale /division number. Units and range depend on the current data format.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>10 (db)</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>trac.YScale = 5 'Write</code></td>
</tr>
<tr>
<td></td>
<td><code>yscl = trac.YScale 'Read</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT get_YScale(double *pVal)</td>
</tr>
<tr>
<td></td>
<td>HRESULT put_YScale(double newVal)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ITrace</td>
</tr>
</tbody>
</table>
### Z0 Property

**Description**
Sets and Returns the characteristic impedance for the calibration standard.

**VB Syntax**
```vbnet
calstd.Z0 = value
```

**Variable**
(Type) - Description
- `value` (single) - Impedance in Ohms

**Return Type**
Single

**Default**
Not Applicable

**Examples**
- ```vbnet
calstd.Z0 = 50 'Write
```
- ```vbnet
impedance = calstd.Z0 'Read
```

**C++ Syntax**
- `HRESULT get_Z0(float *pVal)`
- `HRESULT put_Z0(float newVal)`

**Interface**
ICalStandard
Abort Method

**Description**

Ends the current measurement sweep on the channel.

**VB Syntax**

`chan.Abort [sync]`

**Variable**

**(Type) - Description**

- `chan (object)` - A Channel object
- `sync (boolean)` - wait (or not) for the analyzer to stop before processing subsequent commands. Optional argument; if unspecified, value is set to False. Choose from:
  - **True** - synchronize - the analyzer will not process subsequent commands until the current measurement is aborted.
  - **False** - continue processing commands immediately

**Return Type**

None

**Default**

None

**Examples**

```
chan.abort True
chan.abort
```

**C++ Syntax**

`HRESULT Abort(VARIANT_BOOL bSynchronize);`

**Interface**

IChannel
## AbortPowerAcquisition Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Aborts a source power cal acquisition sweep that is currently in progress.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>powerCalibrator.AbortPowerAcquisition</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>powerCalibrator</code></td>
<td><em>(object)</em> - A <code>SourcePowerCalibrator</code> object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>powerCalibrator.AbortPowerAcquisition</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT AbortPowerAcquisition();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>ISourcePowerCalibrator</code></td>
</tr>
</tbody>
</table>
AcquireCalStandard Method - Superseded

**Description**

**Note:** This command has been replaced by AcquireCalStandard2 Method, which provides for acquisition of sliding load standards. All other functionality is identical.

**VB Syntax**

```
cal.AcquireCalStandard std[,index]
```

**Variable**

*Variable (Type) - Description*

- `cal` A Calibrator *(object)*

- `std` *(enum NACalClass)* Standard to be measured. Choose from:
  1. `naClassA`
  2. `naClassB`
  3. `naClassC`
  4. `naClassD`
  5. `naClassE`
  6. `naReferenceRatioLine`
  7. `naReferenceRatioThru`

**SOLT Standards**

- `naSOLT_Open`
- `naSOLT_Short`
- `naSOLT_Load`
- `naSOLT_Thru`
- `naSOLT_Isolation`

**TRL Standards**

- `naTRL_Reflection`
- `naTRL_Line_Reflection`
- `naTRL_Line_Tracking`
- `naTRL_Thru`
- `naTRL_Isolation`

*index* *(long integer)* number of the standard. Optional argument - Used if there is more than one standard required to cover the necessary frequency range. If unspecified, value is set to 1.
Note  The behavior has changed with PNA revisions as follows:

- Before 6.01: Accepted 0 and changed it to 1
- 6.01 to 6.04: Did NOT accept 0
- 6.04.11 and higher: Accepts 0 and changes it to 1

<table>
<thead>
<tr>
<th>Return Type</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

Examples  Cal.AcquireCalStandard naSOLT_Thru Write

C++ Syntax  HRESULT AcquireCalStandard(tagNACalClass enumClass, short standardNumber)

Interface  ICalibrator

Last modified:

10/05/06  Modified Index argument.
AcquireCalStandard2 Method

**Description**
Measures the specified standard from the selected calibration kit. The calibration kit is selected using `app.CalKitType`.

For 2-port calibration, it is also necessary to specify direction with `AcquisitionDirection`.

To omit Isolation from a 2-port calibration, do not Acquire a cal standard for `naSOLT_Isolation`.

For using two sets of standards, see `Simultaneous2PortAcquisition Property`.

**Note:** This command replaces `AcquireCalStandard`. This command provides for the acquisition of a sliding load cal. All other functionality is identical.

**VB Syntax**
```vbnet
cal.AcquireCalStandard2 std[,index][,slide]
```

**Variable (Type) - Description**

- **cal**
  A `Calibrator` (object)

- **std**
  (enum NACalClass) Standard to be measured. Choose from:

  1 - naClassA
  2 - naClassB
  3 - naClassC
  4 - naClassD
  5 - naClassE
  6 - naReferenceRatioLine
  7 - naReferenceRatioThru

**SOLT Standards**

1 - naSOLT_Open
2 - naSOLT_Short
3 - naSOLT_Load
4 - naSOLT_Thru
5 - naSOLT_Isolation

**TRL Standards**

1 - naTRL_Reflection
2 - naTRL_Line_Reflection
3 - naTRL_Line_Tracking
4 - naTRL_Thru
5 - naTRL_Isolation

[index]  (long integer) Number of the standard. Optional argument - Used if there is more than one standard required to cover the necessary frequency range. If unspecified, value is set to 1.

Note  The behavior has changed with PNA revisions as follows:

- Before 6.01: Accepted 0 and changed it to 1
- 6.01 to 6.04: Did NOT accept 0
- 6.04.11 and higher: Accepts 0 and changes it to 1

[slide]  (enum as NACalStandardSlidingState) Optional argument. State of the sliding load. The slide should be set a minimum of five times. Seven is the maximum that can be stored. Choose from:

0 - naNotSlidingStd - not using a sliding load - Default if not specified.
1 - naSlidesSet - slide is set for acquisition
2 - naSlidesDone - this next acquisition will be the last. Calculations will then be performed.

Return Type  None

Default  Not Applicable

Examples
Cal.AcquireCalStandard2 naSOLT_Thru
Cal.AcquireCalStandard2 naSOLT_Thru,2,naNotSlidingStd

'measures the second standard listed in the class of naSOLT_Thru'

C++ Syntax  HRESULT AcquireCalStandard2(NACalClass enumClass, long standardPosition, NACalStandardSlidingState slidingStandardState)

Interface  ICalibrator

Last modified:
10/05/06  Modified Index argument.
AcquireCalConfidenceCheckECALEx Method

Description

This method replaces `AcquireCalConfidenceCheckECAL`

Transfers confidence data from the specified ECal module into the measurement's memory trace. The data is transferred to the specified S-parameter on the same channel as this Calibrator object.

The characterization within the ECal module that the confidence data will be read from is specified by `ECALCharacterizationEx`. The default value is 0.

VB Syntax

`cal.AcquireCalConfidenceCheckECALEx Sparam [,ecalModule]`

Variable (Type) - Description

- `cal` A Calibrator (object)
- `Sparam` (String) S-parameter to transfer confidence data to. This parameter must be present on the same channel as the calibrator object.
- `ecalModule` (Integer) – Optional argument. ECal module. Choose from modules 1 through 8
  
  Use `IsECALModuleFoundEx` to determine the number of modules connected to the PNA
  
  Use `GetECALModuleInfoEx` to return the model and serial number of each module.

Return Type

None

Default

Not applicable

Examples

```
Cal.AcquireCalConfidenceCheckECALEx "S11", 2
```

C++ Syntax

`HRESULT AcquireCalConfidenceCheckECALEx(BSTR strParameter, long moduleNumber = 1);`

Interface

ICalibrator4
**AcquirePowerReadingsEx Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>This command replaces <strong>AcquirePowerReadings Method</strong>. Initiates a source power cal acquisition.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>powerCalibrator.AcquirePowerReadingsEx calMethod, acqdevice [,sync]</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>powerCalibrator</code></td>
<td><em>(object)</em> - A <strong>SourcePowerCalibrator</strong> object</td>
</tr>
<tr>
<td><code>calMethod</code></td>
<td><em>(enum NASourcePowerCalMethod)</em> Selects the calibration method to be used for the source power cal acquisition.</td>
</tr>
<tr>
<td>0 – <code>naPowerMeter</code></td>
<td>Use power meter for all readings.</td>
</tr>
<tr>
<td>1 - <code>naPowerMeterAndReceiver</code></td>
<td>Power meter for the first iteration; then use the reference receiver for remaining readings if necessary.</td>
</tr>
<tr>
<td>2 - <code>naReceiver</code></td>
<td>Use PNA measurement receiver for all readings.</td>
</tr>
<tr>
<td><code>acqdevice</code></td>
<td><em>(String)</em> The specific acquisition device to be used. NOT case sensitive. Choose from: If <code>calMethod</code> = <code>naPowerMeter</code> or <code>naPowerMeterAndReceiver</code>, choose from:</td>
</tr>
<tr>
<td></td>
<td>“ASEN” -- Sensor on power meter channel A.</td>
</tr>
<tr>
<td></td>
<td>“BSEN” -- Sensor on power meter channel B.</td>
</tr>
<tr>
<td></td>
<td>To use the sensor that currently corresponds to the frequency of interest, use the value from the <strong>PowerAcquisitionDevice</strong> property.</td>
</tr>
<tr>
<td></td>
<td>If <code>calMethod</code> = <code>naReceiver</code>, choose from the receiver names for your specific PNA using either physical receiver notation or <strong>logical receiver notation</strong>. For example, &quot;a1&quot; or &quot;A&quot;.</td>
</tr>
<tr>
<td><code>[sync]</code></td>
<td><em>(boolean)</em> Optional argument. If not specified, value is set to False. Choose from:</td>
</tr>
<tr>
<td>True (1)</td>
<td>The method does not return until this acquisition has completed (the program calling this method is halted while waiting for the method to return).</td>
</tr>
<tr>
<td>False (0)</td>
<td>The method initiates an acquisition then returns immediately (while the acquisition still proceeds). The program calling this method can then perform other operations during the acquisition.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>powerCalibrator.AcquirePowerReadingsEx naPowerMeter, &quot;asen&quot;, True</code></td>
</tr>
<tr>
<td></td>
<td><code>powerCalibrator.AcquirePowerReadingsEx naReceiver, &quot;b2&quot;</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT AcquirePowerReadingsEx (tagNASourcePowerCalMethod enumCalMethod, BSTR bstrAcqDevice, VARIANT_BOOL bSync);</code></td>
</tr>
</tbody>
</table>
Last modified:

9/12/06  MQ New command to accommodate receiver only SPC
## AcquireStep Method

### Description
Acquire the measurement data for the specified step in the calibration process.

**Note:** Guided Cal allows you to measure standards in any order. See an example.

### VB Syntax

```vbnet
obj.AcquireStep (n)
```

### Variable *(Type) - Description*

- **obj**
  - Any of the following:
    - `GuidedCalibration` (object)
    - `SMCType` (object)
    - `VMCType` (object)

- **n**
  - Step number in the calibration process.
  - Use `GenerateSteps` to determine the total number of steps.
  - Use `GetStepDescription` to read the description of each step.

### Return Type
Not Applicable

### Default
Not Applicable

### Examples

```vbnet
VMC.AcquireStep (3)
```

### C++ Syntax

```cpp
HRESULT put_AcquireStep([in] long step);
```

### Interface

- `SMCType`
- `VMCType`
- `IGuidedCalibration`

---

**Last Modified:**

- 20-Jan-2007  Added any order note.
### Activate Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Makes an object the Active Object. When making a measurement active, the channel and window the measurement is contained in becomes the active channel and active window. In order to change properties on any of the active objects, you must first have a &quot;handle&quot; to the active object using the <strong>Set</strong> command. For more information, See <a href="#">Getting a Handle to an Object</a>. You do not have to make an object &quot;Active&quot; to set or read its properties remotely. But an object must be &quot;Active&quot; to change its values from the front panel.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>VB Syntax</th>
<th><code>object.Activate</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td>object</td>
<td><code>Measurement (object)</code> or <code>Marker (object)</code></td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>meas.Activate</code> <code>mark.Activate</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT Activate()</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>IMeasurement</code> <code>IMarker</code></td>
</tr>
</tbody>
</table>
### About Markers

**ActivateMarker Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Makes a marker the Active Marker. Use <code>meas.ActiveMarker</code> to read the number of the active marker.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.ActivateMarker(Mnum)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>meas</code></td>
<td>A <code>Measurement</code> (object)</td>
</tr>
<tr>
<td><code>Mnum</code></td>
<td>(long integer) - the number of the marker to make active. Choose any marker number from 1 to 9.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.ActivateMarker(1)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT ActivateMarker(long lMarkerNumber)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IMeasurement</code></td>
</tr>
<tr>
<td><strong>Remarks</strong></td>
<td>Use <code>ReferenceMarkerState</code> to control the Reference marker.</td>
</tr>
</tbody>
</table>
ActivateWindow Method

Description
Makes a window object the Active Window.
In order to change properties on any of the active objects, you must first have a "handle" to the active object using the Set command. For more information, See Programming the Analyzer Object Model.
You do not have to make an object "Active" to set or read its properties remotely. But an object must be "Active" to change its values from the front panel.

VB Syntax
app.ActivateWindow n

Variable (Type) - Description
app An Application (object)
n (long) Number of the window to make active

Return Type Window Object
Default Not Applicable

Examples
app.ActivateWindow 4

C++ Syntax HRESULT ActivateWindow(long WindowNumber)

See the PNA Object Model
### Add (channels) Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Creates a channel and returns a handle to it. If the channel already exists, it returns the handle to the existing channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><em>chans.Add (item)</em></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>chans</code></td>
<td>A Channel collection (object)</td>
</tr>
<tr>
<td><code>item</code></td>
<td>(variant) - Channel number</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Channel</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chans.Add 3 ‘Creates channel 3</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Add(VARIANT numVal, IChannel** pChannel)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannels</td>
</tr>
</tbody>
</table>
## Write-only

### Add (measurement) Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Adds a Measurement to the collection.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.Add channel, param, source[, window]</code></td>
</tr>
<tr>
<td>meas</td>
<td>A Measurements collection (object)</td>
</tr>
<tr>
<td>channel</td>
<td>(long) - Channel number of the new measurement.</td>
</tr>
<tr>
<td>param</td>
<td>(string) - New parameter. Case insensitive.</td>
</tr>
</tbody>
</table>

#### For S-parameters:
- Any S-parameter that can be measured by your PNA.
- Single-digit port numbers can be separated by "_" (underscore). For example: "S21" or "S2_1"
- Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

#### For Ratioed measurements:
- Any two receivers in your PNA separated by "/". For example: "A/R1"
- See the block diagram showing the receivers in YOUR PNA.

#### For Unratioed (absolute power) measurements:
- Any receiver in the PNA. For example: "A"
- See the block diagram showing the receivers in YOUR PNA

With PNA Rev 6.2, Ratioed and Unratioed measurements can also use logical receiver notation to refer to receivers. This notation makes it easy to refer to receivers with an external test set connected to the PNA. You do not need to know which physical receiver is used for each test port. Learn more.

#### For ADC measurements
- Any ADC receiver in the PNA followed by a comma, then the source port.
  - For example: "AI1_2" indicates the Analog Input1 with source port of 2.
  - Learn more about ADC receiver measurements.

#### For Balanced S-parameter measurements:
- "topology:Sabxy"
- **topology** - Choose from:
  - **sbal** - single-ended to balanced
  - **ssb** - single-ended / single-ended to balanced
  - **bbal** - balanced to balanced
$S_{abxy}$ - 

Where

- **a** - device output (receive) mode
- **b** - device input (source) mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

- **x** - device output (receive) logical port number
- **y** - device input (source) logical port number

For example: "$sbal:sdd42$"

See an example program

For **Imbalance** and **Common Mode Rejection** measurements:

"topology:parameter" Choose from:

<table>
<thead>
<tr>
<th>Choose this:</th>
<th>To get this:</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;SBAL:IMBSB&quot;</td>
<td>single-ended to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB1&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection  (Sds21/Scs21)</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB2&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection  (Ssd12/Ssc12)</td>
</tr>
<tr>
<td>&quot;SSB:IMB1SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 1</td>
</tr>
<tr>
<td>&quot;SSB:IMB2SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 2</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB1&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection  (Sds31/Scs31)</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB2&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection  (Sds32/Ssc32)</td>
</tr>
<tr>
<td>&quot;BBAL:IMB1BB&quot;</td>
<td>balanced to balanced</td>
<td>imbalance 1</td>
</tr>
<tr>
<td>BBAL:IMB2BB</td>
<td>balanced to balanced</td>
<td>imbalance 2</td>
</tr>
<tr>
<td>BBAL:CMRRBB</td>
<td>balanced to balanced</td>
<td>common mode rejection (Sdd21/Scc21)</td>
</tr>
</tbody>
</table>

source (long integer) - Source port number; if unspecified, value is set to 1. Only used for non-s-parameter measurements; ignored if s-parameter.

window (long integer) - Optional argument. Window number of the new measurement. If unspecified, the S-Parameter will be created in the Active Window. Choose between 1 and the maximum number of windows allowed on the PNA. If unspecified, the measurement will be created in the Active Window.

See also Traces, Channels, and Windows on the PNA.

Return Type None

Default None

Examples `meass.Add 3,"A/R1",1,1 'Adds A/R1 measurement to channel 3 in window 1`

C++ Syntax `HRESULT Add(long ChannelNum, BSTR strParameter, long srcPort, VARIANT_BOOL bNewWindow)`

Interface IMeasurements

Last modified:

9/12/06 MQ Added logical receiver notation and number of windows.
### Add (NAWindows) Method

**Description**  
Add a window to the display. Does not add a measurement. The window number must not already exist.

**VB Syntax**  
`wins.Add [item]`

**Variable (Type) - Description**

- `wins`  
  A [NAWindow](#) collection *(object)*

- `item` *(variant)* - optional argument; Window number. Choose between 1 and the maximum number of windows allowed on the PNA. If unspecified, the measurement will be created in the Active Window.

  See also [Traces, Channels, and Windows on the PNA](#).

**Return Type**  
Object

**Default**  
Not Applicable

**Examples**

```
wins.Add 3 'Creates a window number 3
```

**C++ Syntax**

```
HRESULT Add(long windowNumber )
```

**Interface**  
INAWindows

---

**Last modified:**

9/12/06  
Modified for number of windows
Add (PowerLossSegment) Method

**Description**
Adds a PowerLossSegment to the PowerLossSegments collection.
To ensure predictable results, it is best to remove all segments before defining a new list of segments. For each segment in the collection, do a seg.Remove.

**VB Syntax**
```
segs.Add (item [ size])
```

**Variable (Type) - Description**
- `segs` *(object)* - A PowerLossSegments collection (object)
- `item` *(variant)* - Number of the new segment. If it already exists, a new segment is inserted at the requested position.
- `size` *(long integer)* - Optional argument. The number of segments to add, starting with item. If unspecified, value is set to 1.

**Return Type**
None

**Default**
Not Applicable

**Examples**
```
segs.Add 1, 4 'Adds segments 1, 2, 3 and 4
```

**C++ Syntax**
```
HRESULT Add(VARIANT index, long size);
```

**Interface**
IPowerLossSegments
### Add (PowerSensorCalFactorSegment) Method

**Description**
Add a PowerSensorCalFactorSegment to the CalFactorSegments collection. To ensure predictable results, it is best to remove all segments before defining a new list of segments. For each segment in the collection, do a `seg.Remove`.

**VB Syntax**
`segs.Add (item [ size])`

**Variable**
- **segs** *(object)* - A `CalFactorSegments` collection (object)
- **item** *(variant)* - Number of the new segment. If it already exists, a new segment is inserted at the requested position.
- **size** *(long integer)* - Optional argument. The number of segments to add, starting with item. If unspecified, value is set to 1.

**Return Type**
None

**Default**
Not Applicable

**Examples**
`segs.Add 1, 4 'Adds segments 1, 2, 3 and 4`

**C++ Syntax**
`HRESULT Add(VARIANT index, long size);`

**Interface**
ICalFactorSegments
### Add (segment) Method

**Description**  
Adds segments to the Segments collection, but does not turn the segments ON.

**VB Syntax**  
`segs.Add (item, [size])`

- **segs**  
  A `segments` collection *(object)*

- **item** *(variant)*  
  Number of the new segment. If it already exists, a new segment is inserted at the requested position.

- **size** *(long integer)*  
  Optional argument. The number of segments to add, starting with `item`. If unspecified, value is set to 1.

**Return Type**  
None

**Default**  
None

**Examples**  
`Segs.Add 1, 4 'Adds segments 1,2,3, and 4. (does NOT automatically turn segments ON)`

**C++ Syntax**  
`HRESULT Add(VARIANT index, long size);`

**Interface**  
ISegments

**Remarks**  
To ensure predictable results, it is best to remove all segments before defining a segment list. For each segment in the collection, do a seg `Remove`. 

Write-only

About External Testset Control

Add (Testset) Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Adds a testset to the ExternalTestsets Collection and loads the configuration file.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>testsets.Add(model,address)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td>testsets</td>
<td>An <code>ExternalTestsets</code> (collection)</td>
</tr>
<tr>
<td>model</td>
<td><em>(String)</em> Model of the testset to be added, NOT case-sensitive.</td>
</tr>
<tr>
<td>address</td>
<td><em>(Integer)</em> Address of the testset to be added.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>testsets.Add(&quot;Z5623AK66&quot;,12)</code> ' add Z5623AK66 test at address 12 to testsets collection</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT Add(BSTR typename, long address)</td>
</tr>
<tr>
<td>Interface</td>
<td>IExternalTestsets</td>
</tr>
</tbody>
</table>

There is no COM command to read a list of currently-supported test sets. However, the following SCPI command can be used with the following format:

```
string = SCPIStringParser.Execute("SENSe:MULTiplexer:CATalog?")
```
### AllowAllEvents Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets event filtering to monitor all events in the analyzer. This is the default setting when subscribing to events. This could slow the measurement speed of the analyzer significantly.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.AllowAllEvents</code></td>
</tr>
<tr>
<td><strong>Variable (Type) - Description</strong></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.AllowAllEvents</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT AllowAllEvents()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
About Cal Window

**AllowChannelToSweepDuringCalAcquisition Method**

**Description**
Specifies the channel to sweep during a Calibration.
When this command is sent, the *SwpChan* channel is 'flagged' to be swept during calibration. The flag is cleared when the channel is deleted or if the Measurement Class is changed. If the same channel number is recreated, this command must be sent again to sweep the channel during a calibration. The flag is NOT saved with an instrument state.

A Preset or Instrument State Recall deletes the channel.

**VB Syntax**
```vbnet
calMgr.AllowChannelToSweepDuringCalAcquisition (CalChan, SwpChan, State)
```

**Variable**

- **calMgr** *(object)* - A *CalManager* object
- **CalChan** *(long)* - Channel to be calibrated.
- **SwpChan** *(long)* - The channel to sweep when waiting to measure a standard.
  This channel must already exist. If this channel is in continuous sweep mode, it must have the same attenuator settings and path configuration (PNA-X only).
- **state** *(Boolean)* - Channel sweep state. Choose from:
  - **True** - Sweep the channel during calibration.
  - **False** - Do NOT sweep the channel during calibration.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```
calMgr.AllowChannelToSweepDuringCalAcquisition 2,1,True
```
See example using this command

**C++ Syntax**
```
HRESULT AllowChannelToSweepDuringCalAcquisition ( long CalChannel, long SwpChannel, VARIANT_BOOL bVal);
```

**Interface**
ICalManager5

Last Modified:
8-Nov-2007   MX New topic
### AllowEventCategory Method

**Description**  
Sets event filtering to monitor a category of event.

**VB Syntax**  
`app.AllowEventCategory, category, state`

**Variable**  
**(Type)** - **Description**

- `app`  
  An Application (object)

- `category`  
  Category to monitor. Choose from list in [Working with the Analyzer's Events](#)

- `state`  
  (boolean)
  - `True` - monitor
  - `False` - do not monitor

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`app.AllowEventCategory`

**C++ Syntax**  
`HRESULT AllowEventCategory(tagNAEventCategory category, VARIANT_BOOL bAllow )`

**Interface**  
IAppllication
### AllowEventMessage Method

**Description**
Sets event filtering to monitor specific events.

**VB Syntax**
```vbnet
app.AllowEventMessage event
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th><strong>app</strong></th>
<th>An Application (object)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>event</strong></td>
<td>Event to monitor. Refer to list in <a href="#">Working with the Analyzer’s Events</a></td>
</tr>
<tr>
<td><strong>state</strong></td>
<td>(boolean)</td>
</tr>
<tr>
<td><strong>True</strong></td>
<td>- monitor</td>
</tr>
<tr>
<td><strong>False</strong></td>
<td>- do not monitor</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
app.AllowEventMessage
```

**C++ Syntax**
```c++
HRESULT AllowEventMessage( tagNAEventID eventID, VARIANT_BOOL bAllow)
```

**Interface**
IAplication
### AllowEventSeverity Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets event filtering to monitor levels of severity.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.AllowEventSeverity severity, state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>app</code></td>
<td>An <a href="object">Application</a></td>
</tr>
<tr>
<td><code>severity</code></td>
<td>(enum [naEventSeverity](error, informational, success, warning)) Choose from: naEventSeverityERROR, naEventSeverityINFORMATIONAL, naEventSeveritySUCCESS, naEventSeverityWARNING</td>
</tr>
<tr>
<td><code>state</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> - monitor</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> - do not monitor</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><a href="#">app.AllowEventSeverity</a></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><a href="#">HRESULT AllowEventSeverity</a>(tagNAEventSeverity severity, VARIANT_BOOL bAllow)</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><a href="#">IApplication</a></td>
</tr>
</tbody>
</table>
Apply Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Applies the mixer setup to the mixer object and turns the channel ON. (Performs the same function as the Apply button on the mixer setup dialog box.)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mxr.Apply</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>mxr</code></td>
<td>Mixer Interface pointer to the Measurement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mxr.Apply</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Apply()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMixer3</td>
</tr>
</tbody>
</table>
ApplyDeltaMatchFromCalSet Method

Description
Specifies a Cal Set as a source of delta match correction.
If 'GUID' is not supplied then the Global Delta Match Cal Set is assumed. An error is returned if the specified Cal Set does not meet the following Delta Match criteria. The Global Delta Match Cal can ALWAYS be applied.

- Must have been performed using ECal or as a guided mechanical cal (not Unguided).
- Must have the same start freq, stop freq, and number of points as the channel being calibrated.
- Must calibrate the ports that are required by the TRL or Unknown Thru Cal as indicated by PortsNeedingDeltaMatch Property.

Learn more about Delta match calibration.
See example of a complete Delta Match calibration.

VB Syntax
`guided.ApplyDeltaMatchFromCalSet [GUID]`

Variable (Type) - Description
- **guidied** GuidedCalibration (object)
- **GUID** Optional Argument. GUID of the Cal Set to use. If unspecified, the Global Delta Match Cal Set is used.

Return Type
Not Applicable

Default
Not Applicable

Examples
`guided.ApplyDeltaMatchFromCalSet "{2B893E7A-971A-11d5-8D6C-00108334AE96}"

C++ Syntax
`HRESULT ApplyDeltaMatchFromCalSet(BSTR calsetGUID);`

Interface
IGuidedCalibration2
# ApplyPowerCorrectionValuesEx Method

**Description**

This command replaces `ApplyPowerCorrectionValues Method`. Applies the array of power correction values to the channel memory and turns correction ON. Perform after completing a source power cal acquisition sweep.

This command does NOT save the correction values. To save correction values, save an instrument / calibration state (*cst file) after performing a source power cal.

Optionally, as part of the source power calibration, perform calibration of the reference receiver used in the power calibration. Learn more.

**VB Syntax**

```
powerCalibrator.ApplyPowerCorrectionValuesEX [rRec]
```

**Variable (Type) - Description**

- `powerCalibrator` **(object)** - A `SourcePowerCalibrator` object
- `rRec` **(Enum as NASourcePowerApplyCorrectionOption) Optional argument. Choose from:**
  - `0` - `naSourcePowerApplyCorrectionDefault` Do NOT perform and save a calibration of the reference receiver. (Default if not specified).
  - `1` - `naIncludeReferenceReceiverPowerCal` Perform and save a calibration of the reference receiver. The Cal Set, which includes only the reference receiver cal, is saved to the destination specified by `RemoteCalStoragePreference`.

**Return Type**

None

**Default**

Not Applicable

**Examples**

```
powerCalibrator.ApplyPowerCorrectionValuesEX
powerCalibrator.ApplyPowerCorrectionValuesEX
  (naIncludeReferenceReceiverPowerCal)
```

**C++ Syntax**

```
HRESULT ApplyPowerCorrectionValuesEx(enum
  NASourcePowerApplyCorrectionOption option);
```

**Interface**

`ISourcePowerCalibrator5`

---

Last Modified:

23-Apr-2007   MX New topic
About Source Power Calibration

ApplySourcePowerCorrectionTo Method

Description
Copies and applies an existing Source Power Calibration to another channel.

VB Syntax
`chan.ApplySourcePowerCorrectionTo(fromPortNum, targetChan, targetPortNum);`

Variable (Type) - Description
- `chan`: A `Channel` (object)
- `fromPortNum`: (Long) Port number of the existing source power correction.
- `targetChan`: (Long) Channel number to which the source power correction will be copied.
- `targetPortNum`: (Long) Port number to which the source power correction will be applied.

Return Type
Not Applicable

Default
Not Applicable

Examples
`chan.ApplySourcePowerCorrectionTo 1,2,1`

C++ Syntax
`HRESULT ApplySourcePowerCalibrationTo(long fromPortNumber, long otherChannelNumber, long portNumber);`

Interface
IChannel11

Last Modified:
20-Jul-2007  MX New topic
AutoPortExtMeasure Method

Description  Measures either an OPEN or SHORT standard. When this command is sent, the PNA acquires the measurement with which to set automatic port extensions. Learn more about choosing which standard to measure.

VB Syntax  

```vbnet
fixture.AutoPortExtMeasure value
```

Variable  (Type) - Description

- **fixture** A **Fixturing (object)**
- **value** (Enum as NAAutoPortExtMeasure)
  - 0 - naAPEM.OPEN - Measure OPEN
  - 1 - naAPEM.SHORT - Measure SHORT

Return Type  ENUM

Default  Not Applicable

Examples  

```vbnet
fixture.AutoPortExtMeasure naAPEM_OPEN
```

C++ Syntax  HRESULT get_AutoPortExtMeasure(tagNAAutoPortExtMeasure *pVal );

Interface  IFixturing2
### AutoPortExtReset Method

**Description**  
Clears old port extension delay and loss data in preparation for acquiring new data. Send this command prior to sending a new series of measurements using `AutoPortExtMeasure Method`. If acquiring both OPEN and SHORT standards, do not send this command between those acquisitions.

**VB Syntax**  
`fixture.AutoPortExtReset`

**Variable**  
*(Type)* - Description  
- `fixture` A Fixturing *(object)*

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`fixture.AutoPortExtReset`

**C++ Syntax**  
`HRESULT AutoPortExtReset();`

**Interface**  
`IFxturing2`
## Autoscale Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Trace Object - Autoscales only the <strong>ONE</strong> trace on which Autoscale is being called. NAWindow Object - Autoscales <strong>ALL</strong> of the traces in the Window on which Autoscale is being called.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>object.Autoscale</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - <strong>Description</strong></td>
</tr>
</tbody>
</table>
| `object`          | **Trace** *(object)*  
|                   | or  
|                   | **NAWindow** *(object)*                                                                                                                                                                           |
| **Return Type**   | Not Applicable                                                                                                                                                                                   |
| **Default**       | Not Applicable                                                                                                                                                                                   |
| **Examples**      | Trac.Autoscale  'Autoscales the trace  
|                   | Win.Autoscale  'Autoscales all the traces in the window -Write                                                                                                                                 |
| **C++ Syntax**    | HRESULT AutoScale()                                                                                                                                                                               |
| **Interface**     | INAWindow  
|                   | ITrace                                                                                                                                                                                          |
### AveragingRestart Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Clears and restarts averaging of the measurement data.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>chan.AveragingRestart</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><code>chan</code></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>chan.AveragingRestart</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT AveragingRestart()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel</td>
</tr>
</tbody>
</table>
**BuildHybridKit Method**

**Description**
Use this method when you have different port connectors. This is a convenient way to combine two kits that match the connectors on your DUT.

**VB Syntax**
```vbnet
app.BuildHybridKit port1Kit,p1sex,port2Kit,p2sex,adapter,user kit
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>app</strong></td>
<td>An Application (object)</td>
<td></td>
</tr>
<tr>
<td><strong>port1Kit</strong></td>
<td>(enum NACalKit)</td>
<td>Specifies the two kits to be used to build the hybrid kit. Choose from:</td>
</tr>
<tr>
<td><strong>port2Kit</strong></td>
<td>(enum NACalKit)</td>
<td></td>
</tr>
<tr>
<td><strong>p1sex</strong></td>
<td>(enum NAPortSex)</td>
<td>Specifies the sex of the connector at that port. Choose from:</td>
</tr>
<tr>
<td><strong>p2sex</strong></td>
<td>(enum NAPortSex)</td>
<td></td>
</tr>
<tr>
<td><strong>adapter</strong></td>
<td>(enum NAAdapter)</td>
<td>Choose from:</td>
</tr>
<tr>
<td><strong>userKit</strong></td>
<td>(enum NACalKit)</td>
<td>The Hybrid kit - Choose from the previous list of kits</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
app.BuildHybridKit
naCalKit_85033E_3_5,naMale,naCalKit_85038A_7_16,naFemale,naUserkit,naCalKit_User8
```

**C++ Syntax**
```cpp
HRESULT BuildHybridKit(tagNACalKit port1Kit, tagNAPortSex port1Sex, tagNACalKit port2Kit, tagNAPortSex port2Sex, tagNAAdapter adapter, tagNACalKit userKit)
```

**Interface**
IApplication
Write-only

About Performing a Calibration

CalculateErrorCoefficients Method

Description
This method is the final call in a calibration process. It calculates error-correction terms, turns error-correction ON and saves the error-correction terms to the channel's Cal Register or a User Cal Set.

Do NOT use this command during an ECAL.

Note: The destination (Cal Register or User Cal Set) is determined by the setting of the RemoteCalStoragePreference property.

VB Syntax

```vbnet
cal.CalculateErrorCoefficients
```

Variable (Type) - Description

- `cal` Calibrator (object)

Return Type
Not Applicable

Default
Not Applicable

Examples

```
Cal.CalculateErrorCoefficients
```

C++ Syntax

```cpp
HRESULT CalculateErrorCoefficients()
```

Interface
ICalibrator

Last Modified:

16-Apr-2007   MX Added link to Remote...
## Calculate Method

**Description**
Calculates the Input or Output frequencies of the mixer setup, applies the mixer setup to the mixer object, and turns the channel ON.

**VB Syntax**
`mrx.Calculate (port)`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mrx</code></td>
<td>Mixer Interface pointer to the Measurement (object)</td>
</tr>
<tr>
<td><code>port</code></td>
<td>(enum as MixerCalculation) Port of the mixer for which to calculate start and stop frequencies. Choose from:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>enum</th>
<th>1st or only stage requires:</th>
<th>In addition, 2nd stage requires:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>mixCalculateINPUT</td>
<td>Output Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LO frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>IF sideband (High or Low)</td>
</tr>
<tr>
<td>1</td>
<td>mixCalculateINPUT AndOUTPUT</td>
<td>NA</td>
</tr>
<tr>
<td>2</td>
<td>mixCalculateOUTPUT</td>
<td>Input Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LO frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>IF sideband (High or Low)</td>
</tr>
<tr>
<td>3</td>
<td>mixCalculateLO1</td>
<td>Input Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output sideband (High or Low)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>IF sideband (High or Low)</td>
</tr>
<tr>
<td>4</td>
<td>mixCalculateLO2</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Input Start and stop frequencies</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1st LO start and stop frequencies</td>
</tr>
</tbody>
</table>
**frequencies**

- Output frequency
- IF sideband (High or Low)
- Output sideband (High or Low)

<table>
<thead>
<tr>
<th><strong>Return Type</strong></th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>mxr.Calculate (mixCalculateOUTPUT)</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT Calculate()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMixer</td>
</tr>
</tbody>
</table>
### ChangeParameter Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Changes the parameter of the measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><em>meas</em>.ChangeParameter(<em>param</em>,<em>src</em>)</td>
</tr>
<tr>
<td>Variable (Type) - Description</td>
<td></td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><em>param</em> (string)</td>
<td>- New parameter. Case insensitive.</td>
</tr>
</tbody>
</table>

**For S-parameters:**

- *Any S-parameter that can be measured by your PNA.*
- Single-digit port numbers can be separated by "_" (underscore). For example: "S21" or "S2_1"
- Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

**For Ratioed measurements:**

- Any two receivers in your PNA separated by "/". For example: "A/R1"
- See the [block diagram](#) showing the receivers in YOUR PNA.

**For Unratioed (absolute power) measurements:**

- Any receiver in the PNA. For example: "A"
- See the [block diagram](#) showing the receivers in YOUR PNA

With PNA Rev 6.2, **Ratioed** and **Unratioed** measurements can also use **logical receiver notation** to refer to receivers. This notation makes it easy to refer to receivers with an external test set connected to the PNA. You do not need to know which physical receiver is used for each test port. [Learn more](#).

**For ADC measurements**

- Any ADC receiver in the PNA.
- For example: "AI1" indicates the Analog Input1.
  - [Learn more about ADC receiver measurements](#).

**For Balanced S-parameter measurements:**

- "*topology*:Sabxy"
  - *topology* - Choose from:
  - `sbal` - single-ended to balanced
  - `ssb` - single-ended / single-ended to balanced
● **bbal** - balanced to balanced

**Sabxy** -

Where

- **a** - device output (receive) mode
- **b** - device input (source) mode

(choose from the following for both a and b:)

- **d** - differential
- **c** - common
- **s** - single ended

- **x** - device output (receive) logical port number
- **y** - device input (source) logical port number

For example: "**sbal:sdd42**"

See an example program

For **Imbalance** and **Common Mode Rejection** measurements:

"**topology:parameter**" Choose from:

<table>
<thead>
<tr>
<th>Choose this:</th>
<th>To get this:</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;SBAL:IMBSB&quot;</td>
<td>single-ended to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB1&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection (Sds21/Scs21)</td>
</tr>
<tr>
<td>&quot;SBAL:CMRRSB2&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection (Ssd12/Ssc12)</td>
</tr>
<tr>
<td>&quot;SSB:IMB1SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 1</td>
</tr>
<tr>
<td>&quot;SSB:IMB2SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 2</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB1&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection (Sds31/Scs31)</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB2&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection (Sds32/Scs32)</td>
</tr>
</tbody>
</table>
### ChangeParameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;BBAL:IMB1BB&quot;</td>
<td>balanced to balanced imbalance 1</td>
</tr>
<tr>
<td>&quot;BBAL:IMB2BB&quot;</td>
<td>balanced to balanced imbalance 2</td>
</tr>
<tr>
<td>&quot;BBAL:CMRRBB&quot;</td>
<td>balanced to balanced common mode rejection (Sdd21/Scc21)</td>
</tr>
</tbody>
</table>

**src**  
(long integer)
- Ignored if *param* is an S-Parameter
- Source port if *param* is a ratioed or unratioed measurement (including ADC measurements).

Use [GetPortNumber](#) to return the port number of a source that only has a string name, such as an External Source.

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**
- `meas.ChangeParameter "S11",2 '2 is the source port`
- `meas.ChangeParameter "A/R1",2 '2 is the source port`
- `meas.ChangeParameter "a1/b1",1 '1 is the source port`
- `meas.ChangeParameter "R1",2 '2 is the source port`

**C++ Syntax**  
HRESULT ChangeParameter(BSTR parameter, long IPort)

**Interface**  
IMeasurement

---

**Last Modified:**  
30-Apr-2007 Modified for ADC and src strings
## CheckPower Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Measures power at a specified frequency. Use this method to test power level before and/or after applying a source power calibration.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>pow = pwrCal.CheckPower(device, freq [, unit])</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>pow</code></td>
<td>(double) Variable to store power value returned by this method.</td>
</tr>
<tr>
<td><code>pwrCal</code></td>
<td>A <code>SourcePowerCalibrator</code> (object)</td>
</tr>
<tr>
<td><code>device</code></td>
<td>(enum <code>NAPowerAcquisitionDevice</code>) The specific sensor on the power meter to be used for the acquisition. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 – <code>naPowerSensor_A</code></td>
</tr>
<tr>
<td></td>
<td>1 – <code>naPowerSensor_B</code></td>
</tr>
<tr>
<td></td>
<td>To use the sensor that currently corresponds to the frequency of interest, use the value from the <code>PowerAcquisitionDevice</code> property.</td>
</tr>
<tr>
<td><code>freq</code></td>
<td>(double) Frequency (Hz) at which the sensor is to read the power.</td>
</tr>
<tr>
<td><code>unit</code></td>
<td>(enum <code>NAPowerUnit</code>)</td>
</tr>
<tr>
<td></td>
<td>Optional argument. Choose from:</td>
</tr>
<tr>
<td></td>
<td><code>naDBM</code> – Returns the power in dBm. (default)</td>
</tr>
<tr>
<td></td>
<td><code>naWATT</code> – Returns the power in Watts.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Double</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>watt = powerCalibrator.CheckPower(naPowerSensor_A, 1E9, naWATT)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_CheckPower(tagNAPowerAcquisitionDevice enumAcqDevice, double dFreq, tagNAPowerUnit enumPowerUnit, double *pdPower);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ISourcePowerCalibrator2</td>
</tr>
</tbody>
</table>
Clear Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Clears the current diagnostic information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>embedLODiag.Clear</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>embedLODiag</td>
<td>An EmbeddedLODiagnostic (object)</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td>embedLO.Clear 'write</td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT Clear();</td>
</tr>
<tr>
<td>Interface</td>
<td>IEmbeddedLODiagnostic</td>
</tr>
</tbody>
</table>

Last Modified:
12-Apr-2007   MX New topic
**Description**  
This command is no longer necessary. The CalSet.get... and put... commands that required this command have been replaced.

Closes read/write access to the Cal Set.

See [OpenCalSet](#) for an explanation of gaining access to the Cal Set.

When you are finished reading and writing data from or to the Cal Set, close the Cal Set. Subsequent read/writes will require a new OpenCal Set call.

Reading and writing Cal Set data is performed with the PutStandard, GetStandard, PutErrorTerm, GetErrorTerm method calls. These methods are provided by the ICal Set and ICalData2 interfaces.

**VB Syntax**  
`CalSet.CloseCalSet`

**Variable**  
**Type** - Description

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalSet</td>
<td>object</td>
<td>A Cal Set object</td>
</tr>
</tbody>
</table>

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`CSet.CloseCalSet`

**C++ Syntax**  
`HRESULT CloseCalSet`

**Interface**  
ICalSet
ComputeErrorTerms Method

Description

Computes error terms for the caltype specified by a preceding OpenCal Set call. The Cal Set must first be opened using OpenCalSet. If this call has not been made, the following error is issued:

E_NA_Cal Set_ACCESS_DENIED

The standards data required for the CalType must be available in the Cal Set or this error will be returned: E_NA_STANDARD_NOT_FOUND.

Note: Error term computation requires data for the actual calibration kit standards from the current kit definition. ComputeErrorTerms assumes that the standards were acquired using only one standard per class.

VB Syntax

CalSet.ComputeErrorTerms

Variable

CalSet (object) - A Cal Set object

Return Type

Not Applicable

Default

Not Applicable

Examples

CalSet.ComputeErrorTerms

C++ Syntax

HRESULT ComputeErrorTerms( )

Interface

ICalSet
**Description**

**Note:** This command replaces ConfigEnhancedNB Method.

This subroutine determines, then returns, the proper configuration for pulsed measurements on the **PNA-X ONLY** using the spectral nulling technique. The configuration returned needs to be sent to the PNA and any other related external equipment.

The routine will take a desired Pulse Repetition Frequency (PRF) and measurement IFBW and return a possibly modified PRF and IFBW for proper pulsed operation on the PNA.

**VB Syntax**

```vbnet
Pulsed.ConfigEnhancedNB2 (PRF, BW, PhysicalIF, NCO, ClockFreq, Stage1TapArray, Stage2TapArray, Stage3TapArray, FixedPRF, GateDelay, GateWidth, SWGateDelay, SWGateWidth, SWGateRamp)
```

**Variable** *(Type) - Description*

- **Pulsed** *(interface)*  
  An interface to the agilentpnapulsed.dll application interface.

- **PRF** *(Double)*  
  The Pulse Repetition Frequency.
  
  **[out]** The pulse repetition frequency that has been optimized for use with the PNA.
  
  **NOTE:** This value may be different from the value requested.

  **[in]** The desired pulse repetition frequency.

- **BW** *(Long)*  
  The PNA IF Bandwidth.

  **[out]** The PNA IF bandwidth that has been optimized for use with the PNA. **NOTE:** This value may be different from the value requested. Zero (0) is returned if no solution is found for the specified **PRF** and **BW**.

  **[in]** The desired PNA IF bandwidth.

- **PhysicalIF** *(Double)*  
  Returns physical intermediate frequency.

- **NCO** *(Double)*  
  Returns numeric controlled oscillator frequency.

- **ClockFreq** *(Double)*  
  Returns the clock frequency (in Hz) of the PNA-X.

- **Stage1TapArray** *(Long array)*  
  Returns the stage 1 filter coefficients

- **Stage2TapArray** *(Long array)*  
  Returns the stage 2 filter coefficients
Stage3TapArray  (Long array)
[out] Returns the stage 3 filter coefficients

FixedPRF  (Boolean)
[in]
  • 1 (True) Signals the .DLL routine to NOT adjust the PRF value; rather adjust
    ONLY the IF Bandwidth. This is the default setting.
  • 0 (False) Adjust both the PRF and IF Bandwidth values as necessary.

GateDelay  (Double)
[in] Highest delay value in seconds used in any of the receiver gates.

GateWidth  (Double)
[in] Widest pulse width value in seconds used in any of the receiver gates.

SWGateDelay  (Double)
[out] Returns the SW gate delay in seconds.

SWGateWidth  (Double)
[out] Returns the SW Gate width in seconds.

SWGateRamp  (Long)
[out] Returns the SW Gate ramp

Return Type  Not Applicable
Default  Not Applicable
Example  See an example using this command.

C++ Syntax  HRESULT ConfigEnhancedNB2(double *pPRF, double *pBW, double *pIF, double *pNCO, double *clock, double *Stg1, double *Stg2, double *Stg3, VARIANT_BOOL fixPRF, double gateDelay, double gateWidth, double SWgateDelay, double SWgateWidth, long SWgateRamp)

Interface  AgilentPNAPulsed.Application

Last Modified:
28-Mar-2007   MX New topic
### ConfigEnhancedNBIFAtten Method

**Description**
Sets PNA-X receivers to auto gain setting.

**VB Syntax**
```
Pulsed.ConfigEnhancedNBIFAtten (PRF, RxWidth, IFAtten)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulsed</td>
<td>(interface)</td>
<td>An interface to the agilentpnapulsed.dll application interface.</td>
</tr>
<tr>
<td>PRF</td>
<td>(Double)</td>
<td>[in] The Pulse Repetition Frequency.</td>
</tr>
<tr>
<td>RxWidth</td>
<td>(Double)</td>
<td>[in] Receiver gate width.</td>
</tr>
<tr>
<td>IFAtten</td>
<td>(Long Integer)</td>
<td>[out] IF attenuation value.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
See an example using this command.

**C++ Syntax**
```
HRESULT ConfigEnhancedNBIFAtten(double *pPRF, double *pWidth, long *pIF)
```

**Interface**
AgilentPNAPulsed.Application

---

Last Modified:
18-Jun-2007   MX New topic
### ConfigNarrowBand3 Method

**Description**

**Note:** This method replaces `ConfigNarrowBand2 Method`. The BW argument now returns 0 if no solution is found for the specified PRF and BW. In addition, adjustments were made to the filter finder algorithm.

This subroutine determines, then returns, the proper configuration for pulsed measurements on the PNA using the spectral nulling technique. The configuration returned needs to be sent to the PNA and any other related external equipment such as pulse generators. The routine will take a desired Pulse Repetition Frequency (PRF) and measurement IFBW and return a possibly modified PRF and IFBW for proper pulsed operation on the PNA. The routine will also return the Sample Rate, Number of Taps, and Offset that must be sent to the PNA to configure it in pulsed mode using the spectral nulling technique.

Although the example below uses COM programming to communicate with the PNA, these commands can be replaced with SCPI equivalents.

**Note:** The pulsed application may set the offset frequency (option 080) of the PNA to some value other than zero (the default value). If the stop frequency is set to the maximum of the PNA model, then an error message may appear on the PNA stating that the response frequency has exceeded the maximum allowed frequency. To fix this, set the stop frequency to a value that is at least 2 KHz less than the maximum allowed. For example, if you have a 20 GHz PNA, and the stop frequency is set to 20 GHz, and the error message appears, then set the stop frequency to 19.999998 GHz.

**VB Syntax**

```
Pulsed.ConfigNarrowBand (PRF, NumTaps, BW, OffSet, SampleRate, Precision, FixedPRF, PG81110)
```

**Variable (Type) - Description**

**Pulsed (interface)** An interface to the agilentpnapulsed.dll application interface.

**PRF (Double)** The Pulse Repetition Frequency.

- **[out]** The pulse repetition frequency that has been optimized for use with the PNA. NOTE: This value may be different from the value requested.
- **[in]** The desired pulse repetition frequency.

**NumTaps (Long)** The number of taps to send to the PNA for pulsed operation.

**BW (Long)** The PNA IF Bandwidth.

- **[out]** The PNA IF bandwidth that has been optimized for use with the PNA. NOTE: This value may be different from the value requested. Zero (0) is returned if no solution is found for the specified PRF and BW.
- **[in]** The desired PNA IF bandwidth.

**Offset (Double)** The offset value to send to the PNA for pulsed operation. The offset value is used to adjust the PNA for the two different possible sample rates that may be returned.
**SampleRate** *(Double)*

[out] The sample rate to send to the PNA for pulsed operation.

[in] Passing a value of 6.2 us will make sure that the offset frequency is not shifted and therefore could be used with converter measurements. Otherwise enter 0.

**Precision** *(Double)* The precision variables sets the precision that will be used to decrement the PRF when running the configuration routines. This variable can be set to the precision required by the external pulse generators so that the configuration routine will not return a PRF that is not within the precision limits of the pulse generators.

**FixedPRF** *(Boolean)*

1 (True) Signals the .DLL routine to NOT adjust the PRF value; rather adjust ONLY the IF Bandwidth. This is the default setting.

0 (False) Adjust both the PRF and IF Bandwidth values as necessary.

**PG81110** *(Boolean)*

1 (True) You are using an Agilent 81110 as the pulse generator. This allows increased accuracy in adjustments for offset and PRF.

0 (False) Not using an Agilent 8110.

**Return Type** Not Applicable

**Default** Not Applicable

**Example** See an example using this command.

**C++ Syntax**

```c++
HRESULT ConfigNarrowBand(double *pPRF, long *pNumTaps, long *pBW, double *pOffset, double *pSampleRate, int Precision)
```

**Interface** AgilentPNAPulsed.Application
### ConfigurationFile Method

**Description**
Recalls an Interface Control file from the hard drive into the analyzer.

**VB Syntax**
`IntControl.ConfigurationFile (filename)`

**Variable**
- **IntControl** *(Type)* - Description
- **filename** *(string)* - Full path, file name, and extension (.xml) of the file to recall.
  - Files are typically stored in "C:\Program Files\Agilent\Network Analyzer\Documents"

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
`IntControl.ConfigurationFile ("C:\Program Files\Agilent\Network Analyzer\Documents\MySettings.xml")`

**C++ Syntax**
`HRESULT ConfigurationFile(BSTR bstrFile)`

**Interface**
`IInterfaceControl`
### Configurations Property

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns an array of stored configuration names that can be used with <a href="#">DeleteConfiguration Method</a> and <a href="#">LoadConfiguration Method</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>names = pathMgr.Configurations</code></td>
</tr>
<tr>
<td>Variable</td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td></td>
<td><code>names</code> <em>(Variant array)</em> Variable to store the returned configuration names.</td>
</tr>
<tr>
<td><code>pathMgr</code></td>
<td><a href="#">PathConfigurationManager</a> <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Variant array</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>names = path.Configurations</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT get_Configurations(VARIANT* configurations);</code></td>
</tr>
<tr>
<td>Interface</td>
<td><a href="#">IPathConfigurationManager</a></td>
</tr>
</tbody>
</table>

Last Modified:

14-Dec-2006    MX New topic
## Configure Method

| Description | Restarts as an "N-port" PNA using the specified multiport test set.  
| See other commands to configure multiport test sets. |
| **VB Syntax** | `app.Configure (model, address)` |
| **Variable** | **(Type)** - Description |
| `app` | An Application (object) |
| `model` | String - Model of the test set with which to restart.  
Use "Native" to restart without a test set.  
To see a list of supported test sets, use |
| `address` | Integer - GPIB Address of the test set. Use 0 for native restart. |
| **Return Type** | Not Applicable |
| **Default** | Not Applicable |
| **Examples** | See an example using this command. |
| `app.Configure ("N44xx",18)` |
| **C++ Syntax** | HRESULT Configure(BSTR model, long address); |
| **Interface** | IApplication9 |
## Continuous Method

**Description**  The channel continuously responds to trigger signals.

**Note:** This command does **NOT** change `TriggerSignal` to Continuous.

**VB Syntax**  `chan.Continuous`

**Variable**  
- **(Type)** - Description
- `chan`  A `Channel` *(object)*

**Return Type**  Not Applicable

**Default**  Not Applicable

**Examples**  `chan.Continuous`

**C++ Syntax**  HRESULT Continuous()

**Interface**  IChannel
**About Cal Sets**

**Copy Method**

**Description**
Creates a new Cal Set and copies the current Cal Set data into it. Therefore, you now have a clone Cal Set with a different ID. Use this command to manipulate data on a Cal Set without corrupting the original cal data.

**VB Syntax**

```vbnet
CalSet.CopyTo()
```

**Variable**

`CalSet` - A `Cal Set` object

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
Dim mgr As CalManager
Dim ocalset As CalSet
Dim newcalset As CalSet
Set mgr = pna.GetCalManager
  'Create a new (empty) Cal Set.
Set ocalset = mgr.CreateCalSet(1)
ocalset.Description = "original calset"
pna.Channel(1).SelectCalSet ocalset.GetGUID, True
  'Launch the cal wizard and allow the user to perform the calibration.
  If pna.LaunchCalWizard(False) Then
    'If the Launch returns true then the calibration finished.
    ocalset.Save

    'Copy the Cal Set to the new one.
    Set newcalset = ocalset.CopyTo
    newcalset.Description = "copy of original calset"
  Else
    'If the cal doesn't finish, delete the old Cal Set
    'so it isn't taking up unnecessary memory.
    mgr.DeleteCalSet ocalset.GetGUID
End If
```

As a result, the programmer can manipulate the data in the new Cal Set and always revert back to the old Cal Set as needed.

**C++ Syntax**

```cpp
HRESULT Copy( ICalSet** pCalSet);
```

**Interface**

ICalSet
Write-only

CopyToChannel Method

**Description**  Sets up another channel as a copy of this object's channel.

**VB Syntax**  
```
chan.CopyToChannel(IChanNum)
```

**Variable**  

- **chan**  
  A Channel (object)

- **IChanNum**  
  (long integer) – Number of the channel to become a copy of this channel.

**Return Type**  None

**Default**  Not Applicable

**Examples**  
```
Dim chan As Channel
Set chan = PNAapp.ActiveChannel
Const INEW_CHAN_NUM As Long = 2
chan.CopyToChannel(INEW_CHAN_NUM)
```

**C++ Syntax**  
```
HRESULT CopyToChannel(long IChanNum);
```

**Interface**  IChannel2
### CreateSParameterEx Method

**Description**
Creates a new S-Parameter measurement in an existing or new window and specifies the load port for 3-port devices.

**VB Syntax**
```vbnet
app.CreateSParameterEx chan, recvr, source[, loadPort][, window]
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>Application (object)</td>
<td></td>
</tr>
<tr>
<td><code>chan</code></td>
<td>(long integer)</td>
<td>Channel number of the new measurement.</td>
</tr>
<tr>
<td><code>recvr</code></td>
<td>(long integer)</td>
<td>Port number of the test port receiver.</td>
</tr>
<tr>
<td><code>source</code></td>
<td>(long integer)</td>
<td>Port number of the source. Use <code>GetPortNumber</code> to return the port number of a source that only has a string name, such as an External Source. To learn more see Remotely Specifying a Source Port.</td>
</tr>
<tr>
<td><code>loadPort</code></td>
<td>(long integer)</td>
<td>Port number of the load. Required for reflection measurements of 3-port devices on multiport PNA models.</td>
</tr>
<tr>
<td><code>window</code></td>
<td>(long integer)</td>
<td>Optional argument. Choose between 1 and the maximum number of windows allowed on the PNA. See also Traces, Channels, and Windows on the PNA. If unspecified, the S-Parameter will be created in the Active Window.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
app.CreateSParameterEx 1,2,1,1 'Creates a new S21 measurement in channel 1 and New window(1)
app.CreateSParameterEx 2,1,1,3,1 'Creates a new S11 measurement on channel 2 with port 3 as the load. Create in the active window
```

**C++ Syntax**
```cpp
HRESULT CreateSParameterEx(long ChannelNum, long RcvPort, long SrcPort, long LoadPort, long windowNumber)
```

**Interface**
IApplication

---

Last modified:
9/12/06 Modified for number of windows
### CreateCalSet Method

**Description**

Creates a new Cal Set.

The new cal set is initialized with the stimulus settings from the channel whose number is passed as the argument to this method. Stimulus settings include frequency, bandwidth, number of points, and so forth.

Use this method when you want to manually upload data to the Cal Set using the returned ICal Set interface handle.

The channel number does not restrict the usage of this Cal Set on any other channel. It simply provides a link to the originating channel so that the stimulus values can be stored in the Cal Set.

**Note:** Be sure to SAVE the CalSet you are creating. Use `ICalSet::Save`.

#### VB Syntax

```vbnet
calMgr.CreateCalSet(chan)
```

#### Variable

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>calMgr</code></td>
<td>object - A <code>CalManager</code> object</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>long - channel number of the new Cal Set.</td>
</tr>
</tbody>
</table>

#### Return Type

ICal Set Interface

#### Default

Not Applicable

#### Example

```vbnet
calMgr.CreateCalSet 1
```

#### C++ Syntax

```cpp
HRESULT CreateCalSet( long ChannelNumber, ICalSet** pCalSet);
```

#### Interface

ICalManager
Write-only
CreateCustomCal Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Creates a custom cal object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calmgr.CreateCustomCal(CalType)</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>calMgr</code></td>
<td><strong>Cal Manager (Object)</strong></td>
</tr>
<tr>
<td><code>CalType</code></td>
<td><strong>(String) Name of the calibration. Choose from:</strong></td>
</tr>
<tr>
<td></td>
<td>&quot;VMC&quot; or &quot;VectorMixerCal.VMCTYPE&quot;</td>
</tr>
<tr>
<td></td>
<td>&quot;SMC&quot; or &quot;ScalarMixerCal.SMCType&quot;</td>
</tr>
<tr>
<td>See Also</td>
<td><strong>SMCTYPE Object</strong></td>
</tr>
<tr>
<td></td>
<td><strong>VMCTYPE Object</strong></td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**

```
Dim CalMgr As ICalManager2
Dim SMC As ISMCTYPE
Set SMC = CreateCustomCal("SMC")
```

See **SMC** and **VMC** examples using this command.

**C++ Syntax**

```
HRESULT CreateCustomCal( BSTR CustomCal)
```

**Interface**

ICalManager2
**CreateCustomCalEx Method**

**Description**
Create a custom calibration object for the specified channel.

*Note:* Use `CreateCustomCal_Method` to create FCA calibration objects.

**VB Syntax**
```
calmgr.CreateCustomCalEx (chan)
```

**Variable** *(Type) - Description*

- `calMgr` *Cal Manager (Object)*

- `chan` *(long integer)* Channel number in which to create the Cal object.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
dim GCA
set GCA = CalMgr.CreateCustomCalEx(1)
```

**See Also**
- [Noise Figure example](#)
- [Gain Compression example](#)

**C++ Syntax**
```
HRESULT CreateCustomCalEx(long  channel, IDispatch** ppObject);
```

**Interface**
ICalManager5

---

**Last Modified:**
29-May-2007   MN New topic
CreateCustomMeasurementEx Method

Description
Creates a new custom measurement.

VB Syntax
app.CreateCustomMeasurementEx chanNum,MeasClass,MeasName [,window]

Variable (Type) - Description

app (object) - An Application object

chanNum (long) - Channel number used by the new measurement; can exist or be a new channel.

MeasClass (string) - Measurement class of the new custom measurement object. The new custom measurement must be installed and registered on the PNA.
Choose from the following:

- "Vector Mixer/Converter"
- "Scalar Mixer/Converter"
- "Gain Compression"
- "Noise Figure Cold Source"

MeasName (variant) Measurement names

<table>
<thead>
<tr>
<th>MeasClass</th>
<th>Measurement Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector Mixer/Converter</td>
<td>&quot;S11&quot;</td>
<td>Learn about VMC parameters</td>
</tr>
<tr>
<td></td>
<td>&quot;VC21&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;S22&quot;</td>
<td></td>
</tr>
<tr>
<td>Scalar Mixer/Converter</td>
<td>&quot;S11&quot;</td>
<td>Learn about SMC parameters</td>
</tr>
<tr>
<td></td>
<td>&quot;SC21&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;SC12&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;S22&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Ipwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;RevIpwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;Opwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;RevOpwr&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&quot;ComplIn21&quot;</td>
<td>Input power at the compression point.</td>
</tr>
</tbody>
</table>
### Gain Compression

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CompOut21</strong></td>
<td>Output power at the compression point.</td>
</tr>
<tr>
<td><strong>CompGain21</strong></td>
<td>Gain at the compression point.</td>
</tr>
<tr>
<td><strong>CompS11</strong></td>
<td>Input Match at the compression point.</td>
</tr>
<tr>
<td><strong>RefS21</strong></td>
<td>Linear Gain.</td>
</tr>
<tr>
<td><strong>DeltaGain21</strong></td>
<td>CompGain21 - Linear Gain.</td>
</tr>
</tbody>
</table>

### Noise Figure Cold Source

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NF</strong></td>
<td>Noise figure.</td>
</tr>
<tr>
<td><strong>T-Eff</strong></td>
<td>Effective noise temperature.</td>
</tr>
<tr>
<td><strong>DUTRNP</strong></td>
<td>DUT noise power ratio.</td>
</tr>
<tr>
<td><strong>SYSRNP</strong></td>
<td>System noise power ratio</td>
</tr>
<tr>
<td><strong>DUTNPD</strong></td>
<td>DUT noise power density.</td>
</tr>
<tr>
<td><strong>SYSNPD</strong></td>
<td>System noise power density.</td>
</tr>
<tr>
<td><strong>OvrRng</strong></td>
<td>Indication that the noise receiver is being over powered.</td>
</tr>
<tr>
<td><strong>T-Rcvr</strong></td>
<td>Temperature reading (in Kelvin) of the noise receiver board.</td>
</tr>
<tr>
<td><strong>S11</strong>, <strong>S21</strong>, <strong>S12</strong>, <strong>S22</strong></td>
<td>Standard S-parameters; measured with the port1 and port2 noise switches set for noise mode.</td>
</tr>
<tr>
<td><strong>A,1</strong>, <strong>A,2</strong> and so forth</td>
<td>Unratioed parameters; with notation: &quot;receiver, source port&quot;</td>
</tr>
</tbody>
</table>

**window** (long) Optional argument. Number of the window the new custom measurement will be placed in. Choose between 1 and the maximum number of windows allowed on the PNA. If unspecified, the measurement is placed in the active window.
Return Type | IMeasurement
---|---
Default | Not Applicable

**Examples**

'To create a scalar mixer measurement in channel 2:
Dim MyMeas as Agilent835x.Measurement
Set MyMeas = app.CreateCustomMeasurementEx (2, "Scalar Mixer/Converter", "SC21")

'To create a vector mixer measurement in channel 2:
Dim MyMeas as Agilent835x.Measurement
Set MyMeas = app.CreateCustomMeasurementEx (2, "Vector Mixer/Converter", "VC21")

**C++ Syntax**

HRESULT put_CreateCustomMeasurementEx (long ChannelNum, BSTR guid, VARIANT initData, long windowNumber, IMeasurement** ppMeasurement );

**Interface**

IApplication3

Last Modified:

8-Nov-2007   Updated for NF and GCA
CreateMeasurement Method

Description
Creates a new measurement.

VB Syntax
app.CreateMeasurement chanNum,param,lPort[,window]

Variable (Type) - Description

app Application (object)

chanNum (long) - Channel number of the new measurement; can exist or be a new channel

param (string) - New parameter. Case insensitive.

For S-parameters:

Any S-parameter that can be measured by your PNA.

Single-digit port numbers can be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

For Ratioed measurements:

Any two receivers in your PNA separated by "/". For example: "A/R1"

See the block diagram showing the receivers in YOUR PNA.

For Unratioed (absolute power) measurements:

Any receiver in the PNA. For example: "A"

See the block diagram showing the receivers in YOUR PNA

With PNA Rev 6.2, Ratioed and Unratioed measurements can also use logical receiver notation to refer to receivers. This notation makes it easy to refer to receivers with an external test set connected to the PNA. You do not need to know which physical receiver is used for each test port. Learn more.

For ADC measurements

Any ADC receiver in the PNA.

For example: "AI1" indicates the Analog Input1.

Learn more about ADC receiver measurements.

For Balanced S-parameter measurements:

"topology: Sabxy"

 topology - Choose from:

- sbal - single-ended to balanced
- **ssb** - single-ended / single-ended to balanced
- **bbal** - balanced to balanced

**Sabxy** -

Where

a - device output (receive) mode  
b - device input (source) mode  

(choose from the following for both a and b:)

- **d** - differential  
- **c** - common  
- **s** - single ended

x - device output (receive) logical port number  
y - device input (source) logical port number

For example: "**sbal:sdd42**"

See an example program

For **Imbalance** and **Common Mode Rejection** measurements:

"**topology:parameter**" Choose from:

<table>
<thead>
<tr>
<th>Choose this:</th>
<th>To get this:</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;SBAL:IMBSB&quot;</td>
<td>single-ended to balanced</td>
<td>imbalance</td>
</tr>
<tr>
<td>&quot;SBAL:CMRBSB1&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sds21/Scs21)</td>
</tr>
<tr>
<td>&quot;SBAL:CMRBSB2&quot;</td>
<td>single-ended to balanced</td>
<td>common mode rejection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Ssd12/Ssc12)</td>
</tr>
<tr>
<td>&quot;SSB:IMB1SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 1</td>
</tr>
<tr>
<td>&quot;SSB:IMB2SSB&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>imbalance 2</td>
</tr>
<tr>
<td>&quot;SSB:CMRSSSB1&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sds31/Ssc31)</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Measurement Type</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>&quot;SSB:CMRRSSB2&quot;</td>
<td>single-ended / single-ended to balanced</td>
<td>common mode rejection (Sds32/Scs32)</td>
</tr>
<tr>
<td>&quot;BBAL:IMB1BB&quot;</td>
<td>balanced to balanced</td>
<td>imbalance 1</td>
</tr>
<tr>
<td>&quot;BBAL:IMB2BB&quot;</td>
<td>balanced to balanced</td>
<td>imbalance 2</td>
</tr>
<tr>
<td>&quot;BBAL:CMRRBB&quot;</td>
<td>balanced to balanced</td>
<td>common mode rejection (Sdd21/Scc21)</td>
</tr>
</tbody>
</table>

**IPort** *(long)*

- **Ignored** if `param` is an S-Parameter, balanced, imbalance, or CMRR parameter.
- **Source port** if `param` is ratioed or unratioed (including ADC) measurements. Use `cap.getPortNumber` to get the number of named source. To learn more see [Remotely Specifying a Source Port](#).

**window** *(long)* Optional argument. Window number of the new measurement. Choose between 1 and the maximum number of windows allowed on the PNA. If unspecified, the measurement will be created in the Active Window. See also [Traces, Channels, and Windows on the PNA](#).

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**

```csharp
app.CreateMeasurement(1, "A/R1", 1, 0)
app.CreateMeasurement(1, "a1/b1", 1, 0)
app.CreateMeasurement(1, "bbal:Sdd21", 1)
app.CreateMeasurement(1, "AI2", 2)
```

See a Balanced Measurements example program.

**C++ Syntax**

```csharp
HRESULT CreateMeasurement(long ChannelNum, BSTR strParameter, long IPort, long windowNumber)
```

**Interface** `IApplication`

---

Last modified:

- **July 23, 2007**  Added source port link
- **April 25, 2007**  Updated for ADC measurements.
- **9/12/06**  MQ Updated for logical receiver notation.
## DataToMemory Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Stores the active measurement data into memory creating a memory trace. The memory can then be displayed or used in calculations with the measurement data.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>meas.DataToMemory</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>meas</td>
<td>A Measurement (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>meas.DataToMemory</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT DataToMemory()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMeasurement</td>
</tr>
</tbody>
</table>
## Delete Method

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Deletes the measurement.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>meas.Delete</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>meas</code></td>
<td>The Measurement object to delete <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>meas.Delete</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Delete()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IMeasurement</code></td>
</tr>
</tbody>
</table>
### DeleteMarker Method

**Description**  
Deletes a marker from the measurement.

**VB Syntax**  
`meas.DeleteMarker(Mnum)`

**Variable**  
- **`meas`**  
  A Measurement (object)

- **`Mnum`**  
  (long) - Any existing marker number in the measurement

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`meas.DeleteMarker(1)`

**C++ Syntax**  
`HRESULT DeleteMarker(long IMarkerNumber)`

**Interface**  
IMeasurement
## DeleteAllMarkers Method

**Description**
Deletes all of the markers from the measurement.

**VB Syntax**
meas.DeleteAllMarkers

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>(object) - The Measurement object from which markers will be deleted.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
meas.DeleteAllMarkers

**C++ Syntax**
HRESULT DeleteAllMarkers()

**Interface**
IMeasurement
DeleteCalSet Method

**Description**
Deletes a Cal Set from the set of available Cal Sets. This method immediately updates the Cal Set file on the hard drive. If the Cal Set is currently being used by a channel or does not exist, this request will be denied and an error is returned.

Using the [Cal Sets collection](#) is a convenient way to manage Cal Sets.

**VB Syntax**
```vbnet
calMgr.DeleteCalSet (calset)
```

**Variable**
- **(Type)** - **Description**
- **calMgr** (object) - A *CalManager* object
- **calset** (string) - Cal Set to be deleted. Specify the Cal Set by **GUID** or **Name**. Use [EnumerateCalSets](#) to list the available Cal Sets by name.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```vbnet
Set pna=CreateObject("AgilentPNA835x.Application")
Set cmgr = pna.GetCalManager
cmgr.DeleteCalSet ("MyCalSet")
```

**C++ Syntax**
```cpp
HRESULT DeleteCalSet( BSTR strCalset);
```

**Interface**
ICalManager

---

**Last Modified:**
6-Mar-2008  Added Name argument
## DeleteConfiguration Method

**Description**
Deletes the specified configuration name from the PNA. The factory configurations cannot be deleted. This is the only method of programmatically distinguishing a factory configuration from a user-named configuration.

**VB Syntax**

```vbnet
pathMgr.DeleteConfiguration name
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathMgr</td>
<td>PathConfigurationManager (object)</td>
</tr>
<tr>
<td>name</td>
<td>(String) Configuration name to be deleted.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
path.DeleteConfiguration "myMixer"
```

**C++ Syntax**

```cpp
HRESULT StoreConfiguration (long channelNum, BSTR configName );
```

**Interface**
IPathConfigurationManager

---

**Last Modified:**
14-Dec-2006   MX New topic
### DeleteShortCut Method

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Removes a macro from the list of macros in the analyzer. Does not remove the file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>There are always 12 macro positions. They do not have to be sequential. For example, you can have number 7 but no numbers 1 to 6.</td>
</tr>
</tbody>
</table>

#### VB Syntax
```vbnet
app.DeleteShortCut item
```

#### Variable (Type) - Description

- **app** An **Application** *(object)*
- **item** *(long integer)* number of the macro to be deleted.

#### Return Type
Not Applicable

#### Default
Not Applicable

#### Examples
```vbnet
app.DeleteShortCut 2
```

#### C++ Syntax
`HRESULT DeleteShortcut(long Number )`

#### Interface
`IApplication`
## DisallowAllEvents Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Sets event filtering to monitor NO eventst.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.DisallowAllEvents</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.DisallowAllEvents</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT DisallowAllEvents()</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
DisplayNAWindowDuringCalAcquisition Method

**Description**
Set and read the 'show' state of the window to be displayed during a calibration.

When this command is sent, the specified window is 'flagged' to be shown during calibration. The flag is cleared when the window is closed. A Preset or Instrument State Recall also closes the window. If the same window number is reopened, this command must be sent again to show the window during a calibration. The flag is NOT saved with an instrument state.

Send this command for each additional window to show during a calibration.

**VB Syntax**
```
calMgr.DisplayNAWindowDuringCalAcquisition (winNum, State)
```

**Variable**
<table>
<thead>
<tr>
<th><strong>(Type)</strong></th>
<th><strong>- Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>calMgr</td>
<td>(object) - A CalManager object</td>
</tr>
<tr>
<td>winNum</td>
<td>(long) - Window number to show during a calibration. The calibration window will also be shown with this window. The window must already be created. Use NaWindows.count or app.WindowNumber to read existing window numbers.</td>
</tr>
<tr>
<td>state</td>
<td>(Boolean) Window state. Choose from: True - Show the specified window during calibration. False - Do NOT show the specified window during calibration.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```
calMgr.DisplayNAWindowDuringCalAcquisition 2,True
```

See example using this command

**C++ Syntax**
```
HRESULT DisplayNAWindowDuringCalAcquisition( long WinNum, VARIANT_BOOL bVal);
```

**Interface**
ICalManager5
### DisplayOnlyCalWindowDuringCalAcquisition Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Clears the flags for windows to be shown during calibrations other than the Cal Window. To flag a window to be shown see DisplayNAWindowDuringCalAcquisition</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>calMgr.DisplayOnlyCalWindowDuringCalAcquisition</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>calMgr</strong> <em>(object)</em> - A CalManager object</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Example</td>
<td><code>calMgr.DisplayOnlyCalWindowDuringCalAcquisition</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT DisplayOnlyCalWindowDuringCalAcquisition()</code></td>
</tr>
</tbody>
</table>

**Interface**

ICalManager5

Last Modified:

8-Nov-2007 MX New topic
### DoPrint Method

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Prints the screen to the default Printer.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.DoPrint</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An Application <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.DoPrint</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT DoPrint()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>
# DoECAL1PortEx Method

**Description**
This method replaces [DoECAL1Port Method](#). Does a 1-Port calibration using an ECAL module. You must first have a 1-port measurement active to perform the calibration.

The characterization within the ECAL module that will be used for the calibration is specified by [ECALCharacterizationEx](#). The default value is 0.

**VB Syntax**
```
cal.DoECAL1PortEx [port][,module]
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cal</td>
<td>A Calibrator (object)</td>
</tr>
<tr>
<td><strong>port</strong></td>
<td>(long integer) Optional argument - Port number to calibrate. Choose from: 1 - Calibrate port 1 (default if unspecified) 2 - Calibrate port 2</td>
</tr>
<tr>
<td><strong>module</strong></td>
<td>(long integer) Optional argument. ECAL module. Choose from modules 1 through 8</td>
</tr>
</tbody>
</table>

Use [IsECALModuleFoundEx](#) to determine the number of modules connected to the PNA. Use [GetECALModuleInfoEx](#) to return the model and serial number of each module.

**Return Type**
None

**Default**
Not Applicable

**Examples**
```
cal.DoECAL1PortEx, 2, 2
```

**C++ Syntax**
```
HRESULT DoECAL1PortEx(long port, long moduleNumber = 1);
```

**Interface**
ICalibrator4
DoECAL2PortEx Method

**Description**
This method replaces DoECAL2Port Method. Does a 2-port calibration using an ECal module.

2-port refers to the number of ports to calibrate; NOT to the number of ECal module ports.

You must first have a measurement active to perform the calibration.

The characterization within the ECal module that will be used for the calibration is specified by ECalCharacterizationEx. The default value is 0.

**VB Syntax**

```
cal.DoECAL2PortEx [portA],[portB],[module]
```

**Variable (Type) - Description**

- **cal**
  A Calibrator (object)

- **portA** (long integer)
  Optional argument - Number of the receive port to calibrate. Choose from:
  1 - Calibrate port 1 (default, if unspecified)
  2 - Calibrate port 2
  3 - Calibrate port 3

  And so forth for all available PNA / test set ports.

- **portB** (long integer)
  Optional argument - Number of the source port to calibrate. Choose from:
  1 - Calibrate port 1
  2 - Calibrate port 2 (default, if unspecified)
  3 - Calibrate port 3

  And so forth for all available PNA / test set ports.

- **module** (long integer)
  Optional argument. ECal module.
  Choose from modules 1 through 8
  Use IsECALModuleFoundEx to determine the number of modules connected to the PNA
  Use GetECALModuleInfoEx to returns the model and serial number of each module.

**Return Type**
None

**Default**
Not Applicable

**Examples**
```
cal.DoECAL2PortEx, 1, 2, 3
```

**C++ Syntax**

```
HRESULT DoECAL2PortEx( long portA = 1, long portB =2, long moduleNumber = 1);
```

**Interface**
ICalibrator4
1-Jan-2007 Corrected Port B default
### DoneCalConfidenceCheckECAL Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Concludes the Confidence Check and sets the ECal module back into the idle state.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>cal.DoneCalConfidenceCheckECAL</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>cal</code></td>
<td>A Calibrator (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>cal.DoneCalConfidenceCheckECAL</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT DoneCalConfidenceCheckECAL();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>ICalibrator</td>
</tr>
</tbody>
</table>
**DoReceiverPowerCal Method**

**Description**

Note: This command replaces DataToDivisor, LogMagnitudeOffset, Normalization, InterpolateNormalization.

Immediately performs a receiver power calibration. The connection to the receiver must be in place when this command is sent.

A Receiver Power Cal requires that the active measurement be an Unratioed power measurement.

**VB Syntax**

```vbnet
cal.DoReceiverPowerCal(param, srcPort [,pwrOffset])
```

**Variable (Type) - Description**

*cal*    A *Calibrator* (object)

*param*  (string) – Receiver to be calibrated. Choose any receiver in your PNA. See a block diagram of your PNA.

With PNA Rev 6.2, receivers can also be referred to using logical receiver notation. This notation makes it easy to refer to receivers with an external test set connected to the PNA. You do not need to know which physical receiver is used for each test port. Learn more.

*srcPort* (long integer) – Number of the port which will supply source power to the receiver during this cal.

Use GetPortNumber to return the port number of a source that only has a string name, such as an External Source.

*pwrOffset* (double) – Optional argument. Offset value in dB. Adjusts a receiver power cal to account for components or adapters that are added between the source port and receiver while performing this cal. Specify loss as a negative number; and gain as a positive number.

**Return Type**

None

**Default**

Not Applicable

**Examples**

```vbnet
cal.DoReceiverPowerCal "B", 1, -10
```

**C++ Syntax**

```cpp
HRESULT DoReceiverPowerCal(BSTR parameter, long ISrcPort, double dPowerOffset);
```

**Interface**

ICalibrator5

---

Last Modified: 30-Apr-2007   Edited for src strings
**EnumerateCalSets Method**

**Description**
Returns an array of Cal Set names being stored on the PNA.

**VB Syntax**
```
value = calMgr.EnumerateCalSets
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(variant) - Variable to store the returned Cal Set names</td>
</tr>
<tr>
<td>calMgr</td>
<td>(object) - A CalManager object</td>
</tr>
</tbody>
</table>

**Return Type**
VARIANT array

**Default**
Not Applicable

**Example**
```
Dim pna
set
pna=CreateObject("AgilentPNA835x.Application")
Dim catalog
catalog=pna.getcalmanager.EnumerateCalSets
for i=lbound(catalog) to Ubound(catalog)
  wscript.echo catalog(i)
next
```

**C++ Syntax**
```
HRESULT EnumerateCalSets(VARIANT* names);
```

**Interface**
ICalManager4
### Execute Method

**Description**
Allows the use of COM to send a SCPI command.
This method can be used with :SYST:ERR? to convert scpi errors into text.
See an example of how to return error information when using the Parse method.

**VB Syntax**
```vbnet
Scpi.Execute (SCPI_Command)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scpi</td>
<td>A ScpiStringParser (Object)</td>
<td></td>
</tr>
</tbody>
</table>

**SCPI_Command**

- **(String)** - Any valid SCPI command

**Return Type**

- **String**

**Default**

- Not Applicable

**Examples**
```vbnet
Dim scpi As ScpiStringParser
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:PRES");
ErrorString = scpi.Execute("SYST:ERROr?");
```

**C++ Syntax**

```cpp
Execute(BSTR SCPI_Command, BSTR * pQueryResponse);
```

**Interface**

- IScpiStringParser2
### ExecuteShortcut Method

**Description**
Executes a Macro (shortcut) stored in the analyzer. Use `app.getShortcut` to list existing macros. Use `app.putShortcut` to associate the macro number with the file.

**VB Syntax**
```
app.ExecuteShortcut index
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>Application (object)</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>index</td>
<td>long integer</td>
<td>Number of the macro stored in the analyzer.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
app.ExecuteShortcut 1
```

**C++ Syntax**
```c++
HRESULT ExecuteShortcut(long index)
```

**Interface**
IAplication
# GenerateGlobalDeltaMatchSequence Method

**Description**

Initiates a global delta match calibration.  
[Learn more about Delta match calibration.](#)

See example of a complete Delta Match calibration.

**VB Syntax**

```vbnet
numSteps = guided.GenerateGlobalDeltaMatchSequence conn,cKit
```

**Variable (Type) - Description**

- **numSteps**: `Long Integer` - Variable to store the returned number of connection steps required by the Global Delta Match Cal.
- **guided**: `GuidedCalibration` (object)
- **conn**: `String` - Connector Type for port 1.
- **cKit**: `String` - Cal Kit for all ports.

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

```vbnet
guided.GenerateGlobalDeltaMatchSequence "APC 3.5 female","85052B"
```

**C++ Syntax**

```c++
HRESULT GenerateGlobalDeltaMatchSequence(BSTR port_1_conn, BSTR cal_kit, long *num_steps);
```

**Interface**

IGuidedCalibration2
### GenerateErrorTerms Method

**Description**
Generates the error terms for the specified calibration type, stores the error terms in a Cal Set, saves the Cal Set, and returns the Cal Set GUID.

If ALL the data for the cal type has NOT been acquired an error message is returned.

**Note:** The manner in which the calibration is assigned to a Cal Set (Cal Register or User Cal Set) is determined by the setting of RemoteCalStoragePreference.

#### VB Syntax

```vbnet
value = obj.GenerateErrorTerms
```

#### Variable *(Type) - Description*

- **value** (String) - Variable to store the returned GUID or error message.
- **obj** Any of the following:
  - `GuidedCalibration` (object)
  - `SMCType` (object)
  - `VMCType` (object)

#### Return Type

String

#### Default

Not Applicable

#### Examples

```vbnet
string = SMC.GenerateErrorTerms
```

#### C++ Syntax

```cpp
HRESULT put_GenerateErrorTerms(BSTR* calsetGUID);
```

#### Interface

- `IGuidedCalibration`
- `SMCType`
- `VMCType`
GenerateSteps Method

**Description**
Returns the number of steps required to complete the calibration type.

**VB Syntax**
```
value = obj.GenerateSteps
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(long) - Variable to store the returned number of steps</td>
</tr>
<tr>
<td>obj</td>
<td>Any of the following:</td>
</tr>
<tr>
<td>GuidedCalibration</td>
<td>(object)</td>
</tr>
<tr>
<td>SMCType</td>
<td>(object)</td>
</tr>
<tr>
<td>VMCType</td>
<td>(object)</td>
</tr>
</tbody>
</table>

**Return Type**
Long

**Default**
Not Applicable

**Examples**
```
value = SMC.GenerateSteps
```

**C++ Syntax**
```
HRESULT put_GenerateSteps(long* steps);
```

**Interface**
IGuidedCalibration
SMCType
VMCType
GetAuxIO Method

Description
This method returns the AuxIO interface.

VB Syntax
app.GetAuxIO

Variable
(Type) - Description
app  An Application (object)

Return Type
IHWAuxIO

Default
Not Applicable

Example
Dim app As AgilentPNA835x.Application
Dim aux As IHWAuxIO
Set aux = app.GetAuxIO

C++ Syntax
HRESULT GetAuxIO (IHWAuxIO **pAux);

Interface
IApplication
GetCalStandard Method

**Description**
Returns a handle to a calibration standard for modifying its definitions. To select a standard for performing a calibration (use Calibrator.AquireCalStandard).

**VB Syntax**
calkit.GetCalStandard(index)

**Variable (Type) - Description**
- **calkit** A calKit (object)
- **index** (long) - Number of calibration standard. Choose 1 to 30; (there are 30 cal standards in every kit).

**Return Type**
calStandard

**Default**
Not Applicable

**Examples**
Dim short As CalStandard
Set short = calKit.getCalStandard(1)
short.label = "myShort"

**C++ Syntax**
HRESULT GetCalStandard(long standardNumber, ICalStandard **pCalStd)

**Interface**
ICalKit
GetCalManager Method

Description
This method returns the ICalManager interface.

VB Syntax
app.GetCalManager()  

Variable Type - Description

app An Application (object)

Return Type
ICalManager*

Default Not Applicable

Example
dim app as AgilentPNA835x.Application  
dim mgr as CalManager  
set mgr = app.GetCalManager()

C++ Syntax
HRESULT GetCalManager( ICalManager **mgr);

Interface IApplication

Last Modified:
30-Jan-2008     Added parenthesis
Get CalSetByGUID Method

**Description**
Requests a Cal Set by GUID. Returns an ICal Set interface.

**VB Syntax**
```vbnet
calMgr.GetCalSetByGUID (GUID)
```

**Variable**
- **calMgr** *(object)* - A CalManager object
- **GUID** *(string)* - GUID of the Cal Set being requested.

**Return Type**
Interface object

**Default**
Not Applicable

**Example**
```vbnet
calMgr.GetCalSetByGUID (2B893E7A-971A-11d5-8D6C-00108334AE96)
```

**C++ Syntax**
```cpp
HRESULT GetCalSetByGUID( BSTR* strGUID, ICalSet* pCalSet);
```

**Interface**
ICalManager
GetCalSetCatalog Method - Superseded

Description
This method is replaced with EnumerateCalSets.
Returns a string containing a list of comma-separated GUIDs in the following format:
{FD6F863E-9719-11d5-8D6C-00108334AE96},
{1B03B2CE-971A-11d5-8D6C-00108334AE96},
{2B893E7A-971A-11d5-8D6C-00108334AE96}

VB Syntax
value = calMgr.GetCalSetCatalog

Variable
(value) - Description
value (string) - Variable to store the returned GUID list
calMgr (object) - A CalManager object

Return Type
String

Default
Not Applicable

Example
value = calMgr.GetCalSetCatalog

C++ Syntax
HRESULT GetCalSetCatalog(BSTR);

Interface
ICalManager

Last Modified:
6-Mar-2008  Superseded
Description
Returns a string identifying the Cal Set currently in use by the specified channel. This method identifies the Cal Set being used by returning its GUID.

This method also identifies the "Error Term set" within the Cal Set.

Error term sets are identified by integers, with set 0 belonging to the original (non-interpolated) terms. As stimulus values for a channel are changed causing interpolation to be required, a new Error Term set is constructed within the Cal Set to hold the interpolated Error Terms. The sets are sequentially numbered 1, 2, 3, and so forth. These Error Term sets are destroyed when they are no longer being used.

If there is no Cal Set in use for the given channel, the <GUID> argument is set to the empty string.

VB Syntax
```
calMgr.GetCalSetUsageInfo (chan, GUID, setNumber)
```

Variable (Type) - Description
```
calMgr (object) - A CalManager object
chan (long [in]) - channel of the Cal Set being requested
GUID (string [out]) - variable to store the GUID of the Cal Set being requested. If there is no Cal Set in use for the given channel, the <GUID> argument is set to the empty string.
setNumber (long [out]) - variable to store the error term ID being requested. If the returned argument is greater than 0, the set is being interpolated.
```

Return Type
String , Long Integer

Default
Not Applicable

Example
```
calMgr.GetCalSetUsageInfo (1, GUID, EtermID)
```

C++ Syntax
```
HRESULT GetCalSetUsageInfo (long lChannel, BSTR* CalSetGUID, long* etermSetID);
```

Interface
ICalManager
**GetCalTypes Method**

**Description**
Returns a list of available calibration types known to the PNA. The Standard CalTypes are the same on all PNA's, but the Custom CalTypes are not necessarily the same. They are dependent on the custom measurement in the PNA. Learn more about applying CalTypes.

See also CalibrationTypeID to apply a Cal Type containing in a Cal Set.

**VB Syntax**

```vb
v = mgr.GetCalTypes
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mgr</code></td>
<td>A CalManager (Object)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>v</code></td>
<td>Name/GuidPair that contains the calibration type name and associated GUID for each cal type known to the PNA.</td>
</tr>
</tbody>
</table>

**Return Type**

(variant) Two dimensional array.

**Default**

Not Applicable

**Examples**

```vb
v = CalManager.GetCalTypes
```

**C++ Syntax**

```c++
HRESULT GetCalTypes( VARIANT * NameGuidPair )
```

**Interface**

ICalManager2
GetComplex Method

**Description**
Retrieves complex data from the specified location.
See also `getNAComplex`, `getData`, and `getPairedData` Methods

**VB Syntax**
measData.getComplex location, numPts, real(), imag()

**Variable**
(Type) - Description

*measData*  An IArrayTransfer interface which supports the Measurement object

*location*  (enum NADataStore - IArrayTransfer) - Where the data you want is residing. Choose from:
0 - naRawData
1 - naCorrectedData
2 - naMeasResult
3 - naRawMemory
4 - naMemoryResult
5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

*numPts*  (long integer) - Number of data points requested
[out] - specifies number of data elements returned
[in] - specifies the data being requested or the capacity of the arrays

*real*  (single) - Array to store the real values

*imag*  (single) - Array to store the imaginary values

**Return Type**
Single

**Default**
Not Applicable

**Examples**
Dim real(201) AS Single
Dim imag(201) AS Single
Dim pts as Integer
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
measData.getComplex naCorrectedData, pts, real(0), imag(0)

**C++ Syntax**
IArrayTransfer - HRESULT getComplex(tagNADataStore DataStore, long* pNumValues, float* pReal, float* pImag)

**Interface**
IArrayTransfer
**getDataByString Method**

**Description**
Retrieves variant data from the specified location in your choice of formats. The PNA returns gather complex trace data which is ratioed if required by the measurement parameter, such as S11 or A/B. Otherwise it is raw receiver data, such as A or B.

**VB Syntax**
```
data = meas.getDataByString location, format
```

**Variable**

- **data** *(variant)* - Array to store the data.
- **meas** *(object)* - A Measurement object
- **location** *(string)* – Name of the buffer to be read. Choose from:
  - "naRawData"
  - "naCorrectedData"
  - "naMeasResult"
  - "naRawMemory"
  - "naMemoryResult"
  - "naDivisor" - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

  See Data Access Map

- **format** *(enum NADataFormat)* - Format in which you would like the data. It does not have to be the displayed format. Choose from:
  - 0 - naDataFormat_LinMag
  - 1 - naDataFormat_LogMag
  - 2 - naDataFormat_Phase
  - 3 - naDataFormat_Polar
  - 4 - naDataFormat_Smith
  - 5 - naDataFormat_Delay
  - 6 - naDataFormat_Real
  - 7 - naDataFormat_Imaginary
  - 8 - naDataFormat_SWR
  - 9 - naDataFormat_PhaseUnwrapped
  - 10 - naDataFormat_InverseSmith
  - 11 - naDataFormat_Kelvin
  - 12 - naDataFormat_Fahrenheit
13 - naDataFormat_Centigrade

Learn more about Data Format.

* Specify Smith or Polar formats to obtain complex data pairs, which require a two-dimensional array varData (numpts, 2) to accommodate both real and imaginary data.

All scalar formats return a single dimension varData(numpts).

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Variant array</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**  
meas.getDataByString "naMeasResult", naDataFormat_Phase

**C++ Syntax**  
HRESULT getDataByString( BSTR location, tagDataFormat dataFormat, VARIANT * pData );

**Interface**  
IMeasurement

---

Last Modified:

1-Oct-2007    Added temperature formats

19-Jul-2007    Corrected example
GetData Method

Description
Retrieves variant data from the specified location in your choice of formats. To get smoothed data from any of the specified locations, the format must be the same as the displayed format.

The PNA returns gather complex trace data which is ratioed if required by the measurement parameter, such as S11 or A/B. Otherwise it is raw receiver data, such as A or B.

This method returns a variant which is less efficient than methods available on the IArrayTransfer interface.

If you plan to Put this data back into analyzer, putDataComplex (variant data) method requires complex, two-dimensional data. Therefore, request the data in Polar format.

VB Syntax

```vbnet
data = meas.GetData location, format
```

Variable (Type) - Description

- **data**: Variant array to store the data.
- **meas**: A Measurement (object)
- **location**: (enum NADataStore) - Where the data you want is residing. See Data Access Map.
  - Choose from:
    - 0 - naRawData
    - 1 - naCorrectedData
    - 2 - naMeasResult
    - 3 - naRawMemory
    - 4 - naMemoryResult
    - 5 - naDivisor  When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

- **format**: (enum NADataFormat) - Format in which you would like the data. It does not have to be the displayed format. Choose from:
  - 0 - naDataFormat_LinMag
  - 1 - naDataFormat_LogMag
  - 2 - naDataFormat_Phase
  - 3 - naDataFormat_Polar*
  - 4 - naDataFormat_Smith*
  - 5 - naDataFormat_Delay
  - 6 - naDataFormat_Real
  - 7 - naDataFormat_Imaginary
Learn more about Data Format.

* Specify Smith or Polar formats to obtain complex data pairs, which require a two-dimensional array `varData (numpts, 2)` to accommodate both real and imaginary data. All scalar formats return a single dimension `varData(numpts)`. `naDataFormat_Phase` and `naDataFormat_PhaseUnwrapped` returns degrees. However, `putDataScalar` method accepts data in radians (not degrees) and displays in degrees.

**Return Type**  
Variant array - automatically dimensioned to the size of the data

**Default**  
Not Applicable

**Examples**

```vba
Dim varData As Variant
varData = meas.GetData(naMeasResult, naDataFormat_Phase)
'Print Data
For i = 0 to chan.NumberOfPoints-1
    Print varData(i)
Next i
```

**C++ Syntax**

```cpp
HRESULT getData(tagNADataStore DataStore, tagDataFormat DataFormat, VARIANT *pData)
```

**Interface**  
IMeasurement

---

Last Modified:

1-Oct-2007  Added temperature formats
GetECALModuleInfoEx Method

**Description**
This property replaces Get ECALModuleInfo Method. Returns the following information about the connected ECAL module: model number, serial number, connector type, calibration date, min and max frequency.

The characterization within the ECAL module that this information will be read from is specified by ECALCharacterizationEx. The default value is 0.

**VB Syntax**
```
moduleInfo = cal.GetECALModuleInfoEx(module)
```

**Variable (Type) - Description**
- **moduleInfo** (string) - variable to store the module information
- **cal** A Calibrator (object)
- **module** (long integer) - ECAL module.
  
  Choose from modules 1 through 8
  
  Use IsECALModuleFoundEx to determine the number of modules connected to the PNA

**Return Type**
String

**Default**
Not Applicable

**Examples**
```
info = cal.GetECALModuleInfoEx(2)
```

Example return string:
```
ModelNumber: 85092-60007, SerialNumber: 01386, ConnectorType: N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type N (50) female, MinFreq: 30000, MaxFreq: 9100000000, NumberOfPoints: 250, Calibrated: July 4 2002
```

**C++ Syntax**
```
HRESULT GetECALModuleInfoEx(long moduleNumber, BSTR* info);
```

**Interface**
ICalibrator4
### GetErrorCorrection Method

**Description**
Reads the error correction state for the channel.
Use [ErrorCorrection Property](#) to set this value.

When this command returns true, some measurements on the channel MAY not have error correction ON. This is because the Cal Set currently in place may not contain the appropriate calibration data. To read the error correction state for a measurement, use [Error Correction Property](#).

**VB Syntax**
```
chan.GetErrorCorrection (boolean)
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chan</td>
<td>A Channel (object)</td>
</tr>
<tr>
<td>boolean</td>
<td>(boolean) Variable to store the returned value.</td>
</tr>
<tr>
<td>False</td>
<td>Error correction has been set OFF</td>
</tr>
<tr>
<td>True</td>
<td>Error correction has been set ON</td>
</tr>
</tbody>
</table>

**Return Type**
Boolean

**Default**

**Examples**
```
chan.GetErrorCorrection(value)
```

**C++ Syntax**
```
HRESULT GetErrorCorrection (VARIANT_BOOL *bState)
```

**Interface**
IChannel8
**GetErrorTerm Method - Superseded**

**Description**

Note: This command is replaced by Get ErrorTermByString Method

Retrieves error term data that is used for error correction. The data is complex pairs. Memory for the returned Variant is allocated by the server. The server returns a variant containing a two-dimensional safe Array. This method returns a variant which is less efficient than getErrorTermComplex on the ICalData interface.

Learn about reading and writing Calibration data.

**VB Syntax**

```vbnet
data = cal.getErrorTerm term, rcv. src
```

**Variable**

*(Type) - Description*

- `data` Variant array to store the data.
- `cal` A Calibrator *(object)*
- `term` *(enum As NaErrorTerm)*. Choose from:
  - `naErrorTerm_Directivity_Isolation`
  - `naErrorTerm_Match`
  - `naErrorTerm_Tracking`
- `rcv` *(long integer)* - Receiver Port
- `src` *(long integer)* - Source Port

**To get this**

<table>
<thead>
<tr>
<th>Error Term</th>
<th>Specify these parameters:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fwd Directivity</td>
<td><code>term</code> = <code>naET_Directivity_Isolation</code></td>
</tr>
<tr>
<td>Rev Directivity</td>
<td><code>term</code> = <code>naET_Directivity_Isolation</code></td>
</tr>
<tr>
<td>Fwd Isolation</td>
<td><code>term</code> = <code>naET_Directivity_Isolation</code></td>
</tr>
<tr>
<td>Rev Isolation</td>
<td><code>term</code> = <code>naET_Directivity_Isolation</code></td>
</tr>
<tr>
<td>Fwd Source Match</td>
<td><code>term</code> = <code>naErrorTerm_Match</code></td>
</tr>
<tr>
<td>Rev Source Match</td>
<td><code>term</code> = <code>naErrorTerm_Match</code></td>
</tr>
<tr>
<td>Fwd Load Match</td>
<td><code>term</code> = <code>naErrorTerm_Match</code></td>
</tr>
<tr>
<td>Rev Load Match</td>
<td><code>term</code> = <code>naErrorTerm_Match</code></td>
</tr>
<tr>
<td>Fwd Reflection Tracking</td>
<td><code>term</code> = <code>naErrorTerm_Tracking</code></td>
</tr>
<tr>
<td>Rev Reflection Tracking</td>
<td><code>term</code> = <code>naErrorTerm_Tracking</code></td>
</tr>
</tbody>
</table>
Return Type  Variant

Default  Not Applicable

Examples  Dim varError As Variant
  varError = cal.getErrorTerm(naErrorTerm_Tracking, 2, 1)

C++ Syntax  HRESULT getErrorTerm(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, VARIANT* pData)

Interface  ICalibrator
**GetErrorTerm Method**  Superseded

**Description**
This command has been replaced with Get ErrorTermByString

Returns error term data from the Cal Set. The returned data is complex pairs.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

**VB Syntax**

```
data = calSet.getErrorTerm (setNumber, term, rcv, src)
```

**Variable**  (Type) - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>(Variant)</td>
<td>Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the PNA and must be released by client. <strong>Note:</strong> See also getErrorTermComplex on the ICalData2 interface to avoid using the variant data type.</td>
</tr>
<tr>
<td>calSet</td>
<td>A Cal Set (object)</td>
<td></td>
</tr>
<tr>
<td>setNumber</td>
<td>(Long)</td>
<td>There can be more than one set of error terms in a Cal Set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SetNumber 0 contains the original set of error terms for a Cal Set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SetNumbers &gt; 0 contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used. Learn about Interpolation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- To determine the SetNumber in use by a channel, see GetCalSetUsageInfo</td>
</tr>
<tr>
<td>term</td>
<td>(enum As NaErrorTerm2)</td>
<td>Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naET_Directivity  (rcv = src)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 - naET_SourceMatch   (rcv = src)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 - naET_ReflectionTracking  (rcv = src)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 - naET_TransmissionTracking (rcv src)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 - naET_LoadMatch     (rcv src)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 - naET_Isolation     (rcv src)</td>
<td></td>
</tr>
<tr>
<td>rcv</td>
<td>(Long) - Receiver Port</td>
<td></td>
</tr>
<tr>
<td>src</td>
<td>(Long) - Source Port</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**  Variant

**Default**  Not Applicable

**Examples**

```
Dim varError As Variant
varError = CalSet.getErrorTerm(0,naET_TransmissionTracking,2,1)
```
C++ Syntax

HRESULT getErrorTerm(long setID, tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, VARIANT* pData)

Interface

ICalSet
### GetErrorTermByString Method

**Description**
Returns error term data from the Cal Set by specifying the string name of the error term.

- Learn more about Reading and Writing Cal Data
- See examples of Reading and Writing Cal Set Data
- See GetCalSetUsageInfo to determine the setNumber.

#### VB Syntax
```
pdata = calset.GetErrorTermByString(setNumber, errorTerm)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pdata</code></td>
<td>Variant Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the PNA and must be released by client.</td>
</tr>
<tr>
<td><code>calset</code></td>
<td>A Cal Set (object)</td>
</tr>
<tr>
<td><code>setNumber</code></td>
<td>(Long) Set number of the required Cal Set data. See GetCalSetUsageInfo to determine the setNumber.</td>
</tr>
<tr>
<td><code>errorTerm</code></td>
<td>(String) The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be &quot;Directivity(3,3)&quot;. To determine the string names of error terms, see GetErrorTermList2.</td>
</tr>
</tbody>
</table>

**Return Type**
Variant

**Default**
Not Applicable

**Examples**
See an Example
**C++ Syntax**

HRESULT GetErrorTermByString(long SetNumber, BSTR bufferName, VARIANT* pData);

**Interface**

ICalSet2
GetErrorTermComplex Method  Superseded

**Description**
This command has been replaced by GetErrorTermComplexByString.
Retrieves error term data from the error correction buffer. The data is in complex pairs.
Learn more about reading and writing Cal Data using COM.
This method exists on a non-default interface. If you cannot access this method, use the GetErrorTerm Method on ICalibrator.

**VB Syntax**
```vb
eData.GetErrorTermComplex term, rcv, src, numPts, real(), imag()
```

**Variable**  
**Type** - Description

- **eData**  
  An ICalData pointer to the Calibrator object

- **term**  
  (enum NAErrorTerm) - The error term to be retrieved. Choose from:
  - naErrorTerm_Directivity_Isolation
  - naErrorTerm_Match
  - naErrorTerm_Tracking

- **rcv**  
  (long integer) - Receiver Port

- **src**  
  (long integer) - Source Port

- **numPts**  
  (long integer) - on input, max number of data points to return; on output: indicates the actual number of data points returned.

- **real()**  
  (single) - array to accept the real part of the error-term. One-dimensional for the number of data points.

- **imag()**  
  (single) - array to accept the imaginary part of the error-term. One-dimensional for the number of data points.
<table>
<thead>
<tr>
<th>Error Term</th>
<th>Specify these parameters:</th>
<th>term</th>
<th>rcv</th>
<th>src</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fwd Directivity</td>
<td>naET_Directivity Isolation</td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Rev Directivity</td>
<td>naET_Directivity Isolation</td>
<td></td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Isolation</td>
<td>naET_Directivity Isolation</td>
<td></td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Rev Isolation</td>
<td>naET_Directivity Isolation</td>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Source Match</td>
<td>naErrorTerm_Match</td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Rev Source Match</td>
<td>naErrorTerm_Match</td>
<td></td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Load Match</td>
<td>naErrorTerm_Match</td>
<td></td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Rev Load Match</td>
<td>naErrorTerm_Match</td>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Rev Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
<td></td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
<td></td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Rev Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
<td></td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

**Return Type**  
Single

**Default**  
Not Applicable

**Examples**  
ReDim rel(numpts)
ReDim img(numpts)
Dim eData As ICalData
Set eData = chan.Calibrator
eData.getErrorTermComplex naErrorTerm_Directivity_Isolation, 1, 1, 201, rel(0), img(0)

**C++ Syntax**  
HRESULT raw_getErrorTermComplex(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* plmag)

**Interface**  
ICalData
GetErrorTermComplex Method  Superseded

Description
This command is replaced with GetErrorTermComplexByString.

Returns error term data from the Cal Set. The data is in complex pairs.

Learn more about Reading and Writing Cal Data.

See examples of Reading and Writing Cal Set Data.

Note: This method exists on a non-default interface. If you cannot access this method, use the GetErrorTerm Method on ICal Set.

VB Syntax
icalData2.GetErrorTermComplex setNumber, term, rcv, src, numPts, real(), imag()

Variable (Type) - Description

icalData2 An ICalData2 pointer to the Cal Set object

setNumber (Long) There can be more than one set of error terms in a Cal Set.

- setNumber 0 contains the original set of error terms for a Cal Set.
- setNumbers > 0 contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used.

Learn about Interpolation.

- To determine the setNumber in use by a channel, see GetCalSetUsageInfo.

term (enum NAErrorTerm2) - The error term to be retrieved. Choose from:

0 - naET_Directivity
1 - naET_SourceMatch
2 - naET_ReflectionTracking
3 - naET_TransmissionTracking
4 - naET_LoadMatch
5 - naET_Isolation

rcv (Long) - Receiver Port

src (Long) - Source Port

numPts (Long) An In/Out parameter.

On the way in, you specify the max number of values being requested.

On the way out, the PNA returns number of values actually returned.

real() (Single) - array to accept the real part of the error-term. One-dimensional for the number of data points.
imag() *(single)* - array to accept the *imaginary* part of the error-term. One-dimensional for the number of data points.

**Return Type** Single

**Default** Not Applicable

**Examples**

```plaintext
dim numpts as long
numpts = ActiveChannel.NumberOfPoints
ReDim r(numpts) ' real part
ReDim i(numpts) ' imaginary part
Dim CalSet as CalSet
set CalSet = pna.GetCalManager.GetCal SetByGUID( txtGUID )
Dim eData As ICalData2
Set eData = CalSet
eData.getErrorTermComplex 0, naET_LoadMatch, 1, 2, numpts,
r(0),i (0)
```

**C++ Syntax**

```c++
HRESULT getErrorTermComplex(long setID, tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)
```

**Interface** ICalData2
### GetErrorTermComplexByString Method

**Description**
Returns error term data from the Cal Set by specifying the string name.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

**Note:** This method exists on a non-default interface. If you cannot access this method, use GetErrorTermByString.

**VB Syntax**
```
ICalData3.GetErrorTermComplexByString setNumber, errorTerm, numPoints, real(0), imag(0)
```

**Variable - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICalData3</td>
<td>ICalData3 pointer to a CalSet (Object)</td>
<td></td>
</tr>
<tr>
<td>setNumber</td>
<td>(Long)</td>
<td>There can be more than one set of error terms in a Cal Set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- setNumber 0 contains the original set of error terms for a Cal Set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- setNumbers &gt; 0 contain Interpolated error terms. Interpolated error terms are generated when interpolation is required and destroyed when no longer used. Learn about Interpolation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- To determine the setNumber in use by a channel, see GetCalSetUsageInfo</td>
</tr>
<tr>
<td>errorTerm</td>
<td>(String)</td>
<td>The string name of error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be &quot;Directivity(3,3)&quot;.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For a list error term string names, use GetErrorTermList2</td>
</tr>
<tr>
<td>numPoints</td>
<td>(Long)</td>
<td>An In/Out parameter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>On the way in, you specify the max number of values being requested.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>On the way out, the PNA returns number of values actually returned.</td>
</tr>
<tr>
<td>real</td>
<td>(Single)</td>
<td>The real component of the complex data.</td>
</tr>
<tr>
<td>imag</td>
<td>(Single)</td>
<td>The imaginary component of the complex data.</td>
</tr>
</tbody>
</table>

**Return Type**
Single

**Default**
Not Applicable

**Examples**
See example

**C++ Syntax**
```
HRESULT GetErrorTermComplexByString(long etermSetID, BSTR bufferName, long* inumPoints, single* real, single* imag);
```
## GetErrorTermList Method  Superseded

### Description

**Note:** This command is replaced by `CalSet.getErrorTermList2`.

Returns the list of Error Terms contained in this Cal Set for the CalType specified in the `OpenCal Set` method. Learn more about reading and writing Cal Data using COM.

The list is a comma separated, textual representation of the error terms with the term name followed by the port path in parentheses:

- `Term (n, n)`,
- `Term (m, n)`

Before calling this method you must open the Cal Set with `OpenCal Set`. If the Cal set is not open, this method returns `E_NA_Cal Set_ACCESS_DENIED`.

Use `StringToNAErrorTerm2` to convert the list entries to values that can be used with `GetErrorTerm` and `PutErrorTerm`.

**Note:** The port path designation `(m n)` indicates the ports that contribute to the error being compensated. Directivity, source match and reflection tracking are single port characteristics, designated in this list by `(n n)` where `n` equals the port being characterized. Other terms characterize the interaction between ports. For example, the load match term is describing the match at port `(m)` while looking into port `(n)`. Thus the notation `(m n)` indicates the two ports that contribute to the loadmatch error.

### VB Syntax

```vbnet
CalSet.GetErrorTermList (SetID, count, strList)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalSet</td>
<td>(object)</td>
<td>A Cal Set object</td>
</tr>
<tr>
<td>SetID</td>
<td>(long)</td>
<td>specifies the error term set to query. Use 0 for the master set.</td>
</tr>
<tr>
<td>count</td>
<td>(long)</td>
<td>the number of error terms in the returned list</td>
</tr>
<tr>
<td>strList</td>
<td>(string)</td>
<td>comma separated list of error terms found in Cal Set</td>
</tr>
</tbody>
</table>

**Return Type** Not Applicable

**Default** Not Applicable

### Examples

```vbnet
dim count as Integer
dim list as string
OpenCalSet (naCalType_TwoPortSOLT 1, 2)
GetErrorTermList ( 0, count, list)
CloseCalSet ( )
```

Assuming the cal set contained the full set of error terms for this two-port Cal, the returned list would be:

```
"Directivity(1 1),SourceMatch(1 1),ReflectionTracking(1 1),TransmissionTracking(2 1),LoadMatch(2 1),Isolation(2
```

1586
C++ Syntax
HRESULT GetErrorTermList (long etermSetID, long* count, BSTR* strList);

Interface
ICalSet
Description  Returns a list of error terms names found in the Cal Set containing the specified prefix. Learn more about Reading and Writing Cal Data See examples of Reading and Writing Cal Set Data

VB Syntax  

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>list</td>
<td>Variant containing a string array of error term names.</td>
</tr>
<tr>
<td>CalSet</td>
<td>object - A CalSet object</td>
</tr>
<tr>
<td>setNumber</td>
<td>(Long) There can be more than one set of error terms in a Cal Set.</td>
</tr>
<tr>
<td></td>
<td>• setNumber 0 contains the original set of error terms for a Cal Set.</td>
</tr>
<tr>
<td></td>
<td>• setNumbers &gt; 0 contain Interpolated error terms. Interpolated error terms are</td>
</tr>
<tr>
<td></td>
<td>generated when interpolation is required and destroyed when no longer used.</td>
</tr>
<tr>
<td></td>
<td>Learn about Interpolation.</td>
</tr>
<tr>
<td></td>
<td>• To determine the setNumber in use by a channel, see GetCalSetUsageInfo.</td>
</tr>
<tr>
<td>caltypePrefix</td>
<td>(String) The string used to identify Cal Set data as belonging to a specific Cal Type.</td>
</tr>
<tr>
<td></td>
<td>This string is used as a filter so that only the error term names of interest are</td>
</tr>
<tr>
<td></td>
<td>returned. If the prefix is empty, all terms are returned.</td>
</tr>
<tr>
<td></td>
<td>An example prefix for a two port cal on ports 2 and 3 might be: &quot;Full 2 Port Cal (2,3)&quot;.</td>
</tr>
</tbody>
</table>

Return Type  Variant

Default  Not Applicable

Examples  See an Example

C++ Syntax  HRESULT GetErrorTermList2 (long SetNumber, BSTR caltypePrefix, VARIANT* list)

Interface  ICalSet2
GetExtendedCalInterface Method

**Description**
Returns an interface that exposes the properties of Noise Calibration.

**VB Syntax**
```vbnet
Cal2.GetExtendedCalInterface (interface)
```

**Variable**
- **Cal2**: An ICalibrate2 (object)
- **interface**: (object) Returns a handle to the specified interface. Choose from 'NoiseCal'

**Return Type**
- Default

**Example**
```vbnet
dim noiseCal
dim noiseCalExtensions
set noiseCal = Get Calmanager?.CreateCustomCalEx("NoiseCal")
set noiseCalExtensions = noiseCal.GetExtendedCalInterface("INoiseCal")
```

**C++ Syntax**
```csharp
HRESULT GetExtendedCalInterface();
```

**Interface**
ICalibrate2

Last Modified:
29-May-2007   MN New topic
### Get ExternalTestSetIO Method

**Description**  This method returns the `IExternalTestSetIO` interface.

**VB Syntax**  
`app.GetExternalTestSetIO`

**Variable**  
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>app</code></td>
<td>An <code>Application</code> (object)</td>
</tr>
</tbody>
</table>

**Return Type**  IHWExternalTestSetIO

**Default**  Not Applicable

**Example**  
```vbnet
Dim app As AgilentPNA835x.Application
Dim ets As HWEternalTestSetIO
Set ets = app.GetExternalTestSetIO
```

**C++ Syntax**  
`HRESULT GetExternalTestSetIO (IHWExternalTestSetIO **pTestset);`

**Interface**  IApplication
Description


VB Syntax

```vbnet
meas.GetFilterStatistics cf,bw,loss,q
```

Variable (Type) - Description

- `meas` A Measurement (object)
- `cf,bw,loss,q` Dimensioned variables to store the returned values

Return Type (double) `cf`

(single) `bw,loss,q`

Default Not Applicable

Examples

```vbnet
'Dimension variables
Dim cf as Double
Dim bw as Single
Dim loss as Single
Dim q as Single
meas.GetFilterStatistics cf,bw,loss,q
```

C++ Syntax

```cpp
HRESULT GetFilterStatistics(double* centerFreq, float* bw, float* loss, float* quality)
```

Interface IMeasurement
GetGuid Method

Description: Returns a string containing the GUID identifying this Cal Set. Each Cal Set is assigned a GUID (global unique ID). GUIDs are used to retrieve and select Cal Sets on the PNA. Learn more about reading and writing Cal Data using COM.

VB Syntax

```
value = CalSet.GetGuid
```

Variable (Type) - Description

- `value` (string) - Variable to store the returned GUID
- `CalSet` (object) - A Cal Set object

Return Type: String

Default: Not Applicable

Examples

```
guid = CalSet.GetGuid 'Read
```

C++ Syntax

```
HRESULT GetGUID( BSTR* pGUIDString);
```

Interface: ICalSet
get_InputVoltageEX Method

Description
This command replaces get InputVoltage Method
Reads the ADC voltage from the specified location.

VB Syntax
volts = AuxIO.get_InputVoltageEX loc

Variable (Type) - Description
volts (double) - variable to store the return value
AuxIO (object) - A Hardware Auxiliary Input / Output object
loc (Long) Location from which to read data.

For PNA-X models:
1 Reads voltage on Analog In 1 port (pin 7).
2 Reads voltage on Analog In 2 port (pin 8).
3 Reads voltage on GndSens (pin 6).
4 Reads voltage on Analog Out 1 port (pin 3).
5 Reads voltage on Analog Out 2 port (pin 4).

For all other PNA models:
1 Reads voltage on the Analog IN (pin 14) of the AUX IO connector.
4 Reads voltage on Analog Out 1 port (pin 3).
5 Reads voltage on Analog Out 2 port (pin 2).

Return Type Double

Default Not Applicable

Examples
Dim aux as HWAuxIO
Set aux = PNA.getAuxIO
volts = aux.get_InputVoltageEX 1
'for PNA-X, read voltage on PowerI/O pin 7
'for all other models, reads voltage on Aux I/O Analog In (pin 14)

C++ Syntax HRESULT get_InputVoltageEX (long muxLoc, double* vtVoltage );

Interface HWAuxIO2

Last Modified:
### Description
Reads a hardware latch that captures high to low transitions on Input1 of the Material Handler IO. Reading the latch causes it to reset and is ready for the next transition. The hardware latch is only capable of capturing one transition per query. Additional transitions are ignored until after the next query.

Momentarily driving Input1 high, then low, causes a transition to be detected and latched.

### VB Syntax

```vbnet
inp1 = handlerIo.get_Input1
```

### Variable (Type) - Description

- **inp1** *(variant)* - A variable to store the return value
- **handlerIo** *(object)* - A HandlerIO object

### Return Type
Variant -

- **0** - a high to low transition occurred at Input1 since the last time it was queried.
- **1** - no high to low transition occurred.

### Default
Not Applicable

### Examples

```vbnet
input1 = handlerIo.get_Input1  'Read
```

### C++ Syntax

```cpp
HRESULT get_Input1 (VARIANT* Data);
```

### Interface
IHWMaterialHandlerIO
### GetIsolationPaths Method

**Description**
Gets the list of paths (port pairings) for which isolation standards will be measured during calibration.

**VB Syntax**

```
value = obj.GetIsolationPaths
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(Variant) - Variable to store the returned port paths in pairs. One-dimensional array of Long Integers.</td>
</tr>
<tr>
<td>obj</td>
<td>Any of the following:</td>
</tr>
<tr>
<td></td>
<td>GuidedCalibration (object)</td>
</tr>
</tbody>
</table>

**Return Type**
Variant – Containing one-dimensional array of Long Integers.

**Default**
No port pairs (empty Variant variable)

**Examples**

```vbnet
pathList = guidedCal.GetIsolationPaths
'displaying the paths separated by commas, 'with a dash (-) between the pair of port numbers comprising each path
For i = LBound(portList) To UBound(portList) Step 2
    msg = msg + CStr(portList(i)) + "-" + CStr(portList(i+1))
If i+1 < UBound(portList) Then msg = msg + ","
Next
MsgBox msg, 0, "List of isolation paths"
```

**C++ Syntax**

```
HRESULT GetIsolationPaths(VARIANT* pathList);
```

**Interface**
IGuidedCalibration

---

**Last Modified:** 16-Apr-2007   MX New topic
## Get MaterialHandlerIO Method

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>This method returns the <a href="#">MaterialHandlerIO</a> interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.GetMaterialHandlerIO</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type)</em> - <em>Description</em></td>
</tr>
<tr>
<td><code>app</code></td>
<td>An <a href="#">Application</a> <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>IHWMaterialHandlerIO</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>Dim app As AgilentPNA835x.Application</code></td>
</tr>
<tr>
<td></td>
<td><code>Dim hand As IHWMaterialHandlerIO</code></td>
</tr>
<tr>
<td></td>
<td><code>Set hand = app.GetMaterialHandlerIO</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT GetMaterialHandlerIO (IHWMaterialHandlerIO **phand);</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>

---

- [MaterialHandlerIO](#)
GetNAComplex Method

Description
Retrieves complex data from the specified location. See also getComplex and getData Method.

VB Syntax
measData.getNAComplex location, numPts, data

Variable (Type) - Description

measData An IArrayTransfer interface which supports the Measurement object

location (enum NADataStore) - Where the data you want is residing. Choose from:
0 - naRawData
1 - naCorrectedData
2 - naMeasResult
3 - naRawMemory
4 - naMemoryResult
5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

numPts (long integer) - Number of data points requested
[out] - specifies number of data elements returned
[in] - specifies the data being requested or the capacity of the dComplex array

data (NAComplex) - A one-dimensional array of NaComplex to store the data.

Return Type
NAComplex

Default
Not Applicable

Examples
Dim dComplex(201) AS NAComplex
Dim measData As IArrayTransfer
Dim pts as Long
Set measData = app.ActiveMeasurement
measData.getNAComplex naCorrectedData, pts, dComplex(0)

Notes
The data is stored as Real and Imaginary (Re and Im) members of the NaComplex user defined type. You can access each number individually by iterating through the array.

For i = 0 to NumPts-1
    dReal (i) = dcomplex (i).Re
    dImag (i) = dcomplex (i).Im
Next i

C++ Syntax
HRESULT getNAComplex(tagNADataStore DataStore, long* pNumValues, TsComplex* pComplex)
Interface IArrayTransfer
**GetNumberOfGroups Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns the number of groups a channel has yet to acquire. To set the number of groups for a channel, use Number Of Groups Method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>value = chan.GetNumberOfGroups</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>value</code></td>
<td>(Long Integer) - Number of groups</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>Channel (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>(Long Integer)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>groups = chan.GetNumberOfGroups</code> 'Read</td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT GetNumberOfGroups(long* numberOfGroups);</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IChannel3</td>
</tr>
</tbody>
</table>
**get_Output Method**

**Description**

**Type 1 and Type2 configurations:** Returns the last value written to the selected output pin.

**Type3 configuration:** Returns the current state of the selected output pin. If an Input1 trigger occurs, the state may not be the same value as was written.

All configurations: Data is written using `put_Output` Method.

**VB Syntax**  
`data = handlerIo.get_Output(pin)`

**Variable**

- **data** (**variant**) - A variable to store the return value. The returned value will be one of the following:
  - 0 - TTL Low
  - 1 - TTL High

- **handlerIo** (**object**) - A HandlerIO object

- **pin** (**enum as NAMatHandlerOutput**) - output to read. Choose from:
  - `naOutput1` (0)
  - `naOutput1User` (1)
  - `naOutput2` (2)
  - `naOutput2User` (3)

Learn about User Output

**Return Type**

Variant

**Default**

Not Applicable

**Examples**

`data = handlerIo.get_Output(naOutput1)`

**C++ Syntax**

```cpp
HRESULT get_Output ( tagNAMatHandlerOutput Output, VARIANT* Data );
```

**Interface**

`IHWMaterialHandlerIO`
## Description

**E836x and PNA-L**: Reads voltages on the DAC/Analog Output 1|2 of the Auxiliary IO connector.

**PNA-X**: Reads voltage on the Power I/O connector AnalogOut1|2.

## VB Syntax

```vbnet
volts = AuxIO.get_OutputVoltage(output)
```

## Variable (Type) - Description

- **volts** (double) - variable to store the return value
- **AuxIO** (object) - A Hardware Auxiliary Input / Output object
- **output** (variant) Number of the output DAC from which to read voltage. Choose from:
  1. Output 1 (Aux I/O pin 3) and (Power I/O pin 3)
  2. Output 2 (Aux I/O pin 2) and (Power I/O pin 4)

## Return Type

Double

## Default

Not Applicable

## Examples

```vbnet
Dim aux as HWAuxIO
Set aux = PNA.getAuxIO
volts = aux.get_OutputVoltage(1)
' read voltage from Analog Out 1 (Aux I/O pin 3) or (Power I/O pin 3)
```

## C++ Syntax

```cpp
HRESULT get_OutputVoltage(VARIANT Output, double* Voltage);
```

## Interface

IHWAuxIO

---

**Last Modified:**

10-Jul-2007   Added PNA-X capability
Read-only
get OutputVoltageMode Method

<table>
<thead>
<tr>
<th>Description</th>
<th>This command returns the mode of the selected &quot;Analog Out&quot; line on the Auxiliary IO connector and Power I/O connector. The modes give the user the option to have the requested voltage applied immediately or not until the sweep is done. To set the mode, use put_OutputVoltageMode Method.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>VB Syntax</th>
<th><code>mode = auxIo.get_OutputVoltageMode (output)</code></th>
</tr>
</thead>
</table>

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>(enum NAOutputVoltageMode) - variable to store the returned mode.</td>
</tr>
<tr>
<td>naWaitEOS</td>
<td>While in this mode any voltage changes sent to the selected analog out will only get applied to the output between sweeps.</td>
</tr>
<tr>
<td>naNoWait</td>
<td>While in this mode any voltage changes sent to the selected analog out will occur right away without waiting until the end of a sweep, the voltage gets applied immediately.</td>
</tr>
<tr>
<td>auxIo</td>
<td>(object) - A Hardware Auxiliary Input / Output object</td>
</tr>
<tr>
<td>output</td>
<td>(double) Analog Output. Choose from 1 or 2</td>
</tr>
</tbody>
</table>

**Return Type**

eNum as NAOutputVoltageMode

**Default**

naWaitEOS

**Examples**

```vbnet
tOutMode = auxIo.get_OutputVoltageMode (1)
```

**C++ Syntax**

```cpp
HRESULT get_OutputVoltageMode(VARIANT Output, tagNAOutputVoltageMode* pMode);
```

**Interface**

IHWAuxIO

---

**Last Modified:**

10-Jul-2007 Added PNA-X capability
GetPairedData Method

Description
Retrieves pairs of data from the specified location.

Note: This method exists on a non-default interface. If you cannot access this method, use the Get Data Method on IMeasurement.

VB Syntax
`measData.getPairedData location, format, numPts, d1, d2`

Variable Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>measData</code></td>
<td>Any IArrayTransfer interface which supports the Measurement object</td>
<td></td>
</tr>
<tr>
<td><code>location</code></td>
<td>(enum NADataStore)</td>
<td>Where the data you want is residing. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - naRawData</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - naCorrectedData</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - naMeasResult</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - naRawMemory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 - naMemoryResult</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.</td>
</tr>
<tr>
<td><code>format</code></td>
<td>(enum NAPairedDataFormat)</td>
<td>Format in which you would like the Paired data. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - naLogMagPhase - Log magnitude and phase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - naLinMagPhase - Linear magnitude and phase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - naRealImaginary - Real and Imaginary</td>
</tr>
<tr>
<td><code>numPts</code></td>
<td>(long integer)</td>
<td>Number of data points requested</td>
</tr>
<tr>
<td><code>d1</code></td>
<td>(single)</td>
<td>Array to store the magnitude / real values</td>
</tr>
<tr>
<td><code>d2</code></td>
<td>(single)</td>
<td>Array to store the phase / imaginary values</td>
</tr>
</tbody>
</table>

Return Type
Two Single arrays

Default
Not Applicable

Examples
```vbnet
Dim logm() As Single
Dim phase() As Single
Public measData As IArrayTransfer
```
Set measData = app.ActiveMeasurement
Dim numpts As Long
numPoints = app.ActiveChannel.NumberOfPoints
ReDim logm(numPoints)
ReDim phase(numPoints)

measData.getPairedData naCorrectedData, naLogMagPhase, numPoints, logm(0), phase(0)
Print values(0), values(1)

**C++ Syntax**

```cpp
HRESULT getPairedData(tagNADataStore DataStore, tagNAPairedDataFormat PairFormat, long* pNumValues, float* pReal, float* pImag)
```

**Interface**

IArrayTransfer
get_Port Method

**Description**
Returns the value from the specified "readable" port.

**VB Syntax**
```vbnet
data = handlerIo.get_Port(port)
```

**Variable (Type)** - Description

- **data (variant)** - A variable to store the return value. The following table shows what the returned data represents:

<table>
<thead>
<tr>
<th>Port</th>
<th>MSB...........................................LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>...................................................0</td>
</tr>
<tr>
<td>0</td>
<td>...................................................C3...C0</td>
</tr>
<tr>
<td>3</td>
<td>...................................................D3...D0</td>
</tr>
<tr>
<td>2</td>
<td>...................................................D3...D0 + C3...C0</td>
</tr>
</tbody>
</table>

- **handlerIo (object)** - A HandlerIO object
- **port (enum as NAMatHandlerPort)** - port to get data from. Choose from:
  - naPortC - (2)
  - naPortD - (3)
  - naPortE - (4)

**Note**: Reading data from the Write-only ports (A,B,F,G,H) will return an error. Ports C and D must be put in Read mode before reading from C, D, or E using PortMode Property.

**Return Type**
Variant

**Default**
0

**Examples**
```vbnet
data = handlerIo.get_Port(naPortC)
```

**C++ Syntax**
```cpp
HRESULT get_Port ( tagNAMatHandlerPort Port, VARIANT* Data );
```

**Interface**
IHWMaterialHandlerIO
### get_PortCData Method

| Description | Reads a 4-bit value from Port C of the Aux I/O connector (pins 22-25) and the Material Handler IO (pins 21-24 Anritsu) - (pins 22-25 Avantest).  
**Note:** These lines are connected to both the Handler IO and Aux IO in the PNA. |

| VB Syntax |  
| --- | --- |
| value = AuxIO.get_PortCData |  

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>(variant) - Variable to store the returned data</td>
</tr>
<tr>
<td>AuxIO</td>
<td>(object) - A Hardware Auxiliary Input / Output object</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>None</td>
</tr>
</tbody>
</table>

| Examples | value = auxIo.get_PortCData 'Reading a value of 15 when in Positive Logic indicates Port C lines C0, C1, C2, C3 are High. If in Negative Logic they are Low. |

| C++ Syntax | HRESULT get_PortData( VARIANT* Data ); |

| Interface | IHWAuxIO |
GetPortNumber Method

Description
Returns the port number that is associated with the specified port name. These numbers are used with several commands to specify a PNA port.

To learn more, see Remotely Specifying a Source Port.

VB Syntax
\[ value = cap.GetPortNumber PortName \]

Variable (Type) - Description

\[ value \] (Long) - Variable to store the returned Port Number integer value.

\[ cap \] A Capabilities (object)

\[ PortName \] (String) Name of the PNA port.

- Use SourcePortNames Property to return a list of PNA port (string) names.
- If an external source is selected, specify the external source name that is used in the Select an External Source dialog.

Return Type
Long Integer

Default
Not Applicable

Examples
\[ value = cap.GetPortNumber 'Read \]

C++ Syntax
HRESULT GetPortNumber(BSTR name, long *number);

Interface
ICapabilities4

Last Modified:
April 30, 2007   Edited for Ext Source Control
GetRaw2DData Method

**Description**
Returns raw 2D data at all frequency and power data points for the current Gain Compression measurement.
If correction is on, corrected data are returned. Otherwise, raw data are returned.

**VB Syntax**
```
data = gca.GetRaw2DData location, format, param
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Variant array in which to store returned measurement data.</td>
</tr>
<tr>
<td>gca</td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td>location</td>
<td>(enum NADataStore) - Where the data you want is residing. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naRawData</td>
</tr>
<tr>
<td></td>
<td>1 - naCorrectedData</td>
</tr>
<tr>
<td>format</td>
<td>(enum NADataFormat) - Format in which you would like the data. It does not have to be the displayed format. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naDataFormat_LinMag</td>
</tr>
<tr>
<td></td>
<td>1 - naDataFormat_LogMag</td>
</tr>
<tr>
<td></td>
<td>2 - naDataFormat_Phase</td>
</tr>
<tr>
<td></td>
<td>3 - naDataFormat_Polar*</td>
</tr>
<tr>
<td></td>
<td>4 - naDataFormat_Smith*</td>
</tr>
<tr>
<td></td>
<td>5 - naDataFormat_Delay -- <strong>Not valid for this command.</strong></td>
</tr>
<tr>
<td></td>
<td>6 - naDataFormat_Real</td>
</tr>
<tr>
<td></td>
<td>7 - naDataFormat_Imaginary</td>
</tr>
<tr>
<td></td>
<td>8 - naDataFormat_SWR</td>
</tr>
<tr>
<td></td>
<td>9 - naDataFormat_PhaseUnwrapped</td>
</tr>
<tr>
<td></td>
<td>10 - naDataFormat_InverseSmith</td>
</tr>
</tbody>
</table>

* Specify Smith or Polar formats to obtain complex data pairs, which require a two-dimensional array varData (numpts, 2) to accommodate both real and imaginary data.

All scalar formats return a single dimension varData(numpts).
naDataFormat_Phase and naDataFormat_PhaseUnwrapped returns degrees.

<table>
<thead>
<tr>
<th>param</th>
<th>(String) Parameter of data to return. Not case-sensitive. The specified parameter need NOT be displayed.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Choose from:</td>
</tr>
</tbody>
</table>

**Learn more about Data Format.**
- "pin" - input power at each data point.
- "pout" - output power at each data point.
- "gain" - device gain (S21) at each data point.
- "inputmatch" - input match (S11) at each data point.

**Return Type**  
Variant Array

**Default**  
Not Applicable

**Examples**

**C++ Syntax**

```
HRESULT GetRaw2DData(tagNADataStore location, tagNADataFormat format, BSTR data_name, VARIANT* pData);
```

**Interface**  
IGainCompression

---

Last Modified:

22-Oct-2007  
MX New topic
### GetDataIm Method

**Description**  
Reads the Imaginary part of the data acquired from a 2-dimensional sweep.

**VB Syntax**  
\[
data = gca.GetDataIm\; stim,\; dPoint,\; param
\]

**Variable**  
**Type** - Description

- **data**  
  Variant array in which to store returned measurement data.

- **gca**  
  A [GainCompression](#) (object)

- **stim**  
  (NAGCAIndexSelect)
  - **naFrequencySelect** - for the specified frequency data point, returns all of the measured data for each power stimulus.
  - **naPowerSelect** - for the specified power data point, returns all of the measured data for each frequency stimulus.

- **dPoint**  
  Data point (Frequency or Power) for which data is returned.

- **param**  
  Parameter of data to return. Not case-sensitive. Choose from:
  - "pin" - input power at each data point.
  - "pout" - output power at each data point.
  - "gain" - device gain (S21) at each data point.
  - "inputmatch" - input match (S11) at each data point.

**Return Type**  
Variant Array

**Default**  
Not Applicable

**Examples**  
For the fifth frequency data point, returns 'Power Output' imaginary (phase) data from all power stimulus values. If there are 30 power sweep points, 30 values are returned.

```vbnet
data = gca.GetDataIm\; naFrequencySelect,\; 5,\; "pout"
```

For the 30th stimulus power data point, returns 'Power Output' imaginary (phase) data from all frequency stimulus values. If there are 201 frequency sweep points, 201 values are returned.

```vbnet
data = gca.GetDataIm\; naPowerSelect,\; 30,\; "pout"
```

**C++ Syntax**  
HRESULT GetDataIm(tagNAGCAIndexSelect index_select, int index, BSTR data_name, VARIANT* pData);
Interface IGainCompression

Last Modified:

22-Oct-2007 MX New topic
**GetDataRe Method**

**Description**
Reads the REAL part of the data acquired from a 2-dimensional sweep.

**VB Syntax**
```
data = gca.GetDataRe stim, dPoint, param
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>data</code></td>
<td>Variant</td>
<td>Variant array in which to store returned measurement data.</td>
</tr>
<tr>
<td><code>gca</code></td>
<td><code>GainCompression</code> object</td>
<td></td>
</tr>
<tr>
<td><code>stim</code></td>
<td><code>NAGCAIndexSelect</code></td>
<td>For the specified frequency data point, returns all of the measured data for each power stimulus.</td>
</tr>
<tr>
<td><code>dPoint</code></td>
<td>Data point (Frequency or Power)</td>
<td>for which data is returned.</td>
</tr>
<tr>
<td><code>param</code></td>
<td>Parameter</td>
<td>Parameter of data to return. Not case-sensitive. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- &quot;pin&quot; - input power at each data point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- &quot;pout&quot; - output power at each data point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- &quot;gain&quot; - device gain (S21) at each data point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- &quot;inputmatch&quot; - input match (S11) at each data point.</td>
</tr>
</tbody>
</table>

**Return Type**
Variant Array

**Default**
Not Applicable

**Examples**
For the fifth frequency data point, returns 'Power Output' REAL data from all power stimulus values. If there are 30 power sweep points, 30 values are returned.
```
data = gca.GetDataRe naFrequencySelect, 5, "pout"
```

For the 30th stimulus power data point, returns 'Power Output' REAL data from all frequency stimulus values. If there are 201 frequency sweep points, 201 values are returned.
```
data = gca.GetDataRe naPowerSelect, 30, "pout"
```

**C++ Syntax**
```
HRESULT GetDataRe(tagNAGCAIndexSelect index_select, int index, BSTR data_name, VARIANT* pData);
```
Interface IGainCompression

Last Modified:

22-Oct-2007    MX New topic
### GetReferenceMarker Method

**Description**
Returns a handle to the reference marker.

**VB Syntax**
```vbnet
meas.GetReferenceMarker
```

**Variable**
- **Type**: Description
  - `meas`: A Measurement *(object)*

**Return Type**
Object

**Default**
Not Applicable

**Examples**
```vbnet
meas.GetReferenceMarker
```

**C++ Syntax**
```cpp
HRESULT GetReferenceMarker(IMarker** refMarker)
```

**Interface**
IMeasurement
GetRequiredEtermNames

Description
Returns an array of strings specifying the error terms required by the caltype's correction algorithm in order to correct the specified parameter. This function interrogates a specific caltype (caltypeGUID) for the list of error terms it would need in order to correct the specified parameter. All the standard S Parameter calibration types embed port specifiers in the error term name. The specific port information is gleaned from the passed parameter. For example, to query the error term requirements specific to a two port cal on ports 1 and 3, issue this with a parameter of S13 or S31. The buffer names returned will be formatted in this way:

Full 1 Port SOLT(1,3):TransmissionTracking(3,1)

VB Syntax
EtermNames = GetRequiredEtermNames(CalTypeGUID As String, Parameter As String)

Variable (Type) - Description

caltypeGUID: [in] the GUID of the desired calibration type

parameter [in] string specifying the parameter to be corrected

EtermNames [out] array of strings containing the error term names.

Note: In C++ Allocated by server. Must be freed by caller using SysFreeString.

Return Type
Not Applicable

Default
Not Applicable

Examples
enames = GetRequiredEtermNames(ctGUID, Parm)

C++ Syntax
HRESULT GetRequiredEtermNames( BSTR caltypeGUID, BSTR parameter,  VARIANT* EtermNames )

Interface
ICalManager2
GetScalar Method

**Description**
Retrieves scalar data (ONE number per data point) from the specified location.

**Note:** This method exists on a non-default interface. If you cannot access this method, use the Get Data Method on IMeasurement.

**VB Syntax**
```vbnet
measData.getScalar location, format, numPts, data
```

**Variable (Type) - Description**

- **measData**
  An IArrayTransfer interface which supports the Measurement object

- **location** *(enum NADataStore)* - Where the data you want is residing. Choose from:
  0 - naRawData
  1 - naCorrectedData
  2 - naMeasResult
  3 - naRawMemory
  4 - naMemoryResult
  5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

- **format** *(enum NADataFormat)* - Format in which you would like the data. Choose from:
  0 - naDataFormat_LinMag
  1 - naDataFormat_LogMag
  2 - naDataFormat_Phase
  3 - naDataFormat_Polar
  4 - naDataFormat_Smith
  5 - naDataFormat_Delay
  6 - naDataFormat_Real
  7 - naDataFormat_Imaginary
  8 - naDataFormat_SWR
  9 - naDataFormat_PhaseUnwrapped
  10 - naDataFormat_InverseSmith
  11 - naDataFormat_Kelvin
  12 - naDataFormat_Fahrenheit
  13 - naDataFormat_Centigrade

**Note:** Polar, Smith, and Inverse Smith are invalid formats for this command. See Get Complex Method.
Learn more about Data Format.

**numPts**

*(long integer)* - Number of data points requested

- **[out]** - specifies number of data elements returned
- **[in]** - specifies the data being requested or the capacity of the *dScalar* array

**data** *(single)* - Array to store the scalar data.

**Return Type** Single

**Default** Not Applicable

**Examples**

```vba
Dim dScalar() As Single
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
Dim numpts as Long
numpts = app.ActiveChannel.NumberOfPoints
ReDim dScalar(numPoints)
measData.getScalar naCorrectedData, naDataFormat_LogMag, numpts, dScalar(0)
Print dScalar(0), dScalar(1)
```

**C++ Syntax**

```c++
HRESULT getScalar(tagNADataStore DataStore, tagNADataFormat DataFormat, long* pNumValues, float* pVals)
```

**Interface** IArrayTransfer

Last Modified:

1-Oct-2007  Added temperature formats
GetShortcut Method

**Description**
Returns the Title, Path, and optional argument strings, of the specified Macro (shortcut). Use this method to list the titles and paths of macros in the analyzer.

**VB Syntax**
```vbnet
app.GetShortcut index, title, path, arguments
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>Application (object)</td>
<td></td>
</tr>
<tr>
<td>index</td>
<td>long</td>
<td>Number of the macro. Use a number between 1 and 12.</td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>Title of the specified macro. (Appears in the softkey label)</td>
</tr>
<tr>
<td>path</td>
<td>string</td>
<td>Pathname of the specified macro.</td>
</tr>
<tr>
<td>arguments</td>
<td>string</td>
<td>Arguments for the specified macro</td>
</tr>
</tbody>
</table>

**Return Type**
String

**Default**
Not Applicable

**Example**
```vbnet
Dim t As String
Dim p As String
Dim arg As String
Dim i As Integer
For i = 1 to 12
    app.GetShortcut i, t, p, arg
    Print t, p
Next
```

**C++ Syntax**
```c
HRESULT GetShortcut(long Number, BSTR* title, BSTR* pathname, BSTR* arguments)
```

**Interface**
IApplication

**Remarks**
Shortcuts can also be defined and accessed using the macro key on the front panel. However, the benefit of this feature is primarily for the interactive user.
GetSnPData Method  **Superseded**

**Description**

Note: this command is replaced by GetSnpDataWithSpecifiedPorts Method.

Reads SnP data from the selected measurement. Learn more about SnP that is returned from the PNA.

**VB Syntax**

\[
data = meas.GetSnPData type\]

**Variable (Type) - Description**

- **data**: Variant array to store the data.
- **meas**: A Measurement (object)
- **type**: (string) - Type of SnP data to return. If unspecified, \(<n>\) is set to 2. Choose from:
  - "S1P" returns data for the active measurement.
  - "S2P" returns data for the current 2-port measurement (4 S-parameters).
  - "S3P" returns data for the current 3 port measurement (9 S-parameters). Valid only on instruments with 3 ports or more.
  - "S4P" returns data for the current 4 port measurement (16 S-parameters). Valid only on instruments with 4 ports or more.

SnP data can be output using several data formatting options. See SnPFormat Property

**Return Type**

Variant array - automatically dimensioned to the size of the data

**Default**

Not Applicable

**Examples**

\[
snp = meas.GetSnPData("s1p")\]

**C++ Syntax**

HRESULT GetSnPData( BSTR snptype, VARIANT * response)

**Interface**

IMeasurement3
GetSnPDataWithSpecifiedPorts Method

**Description**

**Note:** This command replaces **Get SnPData**. This command is more explicit regarding the data to be returned, and works for PNAs with multiport test sets.

Reads SnP data for the measurement by specifying the PNA port numbers. [Learn more about SnP that is returned from the PNA.](#)

**VB Syntax**

```
data = meas.GetSnPDataWithSpecifiedPorts ports
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>data</em></td>
<td><em>(Variant)</em> array to store the data.</td>
</tr>
<tr>
<td><em>meas</em></td>
<td>A <strong>Measurement</strong> <em>(object)</em></td>
</tr>
<tr>
<td><em>ports</em></td>
<td><em>(Variant Array)</em> One-dimensional array containing the list of port numbers for which data is required.</td>
</tr>
</tbody>
</table>

**Return Type**

Variant array - automatically dimensioned to the size of the data.

**Default**

Not Applicable

**Example**

'**This VBScript example can be pasted into a notepad file and run on the PNA as a macro.** Learn how.

```vbnet
Dim pna
Dim meas
Dim param
Dim point
Dim snp
Dim ports

'List the port numbers for required data
ports = Array(3,4)
Set pna = CreateObject("AgilentPnA835x.application")
Set meas = pna.ActiveMeasurement

'limit amount of data to display
set chan=pna.ActiveChannel
chan.NumberOfPoints=2
snp = meas.GetSnPDataWithSpecifiedPorts (ports)

' returns a 3 dimensional array
' snp(param,point,data pair)
```

---

1621
' show me the data
For param = LBound(snp, 1) To UBound(snp, 1)
    MsgBox ("Parameter: " & (param + 1))
    For point = LBound(snp, 2) To UBound(snp, 2)
        MsgBox "Point:" & (point + 1) & " " & snp(param, point, 0) & "," & snp(param, point, 1)
    Next
Next

C++ Syntax
HRESULT GetSnpDataWithSpecifiedPorts(VARIANT portsToMeasure, VARIANT* response);

Interface
IMeasurement7

Last modified:
9/18/06     MQ Added for multiport
**getSourcePowerCalDataEx Method**

**Description**
Note: This method replaces `getSourcePowerCalData Method`

Retrieves (as variant data type) source power calibration data, if it exists, from the channel.

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the Get X-Axis Values command to return the X-axis values in the displayed order.

**Note:** This method returns a variant which is less efficient than methods available on the ISourcePowerCalData interface.

**VB Syntax**

```vbnet
data = chan.getSourcePowerCalDataEx (buffer, sourcePort)
```

**Variable (Type) - Description**

- **data** (variant) – Array to store the data.
- **chan** (object) – A Channel object
- **buffer** (enum NASourcePowerCalBuffer) - The requested source power cal data buffer.
  - 0 - naCorrectionValues This is the only buffer currently available
- **sourcePort** (long integer) – The source port for which calibration data is being requested.
  Use GetPortNumber to return the port number of a source that only has a string name, such as an External Source.

**Return Type**
Variant array – automatically dimensioned to the size of the data.

**Default**
Not Applicable

**Examples**

```vbnet
Dim varData As Variant
Const port1 As Long = 1
varData = chan.getSourcePowerCalDataEx (naCorrectionValues, port1)
'Print the data
For i = 0 to chan.NumberOfPoints - 1
    Print varData(i)
Next i
```

**C++ Syntax**

```c++
HRESULT getSourcePowerCalDataEx(tagNASourcePowerCalBuffer bufSelect, long sourcePort, VARIANT *pData);
```

**Interface**
IChannel4

Last Modified:
27-Jun-2007   Updated for PNA-X source port names
getSourcePowerCalDataScalarEx Method

Description  
Note: This method replaces `getSourcePowerCalDataScalar Method`.
Retrieves (as scalar values) source power calibration data, if it exists, from this channel.
If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the `Get X-Axis Values2` command to return the X-axis values in the displayed order.

Note: This method exists on a non-default interface. If you cannot access this method, use the `getSourcePowerCalDataEx Method` on IChannel4.

VB Syntax  
```
chanData.getSourcePowerCalDataScalarEx buffer, sourcePort, numValues, data
```

Variable  
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>chanData</code></td>
<td>(interface) – An ISourcePowerCalData2 interface on the Channel object.</td>
</tr>
<tr>
<td><code>buffer</code></td>
<td>(enum NASourcePowerCalBuffer) – The requested source power cal data buffer. 0 - naCorrectionValues Currently this is the only buffer available.</td>
</tr>
<tr>
<td><code>sourcePort</code></td>
<td>(long integer) – The source port for which calibration data is being requested.</td>
</tr>
<tr>
<td></td>
<td>Use <code>GetPortNumber</code> to return the port number of a source that only has a string name, such as an External Source.</td>
</tr>
<tr>
<td><code>numValues</code></td>
<td>(long integer) – Number of data values.</td>
</tr>
<tr>
<td></td>
<td>[out] – specifies number of data values returned.</td>
</tr>
<tr>
<td></td>
<td>[in] – specifies number of values being requested (this must not be larger than the capacity of the data array).</td>
</tr>
<tr>
<td><code>data</code></td>
<td>(single) – Array to store the data.</td>
</tr>
</tbody>
</table>

Return Type  
Single

Default  
Not Applicable

Examples  
```
Dim numValues As Long
Dim scalarCalValues() As Single
Dim chanData As ISourcePowerCalData2
Const port1 As Long = 1
numValues = app.ActiveChannel.NumberOfPoints
ReDim scalarCalValues(numValues)
Set chanData = app.ActiveChannel

chanData.getSourcePowerCalDataScalarEx naCorrectionValues, port1, numValues, scalarCalValues(0)
```
'Print the data
For i = 0 to numValues - 1
Print scalarCalValues(i)
Next I

C++ Syntax
HRESULT getSourcePowerCalDataScalarEx(tagNASourcePowerCalBuffer bufSelect,
long sourcePort, long *pNumValues, float *pData);

Interface ISourcePowerCalData2

Last Modified:
27-Jun-2007 Updated for PNA-X source port names
GetStandard Method  **Superseded**

**Description**
This command has been replaced with [Get StandardByString](#). Returns standard acquisition data from the Cal Set. The returned data is complex pairs. Learn more about [Reading and Writing Cal Data](#). See examples of [Reading](#) and [Writing](#) Cal Set Data.

**VB Syntax**
```
data = CalSet.getStandard (standard, rcv, src)
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>data</strong></td>
<td>(Variant) Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the PNA and must be released by client. <strong>Note:</strong> See also getStandardComplex on the ICalData2 interface to avoid using the variant data type.</td>
</tr>
</tbody>
</table>

**CalSet** A Cal Set (object)

**standard** (enum NACalClass) Standard data to be read. Choose from:

1 - naClassA
2 - naClassB
3 - naClassC
4 - naClassD
5 - naClassE
6 - naReferenceRatioLine
7 - naReferenceRatioThru

**SOLT Standards**

1 - naSOLT_Open
2 - naSOLT_Short
3 - naSOLT_Load
4 - naSOLT_Thru
5 - naSOLT_Isolation

**TRL Standards**

1 - naTRL_Reflection
2 - naTRL_Line_Reflection
3 - naTRL_Line_Tracking
4 - naTRL_Thru
5 - naTRL_Isolation

rcv (long) - Receiver Port
src (long) - Source Port

Return Type (variant)

Default Not Applicable

Examples

Dim varStd As Variant
Dim varStd2 As Variant

Cal Set.OpenCalSet(naCalType_TwoPortSOLT, 1, 2)
varStd = CalSet.getStandard(naSOLT_Thru, 2, 1)
varStd2 = CalSet.getStandard(naSOLT_Thru, 1, 2)
Cal Set.CloseCalSet()

C++ Syntax

HRESULT getStandard(tagNACalClass stdclass, long ReceivePort, long SourcePort, 
VARIANT* pData)

Interface ICalSet
### GetStandardByString Method

**Description**  Returns standard acquisition data from the Cal Set. The returned data is complex pairs. Learn more about Reading and Writing Cal Data
See examples of Reading and Writing Cal Set Data

**VB Syntax**  

```
data = calSet.GetStandardByString(stdName)
```

**Variable (Type) - Description**

- **data (Variant)**  Two-dimensional safe array to store the returned data. Memory for the returned Variant is allocated by the PNA and must be released by client.
  
  **Note:** See also Get StandardComplexByString on the ICalData2 interface to avoid using the variant data type.

- **calSet (Object)**

- **stdName (String)**  The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".

**Return Type**  Variant

**Default**  Not Applicable

**Examples**  See an example

**C++ Syntax**  

```cpp
HRESULT GetStandardByString( BSTR bufferName, VARIANT* pdata)
```

**Interface**  ICalSet2
GetStandardComplex Method  Superseded

Description
This command is replaced with Get StandardComplexByString.

Returns standard acquisition data from the Cal Set. The returned data is complex pairs.

Learn more about Reading and Writing Cal Data.

See examples of Reading and Writing Cal Set Data.

Note: This method exists on a non-default interface. If you cannot access this method, use the GetStandard Method on ICal Set.

VB Syntax
ICalData2.getStandardComplex class, rcv, src, numPts, real(), imag()

Variable (Type) - Description

ICalData2 An ICalData2 pointer to the Cal Set object

class (enum NACalClass) Standard data to be read. Choose from:

1 - naClassA
2 - naClassB
3 - naClassC
4 - naClassD
5 - naClassE
6 - naReferenceRatioLine
7 - naReferenceRatioThru

SOLT Standards

1 - naSOLT_Open
2 - naSOLT_Short
3 - naSOLT_Load
4 - naSOLT_Thru
5 - naSOLT_Isolation

TRL Standards

1 - naTRL_Reflection
2 - naTRL_Line_Reflection
3 - naTRL_Line_Tracking
4 - naTRL_Thru
5 - naTRL_Isolation

rcv  (long) - Receiver Port

src  (long) - Source Port

numPts  (Long) An In/Out parameter.
On the way **in**, you specify the **max** number of values being requested.
On the way **out**, the PNA returns number of values actually returned.

real()  (**single**) - array to accept the real part of the calibration data. One-dimensional for the number of data points.

imag()  (**single**) - array to accept the imaginary part of the calibration data. One-dimensional for the number of data points.

**Return Type**  (**single**)  

**Default**  Not Applicable

**Examples**  

```vba
Dim numpts as long
numpts = ActiveChannel.NumberOfPoints
ReDim r(numpts) ' real part
ReDim i(numpts) ' imaginary part
Dim Cal Set as Cal Set
set Cal Set = pna.GetCalManager.GetCal SetByGUID( txtGUID )
Dim sData As ICalData2
Set sData = Cal Set
sdata.getStandardComplex naSOLT_Open, 1, 1, numpts, r(0), i(0)
```

**C++ Syntax**  

```cpp
HRESULT getStandardComplex(tagNACalClass stdclass, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pimag)
```

**Interface**  ICalData2
GetStandardComplexByString Method

Description
Returns standard acquisition data from the Cal Set.
Learn more about Reading and Writing Cal Data
See examples of Reading and Writing Cal Set Data

VB Syntax
ICalData3.GetStandardComplexByString stdName, lnumPoints, real(0), imag(0)

Variable
(Type) - Description
ICalData3 An ICalData3 pointer to a CalSet (Object)

stdName (String) The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".

lnumPoints (Long) An In/Out parameter.
On the way in, you specify the max number of values being requested.
On the way out, the PNA returns number of values actually returned.

real (Single) The real component of the complex data.

imag (Single) The imaginary component of the complex data.

Return Value
Single

Default
Not Applicable

Examples
See example

C++ Syntax
HRESULT GetStandardComplexByString( BSTR bufferName, long* lnumPoints, float* real, float* imag);

Interface
ICalData3
GetStandardsList Method  Superseded

Description  

Note: This command is replaced by CalSet.getStandardList2.

Returns the list of Standards contained in this Cal Set for the CalType specified in the OpenCal Set method. Learn more about reading and writing Cal Data using COM.

The list is a comma separated, textual representation of the error terms with the term name followed by the port path in parentheses.

Standard (n, n),
Standard (m, n)

Before calling this method you must open the Cal Set with OpenCal Set. If the Cal Set is not open, this method returns E_NA_Cal Set_ACCESS_DENIED.

Use StringToNACalClass to convert the list entrees to values that can be used with GetStandard and PutStandard.

Note: The port path designation (m n) indicates the receive and source ports for the measurement. Shorts, opens and loads are single port devices, designated in this list by (n n) where n equals the port to which the device is connected. These devices are all characterized by reflection measurements.

The dual port thru device is characterized by both transmission and reflection measurements in order to compensate for load match and tracking terms.

The notation (n n) indicates the reflection measurement for this device.

The notation (m n) indicates the transmission measurement, where the source and receive ports are different.

VB Syntax  

CalSet.GetStandardsList (count, list)

Variable  

(Type) - Description

CalSet (object) - A Cal Set object

count (long [out]) - indicates the number of items returned in the list

list (string) - Variable to store the returned Comma separated list of items.

Return Type  String

Default  Not Applicable

Examples  

dim count as Integer

dim list as string

OpenCalSet (naCalType_TwoPortSOLT, 1, 2)

GetStandardsList( count, list)

CloseCalSet( )

Assuming the Cal Set contained the full set of standards for this two port cal, the returned list would be:

"Open(1 1),
Short(1 1),
Load(1 1),
Thru(1 1),
Isolation(2 1),
Open(2 2),
Short(2 2),
Load(2 2),
Thru(2 2),
Isolation(1 2)
Thru(2 1),
Thru(1 2)"

C++ Syntax  HRESULT GetStandardsList( long* count, BSTR* list);

Interface  ICalSet
### GetStandardList2 Method

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Returns a list of standards contained by this Cal Set for the specified Cal Type.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>list = calset.GetStandardList2(calType)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><strong>calset</strong> (object) - A <a href="#">CalSet</a> object</td>
</tr>
<tr>
<td></td>
<td><strong>list</strong> (Variant) - Variant containing a string array of standards for the specified calType.</td>
</tr>
<tr>
<td></td>
<td><strong>calType</strong> (String) - The string used to identify Cal Set data as belonging to a specific Cal Type. This string is used as a filter so that only the standard names of interest are returned. If the prefix is empty, all names are returned. An example prefix for a two port cal on ports 2 and 3 might be: &quot;Full 2 Port Cal (2,3)&quot;.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Variant</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>See an example</td>
</tr>
</tbody>
</table>

**C++ Syntax**

```
HRESULT GetStandardList2 (BSTR caltype, VARIANT* list)
```

**Interface**

ICalSet2
**GetStandardsForClass Method**

**Description**
Get the calibration standard numbers for a specified calibration class. To set the calibration number use `SetStandardsForClass Method`.

**VB Syntax**
```vbnet
calKit.GetStandardsForClass (calclassorder, std1, std2, std3, std4, std5, std6, std7)
```

**Variable (Type) - Description**

- `calKit` A CalKit `(object)`

- `calclassorder` *(enum NACalClassOrder)* Choose from:
  
  0 - naRefl_1_S11  
  1 - naRefl_2_S11  
  2 - naRefl_3_S11  
  3 - naTran_1_S21  
  4 - naRefl_1_S22  
  5 - naRefl_2_S22  
  6 - naRefl_3_S22  
  7 - naTran_1_S12  
  8 - naRefl_1_S33  
  9 - naRefl_2_S33  
  10 - naRefl_3_S33  
  11 - naTran_1_S32  
  12 - naTran_1_S23  
  13 - naTran_1_S31  
  14 - naTran_1_S13  
  15 - naTRL_T  
  16 - naTRL_R  
  17 - naTRL_L

- `std1…std7` *(long)* Calibration Standard Number. Nominal values from 1 through 30. 0 indicates that a standard number has not been selected.
**Return Type**  Not applicable

**Default**  Not applicable

**Examples**  `calkit.GetStandardsForClass naRef1_3_S11, std1, std2, std3, std4, std5, std6, std7`

**C++ Syntax**  `HRESULT GetStandardsForClass(NACalClassOrder calclassorder, long std1, long std2, long std3, long std4, long std5, long std6, long std7)`

**Interface**  `ICalKit`
GetStepDescription Method

**Description**
Returns the description of the specified step in the calibration process.

**VB Syntax**
```vbnet
value = obj.GetStepDescription (n)
```

**Variable**
- **value** (string) - Variable to store the returned number of steps.
- **obj** Any of the following:
  - GuidedCalibration (object)
  - SMCType (object)
  - VMCType (object)
- **n** (Long) Step in the calibration process.
  Use GenerateSteps to determine the total number of steps.

**Return Type**
String

**Default**
Not Applicable

**Examples**
```vbnet
value = SMC.GetStepDescription(5)
```

**C++ Syntax**
```cpp
HRESULT get_GetStepDescription(long step, BSTR* str);
```

**Interface**
- IGuidedCalibration
- SMCType
- VMCType
GetSupportedALCModes Method

**Description**
Returns the valid ALC Modes for the PNA.
See [ALCLevelingMode](#) for a list of supported ALC Modes.

**VB Syntax**
```
value = chan.GetSupportedALCModes(sourcePort)
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Variant Array variable to store the returned valid ALC Modes.</td>
</tr>
<tr>
<td>chan</td>
<td>(object) - A Channel object</td>
</tr>
<tr>
<td>sourcePort</td>
<td>(long integer) - Source port.</td>
</tr>
</tbody>
</table>

Use [GetPortNumber](#) to return the port number of a source that only has a string name, such as an External Source.

**Return Type**
Variant array

**Default**
Not Applicable

**Examples**
```
modes = chan.GetSupportedALCModes(4)  'Read
```

**C++ Syntax**
```
HRESULT get_GetSupportedALCModes(long port, VARIANT * ALCModes );
```

**Interface**
IChannel10
GetTestResult Method

Description
Returns the result of limit line testing. There are three ways to use this command:

- If neither optional parameter is specified, limit results for ALL data is returned.
- If one parameter is specified (start), the limit result for that data point is returned.
- If both parameters are specified, limit results are returned beginning with start, and ending with (start+size)-1

VB Syntax

\[
\text{testRes} = \text{limts.GetTestResult}[\text{start}, \text{size}]
\]

Variable

\text{testRes} (\text{enum NALimitTestResult}) - A dimensioned variable to store test results. If a limit line is not tested, a PASS is returned.

\begin{align*}
0 & \text{ - naLimitTestResult_None} \\
1 & \text{ - naLimitTestResult_Fail} \\
2 & \text{ - naLimitTestResult_Pass}
\end{align*}

\text{limts} \text{ A LimitTest (object)}
\text{start} (\text{long}) - Optional argument. A start data point number to return limit test results.
\text{size} (\text{long}) - Optional argument. Number of data points from start to return limit test results.

Return Type
Long Integer

Default
Not Applicable

Examples

\begin{verbatim}
Dim testRes As NALimitTestResult
testRes = limts.GetTestResult
Select Case testRes
    Case 1
        Print "Fails"
    Case 2
        Print "Pass"
End Select
\end{verbatim}

C++ Syntax

\[
\text{HRESULT GetTestResult}(\text{long lStart}, \text{long lSize}, \text{tagNALimitTestResult *pVal})
\]

Interface
ILimitTest
### GetTraceStatistics Method

**Description**
Returns all four Trace Statistics. To retrieve individual Trace statistics, use `Mean`, `PeakToPeak`, `StandardDeviation` properties. Use `ShowStatistics` to display the statistics of the screen.

**VB Syntax**
```
meas.GetTraceStatistics pp, mean, stdev
```

**Variable (Type)**
- `meas` (Type) - Description

  A Measurement (object)

- `pp, mean, stdev` (double) - Dimensioned variables to store the returned values

**Return Type**
Double

**Default**
Not Applicable

**Examples**
```
' Dimension variables
Dim pp As Double
Dim mean As Double
Dim stdv As Double
meas.GetTraceStatistics pp, mean, stdv
```

**C++ Syntax**
```
HRESULT GetTraceStatistics(double* pp, double* mean, double* stdDeviation)
```

**Interface**
`IMeasurement`
GetXAxisValues2 Method

Description
Returns the channel's X-axis values into a dimensioned Typed array. GetXAxisValues2 is a convenient method for determining the frequency of each point when the points are not linearly spaced - as in segment sweep.

Note: This method will fail if called using a scripting client such as VBScript or Agilent Vee, (see remarks

Note: In Segment Sweep, chan.NumberofPoints will return the total number of data points for the combined segments.

VB Syntax
chan.GetXAxisValues2 numPts, data

Variable  (Type) - Description
chan  (object) - A Channel object
numPts  (long integer) - Number of data points in the channel
data  (double) Single dimensioned array of data matching the number of points in the channel.

Return Type  double

Default  Not applicable

Examples
Dim App As Application
Set App = New Application
Dim numPoints As Long
Dim values() As Double
numPoints = App.ActiveChannel.NumberOfPoints
ReDim values(numPoints)
App.ActiveChannel.GetXAxisValues2 numPoints, values(0)
Print values(0), values(1)

C++ Syntax
HRESULT GetXAxisValues2(long* pNumValues, double* stimulus)

Remarks:
This method will fail if called using a scripting client such as VBScript or Agilent Vee. Use the GetXAxisValues method as a replacement for these COM environments.

This method also cannot be called using late-bound typing in Visual Basic. For instance, if, in the example above, the first line were replaced with "Dim App as Object", then this method would fail.
GetXAxisValues Method

Description
Returns the stimulus values for the measurement. To understand how this property is useful, see IMeasurement2 Interface.

VB Syntax
```
data = meas.GetXAxisValues
```

Variable (Type) - Description
- **data** (Variant) Array to store the data.
- **meas** A Measurement (object)

Return Type
Variant

Default
Not Applicable

Examples
```
Dim varData As Variant
Dim i As Integer
varData = meas.GetXAxisValues
' Print Data
For i = 0 To meas.NumberOfPoints - 1
    Print varData(i)
Next i
```

See C++ example

C++ Syntax
```
HRESULT GetXAxisValues(VARIANT* xData);
```

Interface
IMeasurement2

Last Modified:
15-Oct-2007 Added link to C++ example
Read-only

**About Segment Sweep**

**GetXAxisValues Method**

**Description**
Returns the channel's X-axis values. GetXAxisValues is a convenient method for determining the frequency of each point when the points are not linearly spaced - as in segment sweep.

See the [Measurement2 Interface](#) to learn how this method differs from `meas.GetXAxisValues`.

*Note:* This method returns a variant which is less efficient than `GetXAxisValues2`.

*Note:* In Segment Sweep, `chan.NumberofPoints` will return the total number of data points for the combined segments.

**VB Syntax**
```vbnet
data = chan.GetXAxisValues
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>data</code></td>
<td>Variant array to store the data.</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>A Channel (object)</td>
</tr>
</tbody>
</table>

**Return Type**
Variant

**Default**
Not Applicable

**Examples**
```vbnet
Dim varData As Variant
Dim i As Integer
varData = chan.GetXAxisValues
'Print Data
For i = 0 To chan.NumberOfPoints - 1
  Print varData(i)
Next i
```

**C++ Syntax**
```c++
HRESULT GetXAxisValues (VARIANT* xData)
```

**Interface**
IChannel
HasCalType Method

**Description**  Verifies that the Cal Set object contains the error terms required to perform the specified correction (CalType) to an appropriate measurement.

The argument list includes specifiers for up to 3 ports. The number of arguments required depends on the CalType specified. The value for each port is set to 0 if not specified.

**VB Syntax**  
```
check = CalSet.HasCalType (calType, p1, p2, p3)
```

**Variable**  
* **check**  (boolean) - variable to store the returned value
  
  **TRUE (1)** - Cal Set has all of the error terms necessary to apply the specified correction CalType.
  
  **FALSE(0)** - Cal Set DOES NOT have all of the error terms necessary to apply the specified CalType.

* **CalSet**  (object) - A Cal Set object

* **calType**  (enum as naCalType) - type of correction to be applied. Choose from:

<table>
<thead>
<tr>
<th>Caltype</th>
<th>p arguments required</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - naCalType_Response_Open</td>
<td>p1</td>
</tr>
<tr>
<td>1 - naCalType_Response_Short</td>
<td>p1</td>
</tr>
<tr>
<td>2 - *naCalType_Response_Thru</td>
<td>p1 (rcv), p2 (src)</td>
</tr>
<tr>
<td>3 - *naCalType_Response_Thru_And_Isol</td>
<td>p1 (rcv), p2 (src)</td>
</tr>
<tr>
<td>4 - naCalType_OnePort</td>
<td>p1</td>
</tr>
<tr>
<td>5 - naCalType_TwoPort_SOLT</td>
<td>p1, p2</td>
</tr>
<tr>
<td>6 - naCalType_TwoPort_TRL</td>
<td>p1, p2</td>
</tr>
<tr>
<td>7 - naCalType_None</td>
<td>N/A</td>
</tr>
<tr>
<td>8 - naCalType_ThreePort_SOLT</td>
<td>p1, p2, p3</td>
</tr>
<tr>
<td>9 - Custom</td>
<td>N/A</td>
</tr>
<tr>
<td>10 - naCalType_FourPort_SOLT</td>
<td>p1, p2, p3</td>
</tr>
</tbody>
</table>

(port 4 is assumed)

* order of port arguments is significant for these CalTypes

* **p1**  (long) - required. This argument must be specified.
This specifies either:
- the one significant port for an open/short response cal or a 1 port cal.
- or one of the ports involved in a 2, 3, or 4 port cal
- or the receive port for a thru response / thru-isolation cal.

*p2* (long) - required for any CalType involving more than one port

This specifies either:
- one of the ports involved in a 2, 3, or 4 port cal (order independent)
- or the source port for a thru response / thru-isolation cal

*p3* (long) - required for 3 and 4-port cal

This specifies one of the ports involved in a 3 or 4 port cal (order independent)

**Return Type** VARIANT_BOOL

**Default** Not Applicable

**Examples**

```c++
value = CalSet.HasCalType(naCalType_TwoPort_TRL, 1, 2)
```

**C++ Syntax**

```c++
HRESULT HasCalType( tagNACalType, long port1, long port2, long port3, BOOL *pVal);
```

**Interface** ICalSet
### Hold Method

| Description | Puts the channel in Hold - not sweeping.  
See chans.Hold to put ALL channels in hold. |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td>chan.Hold [sync]</td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>chan</td>
<td>A Channel (object)</td>
</tr>
<tr>
<td>[sync]</td>
<td>(boolean) - Optional argument - channel object ONLY. A variable set to either True or False.</td>
</tr>
<tr>
<td>True</td>
<td>program control waits until the channel is in the Hold state.</td>
</tr>
<tr>
<td>False</td>
<td>program control continues immediately. You are not guaranteed the channel is in Hold yet.</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>
| Examples | **wait = True**  
**chan.Hold wait** |
| C++ Syntax | HRESULT Hold(VARIANT_BOOL bWait) |
| Interface | IChannel |
### Hold (channels) Method

**Description**
Places ALL channels in hold mode.

To resume all channels sweeping, use `chans.Resume`. (Must be the same instance of `chans`).

To place a single channel in hold mode, use `channel.Hold` Method.

<table>
<thead>
<tr>
<th>VB Syntax</th>
<th><code>chans.Hold</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
<td><code>chans</code></td>
</tr>
<tr>
<td>(Type)</td>
<td>Channel collection (object)</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

- `chans.Hold`

**C++ Syntax**

```
HRESULT Hold();
```

**Interface**
IChannels2
**Initialize Method**

**Description**
Begins a calibration.

*Note:* `chan` must be the active channel.

**VB Syntax**
```vbnet
obj.Initialize (chan, useCalStorPref)
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj</code></td>
<td>Any of the following:</td>
</tr>
<tr>
<td></td>
<td><code>GuidedCalibration</code> (object)</td>
</tr>
<tr>
<td></td>
<td><code>SMCTYPE</code> (object)</td>
</tr>
<tr>
<td></td>
<td><code>VMCTYPE</code> (object)</td>
</tr>
<tr>
<td><code>chan</code></td>
<td>(Long) Channel number to calibrate.</td>
</tr>
<tr>
<td><code>useCalStorPref</code></td>
<td>(boolean)</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> or <strong>1</strong> - Assignment of Cal Set will be based on the setting of the RemoteCalStoragePreference COM property.</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> or <strong>0</strong> – If the channel currently has a selected Cal Set, the calibration will be stored to that Cal Set. Otherwise, the assignment of Cal Set is based upon the setting of the RemoteCalStoragePreference COM property.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
smc.Initialize(2,True)
```

**C++ Syntax**
```cpp
HRESULT put_Initialize(long channelnumber, VARIANT_BOOL bCalPref);
```

**Interface**
```cpp
IGuidedCalibration
SMCTYPE
VMCTYPE
```
**Write-only**

**Item Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Returns an object from the collection of objects.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notes</td>
<td>The order of objects within a collection cannot be assumed.</td>
</tr>
<tr>
<td></td>
<td>Most, but not all, PNA Collections are '1-based'</td>
</tr>
</tbody>
</table>

**VB Syntax**  
`Object.Item(n)`

**Variable**  
(Type) - Description

**Object**  
Any of the following (collections):

- `CalFactorSegments collection`
- `Cal Sets collection`
- `Channels collection`
- `E5091Testset collection`
- `ExternalTestsets collection`
- `FOM collection`
- `LimitTest collection`
- `Measurements collection`
- `NaWindows collection`
- `PowerLossSegments collection`
- `PowerSensors collection`
- `Segments collection`
- `Traces collection`
- `PowerMeterInterfaces Collection`

*Learn more about collections in the PNA*

**.Item**  
Optional - Item is the default property of a collections object and therefore can be called implicitly. For example, the following two commands are equivalent:

```
Channels.Item(3).Averaging = 1
Channels(3).Averaging = 1
```

**n (variant)** - number of the item in the collection.

**Note:** The Measurements, Traces, and FOM collections allow you to specify the name of the measurement as a string. For example:

```
measCollection("CH_S11_1").InterpolateMarkers
```
Return Type: (Object)

Default: Not Applicable

Examples:
```
For i = 1 to Traces.Count
    Traces.Item(i).YScale = .5dB
Next i
```

C++ Syntax: HRESULT Item(VARIANT index, <interface>** pItem)

Interfaces: All listed above.
LaunchCalWizard Method

**Description**
Launches the Cal Wizard on the PNA and does not return until the Cal Wizard is dismissed.

*Note*: The Cal Wizard operates on the active measurement. Therefore, activate the measurement to be calibrated before launching the Cal Wizard.

**VB Syntax**
```
success = app.LaunchCalWizard(newCS)
```

**Variable (Type)** - Description

- **success** *(boolean)* - variable to store the returned value
  - True - The Cal was completed
  - False - The Cal was canceled without completing the calibration.

- **app** An Application *(object)*

- **newCS** *(boolean)*
  - True - Cal will be performed on a new Cal Set.
  - False - Cal will be performed using the existing Cal Set assigned to the channel. If no Cal Set is found, a new Cal Set will be created.

**Return Type**
Boolean

**Default**
Not Applicable

**Example**
```
dim bSuccess as boolean
dim bNewCalset as boolean
bNewCalSet = false
bSuccess = app.LaunchCalWizard( bNewCalSet)
```

**C++ Syntax**
```
HRESULT LaunchCalWizard(VARIANT_BOOL bCalSuccess)
```

**Interface**
IAplication
LaunchDialog Method

**Description**
Launches the specified dialog box.

**VB Syntax**
```
app.LaunchDialog dialog, [data]
```

**Variable (Type) - Description**
- `app` An Application (object)
- `dialog` (String) Dialog box to launch. Choose from:
  - "SourcePowerCal" [See this dialog.](#)
  - "PowerMeterSettings" [See this dialog.](#)
- `[data]` (Optional argument) Reserved for future use.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
app.LaunchDialog "SourcePowerCal"
```

**C++ Syntax**
```
LaunchDialog( BSTR dialog, [defaultvalue(0)] VARIANT dialogData)
```

**Interface**
IApplication10

[See the PNA Object Model](#)

---

**Last Modified:**
- 1-Jun-2007  Added optional argument
- 16-Feb-2007  MX New topic
### Description
Launches the **Power Meter Settings dialog** on the PNA. Changing certain values from that dialog will change values of the corresponding properties on this COM object.

### VB Syntax

```
pwrCal.LaunchPowerMeterSettingsDialog
```

### Variable

- **pwrCal**
  A **SourcePowerCalibrator** *(object)*

### Return Type
None

### Default
Not Applicable

### Examples

```
powerCalibrator.LaunchPowerMeterSettingsDialog
```

### C++ Syntax

```
HRESULT put_LaunchPowerMeterSettingsDialog();
```

### Interface
**ISourcePowerCalibrator2**
### LoadConfiguration Method

**Description**  
Loads the named configuration onto the specified channel.  
Use [Configurations Method](#) to return the configuration names that are stored on the PNA.

**VB Syntax**  
`pathMgr.LoadConfiguration ch, name`

**Variable**  
**(Type)** - **Description**  

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pathMgr</code></td>
<td>PathConfigurationManager <em>(object)</em></td>
</tr>
<tr>
<td><code>ch</code></td>
<td><em>(Long)</em> Channel number of the configuration to be saved.</td>
</tr>
<tr>
<td><code>name</code></td>
<td><em>(String)</em> Configuration name. &quot;Default&quot; is the default factory configuration.</td>
</tr>
</tbody>
</table>

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`path.LoadConfiguration 2, "myMixer"`

**C++ Syntax**  
`HRESULT LoadConfiguration (long channelNum, BSTR configName );`

**Interface**  
IPathConfigurationManager

---

**Last Modified:**  
14-Dec-2006   MX New topic
LoadENRFile Method

Description: Loads an ENR file from disk into PNA memory. This file is typically provided by the manufacturer of the noise source.

VB Syntax: `enr.LoadENRFile (filename)`

Variable (Type) - Description:

- `enr`: An ENRFile (object)
- `filename`: (String) - Absolute path and filename of the ENR file.

Return Type: Not Applicable

Default: Not Applicable

Examples: See example program

C++ Syntax: `HRESULT LoadENRFile(BSTR filename);`

Interface: IENRFile

Last Modified:

2-Aug-2007 MX New topic
## LoadFile Method

**Description**
Loads a previously-configured mixer attributes file (.mxr)

**VB Syntax**
`mixer.LoadFile (filename)`

**Variable**
- **Type** - Description
  - `mixer`: A IMixer Interface pointer to the Meas object
  - `filename`: (String) Full path, file name, and .mxr extension of the mixer attributes file. Files are typically stored in "C:\Program Files\Agilent\Network Analyzer\Documents".

**Return Type**
String

**Default**
Not Applicable

**Examples**

```vbnet
mixer.LoadFile ("C:\Program Files\Agilent\Network Analyzer\Documents\myMixer.mxr")
```

**C++ Syntax**
`HRESULT put_LoadFile(BSTR newVal)`

**Interface**
IMixer
ManualTrigger Method

Description
Triggers the analyzer when TriggerSetup.Source = naTriggerManual.

Note: An SMC Fixed Output measurement cannot be triggered using this command. For more information, see the example program.

VB Syntax
app.ManualTrigger [sync],[timeout]

Variable (Type) - Description
app  An Application (object)
[sync]  (boolean) - Optional argument.
A variable set to either True or False.
True - The analyzer waits until the trigger is completed to process subsequent commands.
False - Subsequent commands are processed immediately (the default setting).

timeout  (long) - Optional argument.
If sync is true, timeout sets the amount of time the PNA will wait until continuing program execution. Units are milliseconds. A value of -1 (the default setting) causes the PNA to wait indefinitely.
If sync is False, the timeout setting is ignored.

Return Type  Not Applicable
Default  Not Applicable

Examples
' After Manual trigger is executed, the PNA will wait 1 second to continue program execution
Dim wait as Boolean
wait = True
app.ManualTrigger wait, 1000

C++ Syntax
HRESULT ManualTrigger(VARIANT_BOOL bSynchronize, long timeout)

Interface IApplication

Last Modified:
12-Jul-2007  Modified link
MessageText Property

**Description**  
Returns text for the specified eventID

**VB Syntax**  
`app.MessageText, eventID, message`

**Variable**  
(**Type** - **Description**)

- **app**  
  An Application (object)

- **eventID**  
  (enum naEventID) Choose from the list in Working with the Analyzer's Events

- **message**  
  (string) - variable to store the returned message

**Return Type**  
String

**Default**  
Not Applicable

**Examples**  
RFNA.MessageText naEventID_ARRANGE_WINDOW_EXCEED_CAPACITY, message

**C++ Syntax**  
`HRESULT get_MessageText(tagNAEventID msgID, BSTR* message)`

**Interface**  
IAplication
**Description**  A function that returns the Next higher IF Bandwidth value. Use to retrieve the list of available IFBandwidth settings.

**VB Syntax**  `chan.Next_IFBandwidth bw`

**Variable**  
- **(Type) - Description**
  - `chan`  A Channel *(object)*
  - `bw`  *(double)* - The argument that you use to send an IFBandwidth. The function uses this argument to return the Next higher IFbandwidth.

**Return Type**  Double

**Default**  Not Applicable

**Examples**

```vbnet
Public pnbw As Double 'declare variable outside of procedure
pnBW = chan.IFBandwidth 'put the current IFBW in pnBW
chan.Next_IFBandwidth pnBW 'function returns the Next higher IFBandwidth.
chan.IFBandwidth = pnBW 'set IFBW to the Next value
```

**C++ Syntax**  
`HRESULT Next_IFBandwidth (double *pVal)`

**Interface**  IChannel
**NumberOfGroups Method**

**Description**
Sets the number of trigger signals the channel will receive. After the channel has received that number of trigger signals, the channel switches to Hold mode.

**VB Syntax**
```
chan.NumberOfGroups num, sync
```

**Variable (Type) - Description**

- **chan** (A Channel object)
- **num** (long integer) Number of trigger signals the channel will receive. Choose any number between 1 and 2 million.
- **sync** (boolean)
  Variable set to either:
  - **True** - subsequent commands are not processed until the groups are complete. Do not use with manual trigger.
  - **False** - subsequent commands are processed immediately.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
chan.NumberOfGroups 5,False
```

**C++ Syntax**
```
HRESULT NumberOfGroups(long count, VARIANT_BOOL bWait)
```

**Interface**
IChannel
### OpenCalSet Method  Superseded

<table>
<thead>
<tr>
<th>Description</th>
<th>This command is no longer necessary. The CalSet.get... and put... commands that required this command have been replaced,</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Open the Cal Set to read/write a particular <strong>CalType</strong>. Learn more about <a href="#">reading and writing Cal Data using COM</a>.</td>
</tr>
<tr>
<td></td>
<td>This method is a prerequisite to several other Cal Set methods.</td>
</tr>
<tr>
<td></td>
<td>A Cal Set can contain more than one CalType. This method opens the Cal Set and allows access to a particular set of terms. Subsequent commands like <code>getErrorTerm</code> use this information to access the correct error terms in the Cal Set. For example:</td>
</tr>
<tr>
<td></td>
<td><code>cset.OpenCalSet(naCalType_TwoPortSOLT, 3, 2)</code></td>
</tr>
<tr>
<td></td>
<td><code>cset.PutErrorTerm(naDirectivity, 1, 1, Buffer)</code></td>
</tr>
<tr>
<td></td>
<td>The directivity error term for port 1 could belong to any number of caltypes: Full1Port (S11), Full2Port (12), Full2Port (13) or Full3Port (123). The <strong>CalType and port</strong> specifiers in OpenCalSet directs the uploaded directivity term to the correct set of error terms.</td>
</tr>
<tr>
<td></td>
<td>To close the Cal Set, see <a href="#">CloseCalSet</a>.</td>
</tr>
</tbody>
</table>

### VB Syntax

```vbnet
CalSet.OpenCalSet(CalType, p1, p2, p3)
```

### Variable (Type) - Description

- **CalSet (object)** - A Cal Set object

- **CalType (enum as naCalType)** - type of correction to be applied. Choose from:

<table>
<thead>
<tr>
<th>Caltype</th>
<th>p arguments required</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - naCalType_Response_Open</td>
<td>p1</td>
</tr>
<tr>
<td>1 - naCalType_Response_Short</td>
<td>p1</td>
</tr>
<tr>
<td>2 - *naCalType_Response_Thru</td>
<td>p1 (rcv), p2 (src)</td>
</tr>
<tr>
<td>3 - *naCalType_Response_Thru_And_Isol</td>
<td>p1 (rcv), p2 (src)</td>
</tr>
<tr>
<td>4 - naCalType_OnePort</td>
<td>p1</td>
</tr>
<tr>
<td>5 - naCalType_TwoPort_SOLT</td>
<td>p1, p2</td>
</tr>
<tr>
<td>6 - naCalType_TwoPort_TRL</td>
<td>p1, p2</td>
</tr>
<tr>
<td>7 - naCalType_None</td>
<td>N/A</td>
</tr>
<tr>
<td>8 - naCalType_ThreePort_SOLT</td>
<td>p1, p2, p3</td>
</tr>
<tr>
<td>9 - Custom</td>
<td>N/A</td>
</tr>
</tbody>
</table>
10 - naCalType_FourPort_SOLT  

**p1, p2, p3**  
(port 4 is assumed)

* order of port arguments is significant for these CalTypes

**p1** *(long)* - required. This argument must be specified.  
This specifies either:  
- the one significant port for an open/short response cal or a 1 port cal.  
- or one of the ports involved in a 2 or 3 port cal  
- or the **receive** port for a thru response / thru-isolation cal.

**p2** *(long)* - required for any caltype involving more than one port  
This specifies either:  
- one of the ports involved in a 2 or 3 port cal (order independent)  
- or the **source** port for a thru response / thru-isolation cal

**p3** *(long)* - required only for 3 port cal  
This specifies either:  
- one of the ports involved in a 3 port cal (order independent)

**Return Type** None

**Default** Not Applicable

**Examples**  
```c
CalSet.OpenCalSet naCalType_ThreePort_SOLT, 3, 2, 1
```

**C++ Syntax**  
```c
HRESULT OpenCalSet ( naCalType, port1, [optional] port2, [optional] port3);
```

**Interface** ICalSet
Parse Method

**Description**
Allows the use of COM to send a SCPI command. See a C++ example of how to return error information when using this command.

**VB Syntax**
```
scri.Parse ("SCPI command")
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scti</td>
<td>A ScpiStringParser (object)</td>
<td></td>
</tr>
</tbody>
</table>

**SCPI command (string) - Any valid SCPI command**

**Return Type**
String

**Default**
Not Applicable

**Examples**
```
Dim scpi As ScpiStringParser
Set scpi = app.ScpiStringParser
Dim startfreq As Double
startfreq = 100e6
's
scpi.Parse "Sens:Freq:Start " & startfreq'Write

Dim str As String
str = scpi.Parse ("Sens:Freq:Start?")'Read
```

**C++ Syntax**
```
HRESULT Parse(BSTR SCPI_Command, BSTR *pQueryResponse)
```

**Interface**
IScpiStringParser

---

_Last Modified:_

1-Jan-2007 Corrected example
### Preset Method

| Description | **Application Object**: Deletes all traces and windows. In addition, resets the analyzer to factory defined default settings and creates an S11 measurement named "CH1_S11_1" in window 1.  
**Channel Object**: Resets the channel (object) to factory defined default settings. Does NOT delete the current measurements or add a new measurement. |
|---|---|
| **VB Syntax** | `app.Preset`  
`chan.Preset` |
| **Variable** | **(Type)** - **Description** |
| `app` | An Application **(object)** |
| `chan` | A Channel **(object)** |
| **Return Type** | Not Applicable |
| **Default** | Not Applicable |
| **Examples** | `app.Preset` |
| **C++ Syntax** | HRESULT Preset(); |
| **Interface** | IApplication  
IChannel |
Write-only

About Dynamic Range

PreviousIFBandwidth Method

Description
A function that returns the previous IF Bandwidth value. Use to retrieve the list of available IFBandwidth settings.

VB Syntax
`chan.Previous_IFBandwidth bw`

Variable
(Type) - Description

`chan` A Channel (object)

`bw` (double) - The argument that you use to send an IFBandwidth. The function uses this argument to return the previous IFbandwidth.

Return Type
Double

Default
Not Applicable

Examples
Public pnbw As Double 'declare variable outside of procedure
PreBW = chan.IFBandwidth 'put the current IFBW in PreBW
chan.Previous_IFBandwidth PreBW 'function returns the Previous IFBandwidth of the current one.
chan.IFBandwidth = PreBW 'set IFBW to the previous value

C++ Syntax
HRESULT Previous_IFBandwidth (double *pVal)

Interface
IChannel
PrintToFile Method

Description
Saves the screen image to a bitmap file.

VB Syntax
app.PrintToFile filename

Variable (Type) - Description
app An Application (object)
filename (string) Full path, file name, and extension of the screen image file.
Files are typically stored in "C:\Program Files\Agilent\Network Analyzer\Documents". Use one of the following extensions:
  - .bmp - not recommended due to large file size
  - .jpg - not recommended due to poor quality
  - .png - recommended

Return Type Not Applicable

Default Not Applicable

Examples
app.PrintToFile "C:\Program Files\Agilent\Network Analyzer\Documents\myfile.png"

C++ Syntax HRESULT PrintToFile(BSTR bstrFile)

Interface IApplication
PutComplex Method

Description
Puts real and imaginary data into the specified location. This method forces the channel into Hold mode to prevent the input data from being overwritten. Learn more about reading and writing Cal Data using COM.

Data put in the raw data store will be **re-processed** whenever a change is made to the measurement attributes such as format or correction.

Data put in the measurement results store will be **overwritten** by any measurement attribute changes.

See also putNAComplex

VB Syntax
```
measData.putComplex location, numPts, real(), imag(), [format]
```

Variable (Type) - Description

**measData** An IArrayTransfer interface which supports the Measurement object

**location** (enum NADataStore) Where the Data will be put. Choose from:

- 0 - naRawData
- 1 - naCorrectedData
- 2 - naMeasResult
- 3 - naRawMemory
- 4 - naMemoryResult
- 5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

**numPts** (long integer) - Number of data points in the channel

**real()** (single) - Array containing real data values

**imag()** (single) - Array containing imaginary data values

**format** (enum NADataFormat) optional argument - display format of the real and imaginary data. Only used if destination is naMeasResult or naMemoryResult buffer. If unspecified, data is assumed to be in naDataFormat_Polar

- 0 - naDataFormat_LinMag
- 1 - naDataFormat_LogMag
- 2 - naDataFormat_Phase
- 3 - naDataFormat_Polar
- 4 - naDataFormat_Smith
- 5 - naDataFormat_Delay
- 6 - naDataFormat_Real
7 - naDataFormat_Imaginary
8 - naDataFormat_SWR
9 - naDataFormat_PlaceUnwrapped
10 - naDataFormat_InverseSmith
11 - naDataFormat_Kelvin
12 - naDataFormat_Fahrenheit
13 - naDataFormat_Centigrade

Learn more about Data Format.

### Return Type
Not Applicable

### Default
Not Applicable

### Examples
```vba
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement

measData.putComplex naMemoryResult, 201,
real(0), imag(0), naDataFormat_SWR
```

### C++ Syntax
```cpp
HRESULT putComplex(tagNADataStore DataStore, long lNumValues, float* pReal,
float* pImag, tagDataFormat displayFormat)
```

### Interface
IArrrayTransfer

---

Last Modified:
1-Oct-2007 Added temperature formats
### PutDataComplex Method

**Description**

Puts complex data into the specified location. This method forces the channel into Hold mode to prevent the input data from being overwritten.

**VB Syntax**

```vbnet
meas.putDataComplex location, data
```

**Variable (Type) - Description**

- **meas**
  - A measurement (object)

- **location**
  - (enum NADataStore) Where the Data will be put. Choose from:
    - 0 - naRawData
    - 1 - naCorrectedData
    - 2 - naMeasResult
    - 3 - naRawMemory
    - 4 - naMemoryResult
    - 5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using DataToDivisor Method.

- Data put in 0 - naRawData will be re-processed whenever a change is made to the measurement attributes such as format or correction.

- Data put in 2 - naMeasResult will be overwritten by any measurement attribute changes.

- When putting data into 3 - naRawMemory:
  1. Put the analyzer in hold mode
  2. Call DataToMemory to initialize a memory buffer
  3. Call putDataComplex(naRawMemory, data)

This ensures that the memory buffer is appropriately initialized before receiving new data.

- **data**
  - (variant) - A two-dimensional variant array.

**Note:** All buffers except naMeasResult and naMemoryResult require Complex data

**Return Type**

Not Applicable

**Default**

Not Applicable

**Examples**

```vbnet
' Put 201 points worth of raw (complex) data into the measurement
' Note that an array of complex numbers is represented by a 2-D
```
array where the first rank is the number of points, and the 2nd rank is always size 2 (max index 1) representing the Real and Imag parts of the complex number.

' complex array of data (2nd dimension of size 2 represents Re/Im
Dim data(200,1) 
For i = 0 to 200
' Set Real part of data point i
data(i,0) = i/200;
' Set Imag part of data point i
data(i,1) = i/200;
Next
app.ActiveMeasurement.putDataComplex naRawData, data

C++ Syntax HRESULT putDataComplex(tagNADataStore DataStore, VARIANT complexData)
Interface IMeasurement
**PutENRData Method**

**Description**
Write ENR calibration data to PNA memory. All of the frequency and ENR data must be sent at the same time.

**VB Syntax**

```
enr.PutENRData (vData)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enr</td>
<td>An ENRFile (object)</td>
<td></td>
</tr>
<tr>
<td>vData</td>
<td>(Variant array) - ENR data. Frequency value in Hz, followed by corresponding ENR value in dB. Enter as many data pairs as necessary.</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
See example program

**C++ Syntax**

```c++
HRESULT PutENRData(VARIANT vdata);
```

**Interface**
IENRFile

---

Last Modified:

2-Aug-2007 MX New topic
PutErrorTerm Method - Superseded

**Note:** This command is replaced by `PutErrorTermByString`

Puts variant error term data into the error-correction buffer.

Learn about reading and writing Calibration data.

**VB Syntax**

```
cal.putErrorTerm(term, rcv, src, data)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cal</code></td>
<td><code>object</code></td>
<td>A Calibrator</td>
</tr>
<tr>
<td><code>term</code></td>
<td><code>enum As NaErrorTerm</code></td>
<td></td>
</tr>
<tr>
<td><code>rcv</code></td>
<td><code>long integer</code></td>
<td>Receiver Port</td>
</tr>
<tr>
<td><code>src</code></td>
<td><code>long integer</code></td>
<td>Source Port</td>
</tr>
<tr>
<td><code>data</code></td>
<td><code>variant</code></td>
<td>Error term data in a two-dimensional array (0:1, 0:numpts-1).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>To get this Error Term</th>
<th>Specify these parameters:</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>term</code></td>
<td></td>
</tr>
<tr>
<td><code>rcv</code></td>
<td></td>
</tr>
<tr>
<td><code>src</code></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Error Term</th>
<th><code>term</code></th>
<th><code>rcv</code></th>
<th><code>src</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fwd Directivity</td>
<td>naET_Directivity_Isolation</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Rev Directivity</td>
<td>naET_Directivity_Isolation</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Isolation</td>
<td>naET_Directivity_Isolation</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Rev Isolation</td>
<td>naET_Directivity_Isolation</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Source Match</td>
<td>naErrorTerm_Match</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Rev Source Match</td>
<td>naErrorTerm_Match</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Load Match</td>
<td>naErrorTerm_Match</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Rev Load Match</td>
<td>naErrorTerm_Match</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Rev Reflection Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Fwd Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Rev Trans Tracking</td>
<td>naErrorTerm_Tracking</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
Fwd Trans Tracking

naErrorTerm_Tracking

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Not Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**Examples**

```vba
Dim varError As Variant
varError = cal.putErrorTerm (naErrorTerm_Tracking, 2, 1, VarData)
```

**C++ Syntax**

```
HRESULT putErrorTerm(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, VARIANT varData)
```

**Interface**

ICalibrator
PutErrorTerm Method  Superseded

**Description**
This command is replaced with **PutErrorTermByString**

Puts error term data into the Cal Set.

Learn more about [Reading and Writing Cal Data](#).

See examples of [Reading](#) and [Writing](#) Cal Set Data.

**VB Syntax**
```
CalSet.putErrorTerm (term, rcv, src, data)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CalSet</strong></td>
<td>(Object)</td>
<td>A CalSet Object</td>
</tr>
<tr>
<td><strong>term</strong></td>
<td>(enum As NaErrorTerm2)</td>
<td>Error Term. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - naET_Directivity   (src = rcv)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - naET_SourceMatch    ( src = rcv)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - naET_ReflectionTracking (src = rcv)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - naET_TransmissionTracking (src rcv)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 - naET_LoadMatch      (src rcv)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 - naET_Isolation      (src rcv)</td>
</tr>
<tr>
<td><strong>rcv</strong></td>
<td>(long integer)</td>
<td>- Receiver Port</td>
</tr>
<tr>
<td><strong>src</strong></td>
<td>(long integer)</td>
<td>- Source Port</td>
</tr>
<tr>
<td><strong>data</strong></td>
<td>(variant)</td>
<td>Error term data in a two-dimensional array (0:1, 0:numpts-1). The data must be complex pairs.</td>
</tr>
</tbody>
</table>

**Note:** See also **PutErrorTermComplex** on the ICalData2 interface to avoid using the variant data type.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
[See an Example](#)

**C++ Syntax**
```
HRESULT putErrorTerm(tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, VARIANT varData)
```

**Interface**
ICalSet
PutErrorTermByString

Description
Puts error term data into the Cal Set.
Learn more about Reading and Writing Cal Data
See examples of Reading and Writing Cal Set Data

VB Syntax

\[
calSet.PutErrorTermByString(errorName, vdata)
\]

Variable

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>calSet</td>
<td>(Object) A CalSet Object</td>
</tr>
<tr>
<td>errorName</td>
<td>(String) The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be &quot;Directivity(3,3)&quot;. To determine the string names of error terms, see GetErrorTermList2.</td>
</tr>
</tbody>
</table>
| vdata   | (Variant) This data array is usually two dimensional. Each element is a type single. The two elements represent the real and imaginary parts of a complex pair.  
Note: This structure is compatible with scripting clients who can only use variants. For alternative methods that use typed arrays, see ICalData3. |

Return Type

Not Applicable

Default

Not Applicable

Examples

See an Example

C++ Syntax

HRESULT PutStandardByString( BSTR bufferName, VARIANT vdata)
**PutErrorTermComplex Method Superseded**

**Description**
Note: This command is replaced by `PutErrorTermComplexByString`

Puts error term data into the error-correction data buffer. Learn more about reading and writing Cal data using COM.

**VB Syntax**
```
data.putErrorTermComplex term, rcv, src, numPts, real(), imag()
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>data</code></td>
<td>An ICalData pointer to the Calibrator object</td>
<td></td>
</tr>
<tr>
<td><code>term</code></td>
<td>(enum NAErrorTerm) - The error term to be retrieved. Choose from:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>naErrorTerm_Directivity_Isolation</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>naErrorTerm_Match</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>naErrorTerm_Tracking</code></td>
</tr>
<tr>
<td><code>rcv</code></td>
<td>(long integer) - Receiver Port</td>
<td></td>
</tr>
<tr>
<td><code>src</code></td>
<td>(long integer) - Source Port</td>
<td></td>
</tr>
<tr>
<td><code>numPts</code></td>
<td>(long integer) - number of data points in the array</td>
<td></td>
</tr>
<tr>
<td><code>real()</code></td>
<td>(single) - array containing the real part of the calibration data. One-dimensional: the number of data points.</td>
<td></td>
</tr>
<tr>
<td><code>imag()</code></td>
<td>(single) - array containing the imaginary part of the calibration data. One-dimensional: the number of data points.</td>
<td></td>
</tr>
</tbody>
</table>

**To get this**

<table>
<thead>
<tr>
<th>Error Term</th>
<th>Specify these parameters:</th>
<th><code>term</code></th>
<th><code>rcv</code></th>
<th><code>src</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fwd Directivity</td>
<td><code>naET_Directivity Isolation</code></td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Directivity</td>
<td><code>naET_Directivity Isolation</code></td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Isolation</td>
<td><code>naET_Directivity Isolation</code></td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Isolation</td>
<td><code>naET_Directivity Isolation</code></td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Source Match</td>
<td><code>naErrorTerm_Match</code></td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Source Match</td>
<td><code>naErrorTerm_Match</code></td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Fwd Load Match</td>
<td><code>naErrorTerm_Match</code></td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Rev Load Match</td>
<td><code>naErrorTerm_Match</code></td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
**Fwd Reflection Tracking**
- **naErrorTerm_Tracking**: 1 1

**Rev Reflection Tracking**
- **naErrorTerm_Tracking**: 2 2

**Fwd Trans Tracking**
- **naErrorTerm_Tracking**: 2 1

**Rev Trans Tracking**
- **naErrorTerm_Tracking**: 1 2

**Fwd Trans Tracking**
- **naErrorTerm_Tracking**: 2 1

---

**Return Type**
- Not Applicable

**Default**
- Not Applicable

**Examples**

```
Dim eData As ICalData
Set eData = chan.Calibrator
eData.putErrorTermComplex naErrorTerm_Directivity_Isolation, 1, 1, 201, rel(0), img(0)
```

**C++ Syntax**

```
HRESULT putErrorTermComplex(tagNAErrorTerm ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float* pImag)
```

**Interface**
- ICalData
**PutErrorTermComplex Method**  
**Superseded**

**Description**  
This command is replaced with **PutErrorTermComplexByString**

Puts error term data into the Cal Set.

Learn more about [Reading and Writing Cal Data](#)

See examples of [Reading](#) and [Writing](#) Cal Set Data

**VB Syntax**

```vbnet
data.putErrorTermComplex term, rcv, src, numPts, real(), imag()
```

**Variable**  
**(Type) - Description**

- `data`  
  An `ICalData2` pointer to a Cal Set object

- `term`  
  (enum `NAErrorTerm2`) - The error term to be written. Choose from:
  - 0 - `naET_Directivity`
  - 1 - `naET_SourceMatch`
  - 2 - `naET_ReflectionTracking`
  - 3 - `naET_TransmissionTracking`
  - 4 - `naET_LoadMatch`
  - 5 - `naET_Isolation`

- `rcv`  
  (long) - Receiver Port

- `src`  
  (long) - Source Port

- `numPts`  
  (long) - number of data points in the real and imaginary arrays.

- `real()`  
  (single) - array containing the real part of the calibration data. One-dimensional: the number of data points.

- `imag()`  
  (single) - array containing the imaginary part of the calibration data. One-dimensional: the number of data points.

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**

```vbnet
dim eData As ICalData2
set eData = app.GetCalManager.Cal Sets.Item(1)
eData.putErrorTermComplex naET_LoadMatch, 1, 2, numpts, real(0), img(0)
```

**C++ Syntax**

```c++
HRESULT putErrorTermComplex(tagNAErrorTerm2 ETerm, long ReceivePort, long SourcePort, long* pNumValues, float* pReal, float plmag)
```
Interface  ICalData2
PutErrorTermComplexByString Method

**Description**

Puts error term data into the Cal Set.

Learn more about Reading and Writing Cal Data

See examples of Reading and Writing Cal Set Data

**VB Syntax**

```vbnet
ICalData3.PutErrorTermComplexByString errorName, InumPoints, real(0), imag(0)
```

**Variable**

- **ICalData3**
  An ICalData3 pointer to a Cal Set object.

- **errorName**
  (String) The string name used to identify a particular error term in the Cal Set. An example string for port 3 directivity in a full 2 port cal might be "Directivity(3,3)". To determine the string names of error terms, see GetErrorTermList2.

- **InumPoints**
  (Long) The number of data points in the real and imaginary arrays.

- **real**
  (Single) The real component of the complex data.

- **imag**
  (Single) The imaginary component of the complex data.

**Note**: The size of the real and imaginary arrays should be the same.

**Return Value**

Not Applicable

**Default**

Not Applicable

**Examples**

See example

**C++ Syntax**

```c++
HRESULT PutErrorTermComplexByString( BSTR bufferName, long InumPoints, float* real, float* imag);
```

**Interface**

ICalData3
### PutScalar Method

**Description**
Puts Scalar data in the Measurement Result buffer. The putScalar array is not processed by the analyzer; it is just displayed. Any change to the measurement state (changing the format, for example) will cause the putScalar data to be overwritten with the data processed from the raw data buffer.

**VB Syntax**
```
measData.putScalar, format, numPts, data
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>measData</td>
<td>An IArrayTransfer interface which supports the Measurement object.</td>
<td></td>
</tr>
<tr>
<td>format</td>
<td>(enum NADataFormat)</td>
<td>Format of the data. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naDataFormat_LinMag</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 - naDataFormat_LogMag</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 - naDataFormat_Phase</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 - naDataFormat_Polar</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 - naDataFormat_Smith</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 - naDataFormat_Delay</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6 - naDataFormat_Real</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7 - naDataFormat_Imaginary</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 - naDataFormat_SWR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9 - naDataFormat_PhaseUnwrapped</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 - naDataFormat_InverseSmith</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11 - naDataFormat_Kelvin</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12 - naDataFormat_Fahrenheit</td>
<td></td>
</tr>
<tr>
<td></td>
<td>13 - naDataFormat_Centigrade</td>
<td></td>
</tr>
</tbody>
</table>

Learn more about Data Format.

**Note:** Smith, InverseSmith, and Polar formats are not allowed.

<table>
<thead>
<tr>
<th>numPts</th>
<th>(integer) - Number of values. Usually the number of points in the trace (chan.NumberOfPoints).</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>(single) - A one-dimensional array of Scalar data matching the number of points in the current measurement.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable
Examples

Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
measData.putScalar naDataFormat_LogMag, 201, dScalar(0)

C++ Syntax

HRESULT putScalar(tagDataFormat eFormat, long lNumValues, float* pArrayOfScalar)

Interface

IArrayTransfer

Last Modified:

1-Oct-2007  Added temperature formats
**PutNAComplex Method**

**Description**

Puts complex data into the specified location. This method forces the channel into Hold mode to prevent the input data from being overwritten. The data is processed and displayed.

Data put in the naRawData store will be **re-processed** whenever a change is made to the measurement attributes such as format or correction.

Data put in the naMeasResult store will be **overwritten** by any measurement attribute changes (such as moving a marker).

**Note:** This method uses NAComplex which is a user-defined data type. If you cannot or prefer not to use this data type, use the `putComplex` method.

**VB Syntax**

```vb
measData.putNAComplex location, numPts, data, [format]
```

**Variable (Type) - Description**

- **measData**
  - An IArrayTransfer interface which supports the Measurement object

- **location**
  - (enum NADataStore) Where the Data will be put. Choose from:
    - 0 - naRawData
    - 1 - naCorrectedData
    - 2 - naMeasResult
    - 3 - naRawMemory
    - 4 - naMemoryResult
    - 5 - naDivisor - When reading data from, or writing data to, the normalization divisor, you must first create a divisor trace using `DataToDivisor Method`.

- **numPts**
  - (long integer) - Number of data points in the channel

- **data**
  - (NAComplex) - A one-dimensional array of Complex data matching the number of points in the current measurement.

- **format**
  - (enum NADataFormat) - Optional argument. Format of the data. If unspecified, naDataFormat_Polar is assumed. Only used when the destination store is naMeasResult or naMemoryResult.
    - 0 - naDataFormat_LinMag
    - 1 - naDataFormat_LogMag
    - 2 - naDataFormat_Phase
    - 3 - naDataFormat_Polar
    - 4 - naDataFormat_Smith
    - 5 - naDataFormat_Delay
    - 6 - naDataFormat_Real
<table>
<thead>
<tr>
<th>Data Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>naDataFormat_Imaginary</td>
<td>Imaginary data format</td>
</tr>
<tr>
<td>naDataFormat_SWR</td>
<td>Short Wave Ratio</td>
</tr>
<tr>
<td>naDataFormat_PhaseUnwrapped</td>
<td>Phase unwrapped data format</td>
</tr>
<tr>
<td>naDataFormat_InverseSmith</td>
<td>Inverse Smith data format</td>
</tr>
<tr>
<td>naDataFormat_Kelvin</td>
<td>Kelvin data format</td>
</tr>
<tr>
<td>naDataFormat_Fahrenheit</td>
<td>Fahrenheit data format</td>
</tr>
<tr>
<td>naDataFormat_Centigrade</td>
<td>Centigrade data format</td>
</tr>
</tbody>
</table>

Learn more about Data Format.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
Dim measData As IArrayTransfer
Set measData = app.ActiveMeasurement
measData.putNAComplex naMemoryResult, 201, dRawComplex(0)

**C++ Syntax**
HRESULT putNAComplex(tagNADataStore DataStore, long lNumValues, TsComplex* pArrayOfComplex, tagDataFormat displayFormat)

**Interface**
IArrayTransfer

---

Last Modified:
1-Oct-2007  Added temperature formats
Write-only

put_Output Method

**Description**
Writes a TTL HI or TTL Low to output pins 3 or 4 of the Material Handler IO connector.
Each pin also has a latched output which is written to with USER. With the latched (USER) outputs, the value is not applied to the associated pin until a positive edge is detected at INPUT1 (pin 2).

**VB Syntax**
```
handlerIo.put_Output (pin) = value
```

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>handlerIo</strong></td>
<td>(object) - A HandlerIO object</td>
</tr>
<tr>
<td><strong>pin</strong></td>
<td>(enum as NAMatHandlerOutput) - pin to write data to. Choose from:</td>
</tr>
<tr>
<td></td>
<td>naOutput1 - (0) - pin3</td>
</tr>
<tr>
<td></td>
<td>naOutput1User (1) - pin3 latched (applied to pin 3 on positive edge of Input1-pin2)</td>
</tr>
<tr>
<td></td>
<td>naOutput2 (2) - pin4</td>
</tr>
<tr>
<td></td>
<td>naOutput2User (3) - pin4 latched (applied to pin 4 on positive edge of Input1-pin2)</td>
</tr>
<tr>
<td><strong>value</strong></td>
<td>(Variant) Value to write to the selected pin. Choose from</td>
</tr>
<tr>
<td></td>
<td>0 - TTL LOW</td>
</tr>
<tr>
<td></td>
<td>1 - TTL HIGH</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
0

**Examples**
```
handlerIo.put_Output(naOutput1)= 1
```

**C++ Syntax**
```
HRESULT put_Output ( tagNAMatHandlerOutput Output, VARIANT Data );
```

**Interface**
IHWMaterialHandlerIO
**put_OutputVoltage Method**

**Description**
- **E836x and PNA-L:** Sets voltages on the DAC/Analog Output 1|2 of the Auxiliary IO connector.
- **PNA-X:** Sets voltage on the Power I/O connector AnalogOut1|2.

Read output voltages using **get OutputVoltage Method**.

**VB Syntax**

```vb
AuxIO.put_OutputVoltage output, voltage
```

**Variable**

- **(Type) - Description**
  - **AuxIO** *(object)* - A Hardware Auxiliary Input / Output object
  - **output** *(variant)* - Number of the output DAC to write voltage to. Choose from:
    1. Output 1 (Aux I/O pin 3) and (Power I/O pin 3)
    2. Output 2 (Aux I/O pin 2) and (Power I/O pin 4)
  - **voltage** *(double)* - Voltage to write to the output DAC. Choose a voltage from -10 to 10

**Return Type**

None

**Default**

None

**Examples**

```vb
HWAuxIO.put_OutputVoltage 1, 9 'set Analog Out1 to +9v
```

**C++ Syntax**

```cpp
HRESULT put_OutputVoltage (VARIANT Output, double Voltage);
```

**Interface**

IHWAuxIO

---

Last Modified:

10-Jul-2007  Added PNA-X capability
### put_OutputVoltageMode Method

**Description**
This command sets the mode of the selected "Analog Out" line on the Auxiliary IO connector and Power I/O connector. The modes give the user the option to have the requested voltage applied immediately or not until the sweep is done. To read the mode on each output use get_OutputVoltageMode Method.

**VB Syntax**
```
auxIo.put_OutputVoltageMode (output, mode)
```

**Variable**
- **Type** - Description
  - **auxIo** *(Object)* An AuxIO object
  - **output** Analog Output to receive mode setting. Choose from 1 or 2
  - **mode** *(enum NAOutputVoltageMode)*
    - **naWaitEOS** - While in this mode any voltage changes sent to the selected analog out will only get applied to the output between sweeps.
    - **naNoWait** - While in this mode any voltage changes sent to the selected analog out will occur right away without waiting until the end of a sweep, the voltage gets applied immediately.

**Return Type**
NAOutputVoltageMode

**Default**
naWaitEOS

**Examples**
```
auxIo.put_OutputVoltageMode 1, naWaitEOS 'Write
```

**C++ Syntax**
```
HRESULT put_OutputVoltageMode(VARIANT Output, tagNAOutputVoltageMode dNewMode);
```

**Interface**
IHWAuxIO

---

**Last Modified:**
10-Jul-2007  Added PNA-X capability
**Description**

Writes a value to the specified port. Use the `get_Port` Method to read the settings from the "readable" ports (C, D, E).

**VB Syntax**

```vb
handlerIo.put_Port (port) = value
```

**Variable**

- **(Type) - Description**
  - `handlerIo` (object) - A HandlerIO object
  - `port` (enum as `NAMatHandlerPort`) - port to put data into. Choose from:
    - `naPortA` - (0)
    - `naPortB` - (1)
    - `naPortC` - (2)
    - `naPortD` - (3)
    - `naPortE` - (4)
    - `naPortF` - (5)
    - `naPortG` - (6)
    - `naPortH` - (7)
  - `value` - The number of the data bits to set. The following table shows what the `value` represents:

**Note:** When writing to port G, port C must be set to output mode
When writing to port H, both port C and port D must be set to output mode. Use Port Mode Property

<table>
<thead>
<tr>
<th>Port</th>
<th>Max allow. &lt;num&gt;</th>
<th>MSB...</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>255</td>
<td>A7...A0</td>
<td>Write-only</td>
</tr>
<tr>
<td>B</td>
<td>255</td>
<td>B7...B0</td>
<td>Write-only</td>
</tr>
<tr>
<td>C</td>
<td>15</td>
<td>C3...C0</td>
<td>Read-Write</td>
</tr>
<tr>
<td>D</td>
<td>15</td>
<td>D3...D0</td>
<td>Read-Write</td>
</tr>
<tr>
<td>E</td>
<td>255</td>
<td>D3...D0 + C3...C0</td>
<td>Read-Write</td>
</tr>
<tr>
<td>F</td>
<td>65535</td>
<td>B7...B0 + A7...A0</td>
<td>Write-only</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>G</td>
<td>1048575</td>
<td>C3...C0 + B7...B0 + A7...A0</td>
<td>Write-only</td>
</tr>
<tr>
<td>H</td>
<td>16777215</td>
<td>D3...D0 + C3...C0 + B7...B0 + A7...A0</td>
<td>Write-only</td>
</tr>
</tbody>
</table>
### put_PortCData Method

**Description**  
Writes a 4-bit value to Port C on the Aux I/O connector (pins 22-25) and the Material Handler IO (pins 21-24 Anritsu) - (pins 22-25 Avantest).

**Note:** These lines are connected to both the Handler IO and Aux IO in the PNA. Therefore, this command will affect both of these connectors in the same way.

**VB Syntax**  
`AuxIO.put_PortCData num`

**Variable**  
- **(Type)** - Description
  - `AuxIO` (object) - A Hardware Auxiliary Input / Output object
  - `num` (variant) - 4 bit binary value. Choose from 0-15

**Return Type**  
None

**Default**  
None

**Examples**  
`HWAuxIO.put_PortCData 15  'If Positive Logic, Port C lines C0, C1, C2, C3 go High. If Negative Logic, they go Low.`

**C++ Syntax**  
`HRESULT put_PortCData( VARIANT Data );`

**Interface**  
`IHWAuxIO`
**PutDataScalar Method**

**Description**

Puts formatted variant scalar data into the measurement result buffer. The data will be immediately processed and displayed. Subsequent changes to the measurement state will be reflected on the display.

Always precede this command by setting the format on the measurement to be consistent with the format of the data being sent to the analyzer. In this way, the display annotation will be correct.

Execution of this command does not change the display format.

**VB Syntax**

`meas.putDataScalar format, data`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meas</td>
<td>A measurement (object)</td>
</tr>
<tr>
<td>format</td>
<td>(enum NADataFormat) Format of the data. This value is presently ignored by the PNA. Data is always presented in the current format. Choose from:</td>
</tr>
<tr>
<td>0</td>
<td>naDataFormat_LinMag</td>
</tr>
<tr>
<td>1</td>
<td>naDataFormat_LogMag</td>
</tr>
<tr>
<td>2</td>
<td>naDataFormat_Phase</td>
</tr>
<tr>
<td>3</td>
<td>naDataFormat_Polar</td>
</tr>
<tr>
<td>4</td>
<td>naDataFormat_Smith</td>
</tr>
<tr>
<td>5</td>
<td>naDataFormat_Delay</td>
</tr>
<tr>
<td>6</td>
<td>naDataFormat_Real</td>
</tr>
<tr>
<td>7</td>
<td>naDataFormat_Imaginary</td>
</tr>
<tr>
<td>8</td>
<td>naDataFormat_SWR</td>
</tr>
<tr>
<td>9</td>
<td>naDataFormat_PhaseUnwrapped</td>
</tr>
<tr>
<td>10</td>
<td>naDataFormat_InverseSmith</td>
</tr>
<tr>
<td>11</td>
<td>naDataFormat_Kelvin</td>
</tr>
<tr>
<td>12</td>
<td>naDataFormat_Fahrenheit</td>
</tr>
<tr>
<td>13</td>
<td>naDataFormat_Centigrade</td>
</tr>
</tbody>
</table>

**Notes:**

- The `getData` (variant) method includes a "format" argument, which allows scalar (one-dimensional) data. To put data back into the "raw" data buffer using this (putDataComplex) method, specify **Polar** format when using the `getData` method.

- **Phase** format accepts data in radians (not degrees) and displays in degrees. To convert to degrees: $\text{radians} \times (57.29577951308233) = \text{degrees}$. The `getData` method returns degrees if the request is for phase data.
**data** (variant) - A 1-dimension array of single precision floating point numbers.

**Return Type** Not Applicable

**Default** Not Applicable

**Examples**

```vbnet
' Put 201 points worth of scalar data into the measurement
' 200 is max index, so 0 to 200 is 201 points
Dim data(200) ' array of 201 (scalar) data points
' Fill the array
For i = 0 to 200
    data(i) = i/200;
Next
app.ActiveMeasurement.putDataScalar 0, data
```

**C++ Syntax**

```c++
HRESULT putDataScalar(tagNADataStore DataStore, VARIANT scalarArray)
```

**Interface** IMeasurement

---

**Last Modified:**

1-Oct-2007 Added temperature formats
PutShortcut Method

**Description**
Defines a Macro (shortcut) file in the analyzer. This command links a file name and path to the Macro file. The file must be put in the PNA at the location indicated by this command.

**VB Syntax**
```
app.PutShortcut index,title,path
```

**Variable**
- **app** *(Type)* - Application (object)
- **index** *(long)* - Number of the macro to be stored in the analyzer. If the index number already exists, the existing macro is replaced with the new macro.
- **title** *(string)* - The name to be assigned to the macro
- **path** *(string)* - Full path, file name, and extension of the existing macro "executable" file.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
app.PutShortcut 1,"Test","C:\Automation\MyTest.vbs"
```

**C++ Syntax**
```
HRESULT PutShortcut(long Number, BSTR title, BSTR pathname)
```

**Interface**
IApplication
## putSourcePowerCalDataEx Method

### Description

**Note:** This method replaces `putSourcePowerCalData Method`

Inputs source power calibration data (as variant data type) to this channel for a specific source port.

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the [Get X-Axis Values](#) command to return the X-axis values in the displayed order.

The calibration is not valid if the current number of points on the channel is not equal to the number of values that were input.

**Note:** This method sends variant data which is less efficient than methods available on the ISourcePowerCalData interface.

### VB Syntax

```vbnet
chan.putSourcePowerCalDataEx buffer, sourcePort, data
```

### Variable (Type) - Description

- **chan** *(object)* – A Channel object
- **buffer** *(enum NASourcePowerCalBuffer)* - The source power cal data buffer to write to.
  - 0 - `naCorrectionValues` This is the only data buffer currently available.
- **sourcePort** *(long integer)* – The source port for which calibration data is being requested.
  Use [GetPortNumber](#) to return the port number of a source that only has a string name, such as an External Source.
- **data** *(variant)* – Array of source power cal data being input.

### Return Type

None

### Default

Not Applicable

### Examples

```vbnet
chan.putSourcePowerCalDataEx naCorrectionValues, 1, varData
```

### C++ Syntax

```c++
HRESULT putSourcePowerCalDataEx(tagNASourcePowerCalBuffer bufSelect, long sourcePort, VARIANT varData);
```

### Interface

IChannel4

---

**Last Modified:**

27-Jun-2007   Updated for PNA-X source port names
**putSourcePowerCalDataScalarEx Method**

**Description**

*Note: This method replaces putSourcePowerCalDataScalar Method*

Inputs source power calibration data (as scalar values) to this channel for a specific source port.

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the Get X-Axis Values2 command to return the X-axis values in the displayed order.

**VB Syntax**

```vbnet
chanData.putSourcePowerCalDataScalarEx buffer, sourcePort, numValues, data
```

**Variable**

*(Type) - Description*

- **chanData** *(interface)* – An ISourcePowerCalData2 interface on the Channel (object)
- **buffer** *(enum NASourcePowerCalBuffer)* – The source power cal data buffer to write to.
  - 0 - naCorrectionValues This is the only buffer currently available.
- **sourcePort** *(long integer)* – The source port for which calibration data is being input.
  Use GetPortNumber to return the port number of a source that only has a string name, such as an External Source.
- **numValues** *(long integer)* – Number of data values being input.
  *Note: If this does not equal the current number of points on the channel, the calibration will not be valid.*
- **data** *(single)* – Array of source power cal data being input.

**Return Type**

None

**Default**

Not Applicable

**Examples**

```vbnet
Dim chanData As ISourcePowerCalData2
Set chanData = app.ActiveChannel
chanData.putSourcePowerCalDataScalarEx naCorrectionValues, 1, 201, scalarCalValues(0)
```

**C++ Syntax**

```c
HRESULT putSourcePowerCalDataScalarEx(tagNASourcePowerCalBuffer bufSelect, long sourcePort, long numValues, float *pData);
```

**Interface**

ISourcePowerCalData2
Write-only

PutStandard Method  Superseded

Description
This command is replaced with PutStandardByString

Puts standard acquisition data into the Cal Set.
Learn more about Reading and Writing Cal Data
See examples of Reading and Writing Cal Set Data.

VB Syntax
CalSet.putStandard class, rcv, src, data

Variable  (Type)  -  Description

CalSet  (object)  -  A Cal Set object

class  (enum NACalClass)  Standard. Choose from:

1 - naClassA
2 - naClassB
3 - naClassC
4 - naClassD
5 - naClassE
6 - naReferenceRatioLine
7 - naReferenceRatioThru

SOLT Standards

1 - naSOLT_Open
2 - naSOLT_Short
3 - naSOLT_Load
4 - naSOLT_Thru
5 - naSOLT_Isolation

TRL Standards

1 - naTRL_Reflection
2 - naTRL_Line_Reflection
3 - naTRL_Line_Tracking
4 - naTRL_Thru
5 - naTRL_Isolation
rcv (long) - Receiver Port

src (long) - Source Port

data (variant) Error term data in a two-dimensional array (0:1, 0:numpts-1). The data must be complex pairs.

Note: See also Put Standard Complex on the ICalData2 interface to avoid using the variant data type.

Return Type Not Applicable

Default Not Applicable

Examples See an Example

C++ Syntax HRESULT putStandard(tagNACalClass stdclass, long ReceivePort, long SourcePort, VARIANT varData)

Interface ICalSet
## PutStandardByString

**Description**

Put standard acquisition data into the Cal Set.

Learn more about **Reading and Writing Cal Data**

See examples of **Reading and Writing Cal Set Data**.

### VB Syntax

```vbnet
PutStandardByString(stdName, vdata)
```

### Variable Description

- **stdName**
  - **Type**: String
  - **Description**: The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be “S11C(3,3)”. (String) The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be “S11C(3,3)”.  

- **vdata**
  - **Type**: Variant
  - **Description**: The variant containing a safearray of variants. This data is usually two dimensional.  

**Note**: The vdata array is a safearray of variants wrapped in a variant. This structure is compatible with scripting clients who can only use variants. For alternative methods that used typed arrays, see `ICalData3`.

### Return Type

**Not Applicable**

### Default

**Not Applicable**

### Examples

See an Example

### C++ Syntax

```cpp
HRESULT PutStandardByString(BSTR bufferName, VARIANT vardata);
```

### Interface

`ICalSet2`

---

1699

1699
PutStandardComplex Method  Superseded

**Description**
This command is replaced with `PutStandardComplexByString`.
Puts standards acquisition data into the Cal Set.
Learn more about [Reading and Writing Cal Data](#).
See examples of [Reading](#) and [Writing](#) Cal Set Data.

**VB Syntax**

```vbnet
ICalData2.putStandardComplex class, rcv, src, numPts, real(), imag()
```

**Variable (Type) - Description**

- **ICalData2**
  - An `ICalData2` pointer to the Cal Set object

- **class**
  - (enum `NACalClass`) Standard. Choose from:
    - 1 - `naClassA`
    - 2 - `naClassB`
    - 3 - `naClassC`
    - 4 - `naClassD`
    - 5 - `naClassE`
    - 6 - `naReferenceRatioLine`
    - 7 - `naReferenceRatioThru`

**SOLT Standards**

- 1 - `naSOLT_Open`
- 2 - `naSOLT_Short`
- 3 - `naSOLT_Load`
- 4 - `naSOLT_Thru`
- 5 - `naSOLT_Isolation`

**TRL Standards**

- 1 - `naTRL_Reflection`
- 2 - `naTRL_Line_Reflection`
- 3 - `naTRL_Line_Tracking`
- 4 - `naTRL_Thru`
- 5 - `naTRL_Isolation`
rcv (long) - Receiver Port

src (long) - Source Port

numPts (long) - The number of data points in the real and imaginary arrays.

real() (single) - one-dimensional array containing the real part of the acquisition data. (0:points-1)

imag() (single) - one-dimensional array containing the imaginary part of the acquisition data. (0:points-1)

Return Type Not Applicable

Default Not Applicable

Examples
Dim sdata As ICalData2
Set sdata = calmanager.CreateCal Set( 1 )
sdata.putStandardComplex naSOLT_Open, 1, 1, numpts, rel(0), img(0)

C++ Syntax
HRESULT putStandardComplex(tagNACalClass stdclass, long ReceivePort, long SourcePort, long lNumValues, float* pReal, float* pImag)

Interface ICalData2
PutStandardComplexByString

Description  Puts standard acquisition data into the Cal Set.
Learn more about Reading and Writing Cal Data
See examples of Reading and Writing Cal Set Data.

VB Syntax  

```
ICalData3.PutStandardComplexByString(stdName, lnumpoints , real(o) , imag(0))
```

Variable  

- **ICalData3**  An ICalData3 pointer to a Cal Set object.
- **stdName**  (String) The string used to identify a particular standard in the Cal Set. An example string requesting the data for the Load standard in a full 2 port cal might be "S11C(3,3)".
- **lnumpoints**  (long) - The number of data points in the real and imaginary arrays.
- **real**  (Single) The real component of the complex data.
- **imag**  (Single) The imaginary component of the complex data.

Return Value  Single

Default  Not Applicable

Examples  See an Example

C++ Syntax  

```
HRESULT PutStandardComplexByString( BSTR bufferName, long InumPoints, float* real, float* imag);
```

Interface  ICalData3
### Quit Method

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Terminates the Network Analyzer application.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.Quit</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.Quit</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Quit()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Remarks</strong></td>
<td>Under the rules of COM, the server should not exit until all references to it have been released. This method is a brute force way of terminating the application. Be sure to release all references (or terminate the client program) before attempting to restart the Network Analyzer application. An alternate approach to terminating the application is to make the application invisible (app. Visible = False) and release all references. The server will shutdown.</td>
</tr>
</tbody>
</table>
Read-only

About the ExtTestSetIO connector

ReadData Method

Description
Reads a 13-bit data word from the specified address. Data is read using the AD0 through AD12 lines of the external test set connector. The instrument generates the appropriate timing signals. It automatically controls timing signals LDS, LAS and RLW to strobe the address, and then read the data, from the external test set. See the timing diagram for Address and Data I/O read.

VB Syntax
value = ExtIO.ReadData (address)

Variable (Type) - Description
value (variant) - Variable to store the returned data
ExtIO (object) - An ExternalTestSetIO object
address (variant) - address to read data from.

Return Type
Variant

Default
Not Applicable

Examples
value = ExtIO.ReadData (15)

C++ Syntax
HRESULT ReadData (VARIANT Address, VARIANT* Data);

Interface
IHWExternalTestSetIO
About the ExtTestSetIO connector

ReadRaw Method

**Description**
Reads a 16-bit value from the external test set. The 16-bit value is comprised of lines AD0 - AD12, Sweep Holdoff In and Interrupt In (inverted).

When this command is used the analyzer does NOT generate the appropriate timing signals; it simply reads the lines. The user needs to first use the `WriteRaw` method to do the initial setup. The RLW line (pin25) must be set to the appropriate level in order to read the test set connected.

Below is the format of data that is read with ReadRaw:

<table>
<thead>
<tr>
<th>Pin</th>
<th>Bit</th>
<th>Signal name</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>0</td>
<td>AD0*</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>AD1*</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>AD2*</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>AD3*</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>AD4*</td>
</tr>
<tr>
<td>21</td>
<td>5</td>
<td>AD5*</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
<td>AD6*</td>
</tr>
<tr>
<td>19</td>
<td>7</td>
<td>AD7*</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>AD8*</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>AD9*</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>AD10*</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>AD11*</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>AD12*</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>Sweep Holdoff In</td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>Interrupt In (inverted internally)</td>
</tr>
<tr>
<td>na</td>
<td>15</td>
<td>Always Zero, grounded internally</td>
</tr>
</tbody>
</table>
*These lines are dependent on the state of RLW (pin25).
Writing a 0(low) to RLW will set lines AD0-AD12 to write mode.
Writing a 1(high) to RLW will set lines AD0-AD12 to read mode.

**VB Syntax**

```
value = ExtIO.ReadRaw (address)
```

**Variable**  
**Type** - Description

- `value` *(variant)* - Variable to store the returned data
- `ExtIO` *(object)* - An External IO object
- `address` *(variant)* - Address to read data from

**Return Type**

Real

**Default**

Not Applicable

**Examples**

```
value = ExtIO.ReadRaw (address)
```

**C++ Syntax**

```cpp
HRESULT ReadRaw( VARIANT* Input );
```

**Interface**

IHWExternalTestSetIO
## Recall Method

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Recalls a measurement state, calibration state, or both, from the hard drive into the analyzer. Use app.Save to save files.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.Recall (filename.ext)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>app</code></td>
<td>An <code>Application</code> <em>(object)</em></td>
</tr>
<tr>
<td><code>filename.ext</code></td>
<td><em>(string)</em> - Full path, file name, and extension, of the file. Files are typically stored in &quot;C:\Program Files\Agilent\Network Analyzer\Documents&quot; Use one of the following extensions:</td>
</tr>
<tr>
<td></td>
<td>• .sta - Instrument State</td>
</tr>
<tr>
<td></td>
<td>• .cal - Calibration file</td>
</tr>
<tr>
<td></td>
<td>• .cst - Both Instrument State and Calibration reference</td>
</tr>
<tr>
<td></td>
<td>• .cti - Citifile (data will always be formatted. See Recalling Citifiles Using the PNA)</td>
</tr>
<tr>
<td></td>
<td>• .csa - Instrument state and calibration data (not a reference pointer).</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.Recall (&quot;C:\Program Files\Agilent\Network Analyzer\Documents\MyState.cst&quot;)</code> <em>Recalls &quot;mystate.cst&quot; from the specified folder</em></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Recall(BSTR bstrFile)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IApplication</code></td>
</tr>
</tbody>
</table>
## Recall Kits Method

**Description**
Recalls the calibration kits definitions that were stored with the SaveKits command.

**VB Syntax**
app.RecallKits

**Variable**
- (Type) - Description
  - **app** An Application (object)

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
app.RecallKits

**C++ Syntax**
HRESULT RecallKits()

**Interface**
IApplication
**Remove Method**

**Description**
Removes an item from a collection of objects.

**VB Syntax**
`Object.Remove item`

**Variable**
*(Type) - Description*

- **Object**
  Any of the following *(objects)*
  - `CalFactorSegments collection`
  - `Cal Sets collection`
  - `Measurements collection`
  - `NAWindows collection`
  - `PowerLossSegments collection`
  - `Segments collection`

**Note:** `Segments`, `CalFactorSegments`, and `PowerLossSegments` have an OPTIONAL argument `[size]` referring to the number of segments to remove, starting with the *item* parameter.

**Note:** `Segments` - When ALL segments are deleted, `SweepType` is automatically set to Linear because there are no segments to sweep.

- **item** *(variant)* - Item number to be removed

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
- `Measurements.Remove 3` 'Removes measurement 3 segments'
- `segments.Remove 2,20` 'Removes 20 segments (2 - 21)'

**C++ Syntax**
- `HRESULT Remove(VARIANT index);` //Measurements
- `HRESULT Remove(VARIANT index);` //Cal Sets
- `HRESULT Remove(long windowNumber);` //NAWindows
- `HRESULT Remove(VARIANT index, long size);` //Segments
- `HRESULT Remove(VARIANT index, long size);` //CalFactorSegments
- `HRESULT Remove(VARIANT index, long size);` //PowerLossSegments

**Interface**
- `IMeasurements`
- `INAWindows`
ISegments
ICalFactorSegments
ICal Sets
IPowerLossSegments
### About Presetting the Analyzer

#### Reset Method

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Removes all existing windows and measurements from the application. (Unlike <strong>Preset</strong>, does not create a new measurement.)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>app.Reset</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>app</code></td>
<td>An <strong>Application</strong> <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>app.Reset</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Reset()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><strong>IApplication</strong></td>
</tr>
</tbody>
</table>
**ResetLOFrequency Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets the LO Delta Frequency to 0 (zero) Hz.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>embedLO.ResetLOFrequency</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>embedLO</code></td>
<td>An EmbeddedLO <em>(object)</em></td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>embedLO.ResetLOFrequency</code> <code>write</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT ResetLOFrequency();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEEmbeddedLO</td>
</tr>
</tbody>
</table>

Last Modified:

18-Jun-2007   MX New topic
### ResetTuningParameters Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Resets the tuning parameters to their default values.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>embedLO.ResetTuningParameters</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>(Type) - Description</td>
</tr>
<tr>
<td>embedLO</td>
<td>An EmbeddedLO (object)</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>embedLO.ResetTuningParameters <code>write</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td>HRESULT ResetTuningParameters();</td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IEmbededLO</td>
</tr>
</tbody>
</table>

Last Modified: 13-Apr-2007    MX New topic
About Modifying Cal Kits

RestoreCalKitDefaults Method

Description
Restores the original properties of the specified Cal Kit, overwriting the last definition with the factory defaults.

NOTE: ONLY works with PNA releases 1.0 through 1.6.

VB Syntax
app.RestoreCalKitDefaults (calKit)

Variable (Type) - Description

app An Application (object)

calKit (enum NACalKit) - Calibration Kit to restore. Choose from:
1 - naCalKit_85032F_N50
2 - naCalKit_85033E_3_5
3 - naCalKit_85032B_N50
4 - naCalKit_85033D_3_5
5 - naCalKit_85038A_7_16
6 - naCalKit_85052C_3_5_TRL
7 - naCalKit_User7
8 - naCalKit_User8
9 - naCalKit_User9
10 - naCalKit_User10

Return Type Not Applicable

Default Not Applicable

Examples app.RestoreCalKitDefaults naCalKit_MechKit10

C++ Syntax HRESULT RestoreCalKitDefaults(tagNACalKit kit)

Interface IApplication
**Description**
Restores the original properties of ALL of the Cal Kits, overwriting the last definitions with the factory defaults.

**NOTE:** ONLY works with PNA releases 1.0 through 1.6.

**VB Syntax**
```
app.RestoreCalKitDefaultsAll
```

**Variable (Type) - Description**
- `app` An `Application` (object)

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
app.RestoreCalKitDefaultsAll
```

**C++ Syntax**
```
HRESULT RestoreCalKitDefaultsAll()
```

**Interface**
`IApplication`
**Resume Method**

**Description**
Resumes the trigger mode of all channels that was in effect before sending the `channels.Hold` method. `Channels.Hold` must be sent before `channels.Resume`, using the same instance of the `Channels` object.

**VB Syntax**
```
chans.Resume
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>chans</code></td>
<td>A <code>Channel collection (object)</code></td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
chans.Resume
```

**C++ Syntax**
```
HRESULT Resume();
```

**Interface**
IChannels2
### Save Method

**Description**
Saves the appropriate content to the hard drive depending on the extension that is provided.

Some saved files can be recalled using `app.Recall`, depending on the content.

**VB Syntax**
`app.Save(filename.ext)`

**Variable**
- **(Type)** - Description
  - `app` - An Application (object)
  - `filename.ext` - Full path, file name, and extension of the file.

Files are typically stored in "C:\Program Files\Agilent\Network Analyzer\Documents"

Use one of the following extensions:

- **.cst** - Saves both Instrument State and Cal Set reference - Recalls a calibrated measurement. (Recallable)
- **.sta** - Saves Instrument State only - recalls the instrument state without calibration. (Recallable)
- **.cal** - Calibration file – saves the active Cal Sets currently in use by any channel. Use this mode for archival purposes only. All Cal Sets are saved to a Cal Set data file. This mode provides a method of safeguarding calibration data. This data can be restored to the list of Cal Sets available in the instrument. (Recallable)
- **.csa** - Saves both instrument state AND actual calibration data, not a reference pointer to the Cal Set.
- **.prn** - Saves active trace in comma-separated format (not recallable)
- **.bmp** - Saves a Bitmap of the screen (not recallable)
- **.s1p** - Saves 1-port measurement data (not recallable)
- **.s2p** - Saves 2-port measurement data (not recallable)
- **.s3p** - Saves 3-port measurement data (not recallable)
- **.s4p** - Saves 4-port measurement data (not recallable)

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbscript
app.Save("C:\Program Files\Agilent\Network Analyzer\Documents\Newfolder\MyState.cst") 'Saves "mystate.cst" to the specified folder
```
<table>
<thead>
<tr>
<th><strong>C++ Syntax</strong></th>
<th>HRESULT Save(BSTR bstrFile)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interface</strong></td>
<td>IApplication</td>
</tr>
</tbody>
</table>

Last Modified:

26-Jun-2007    Corrected example
### Save Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the current Cal Set to disk. This is the recommended method for saving a Cal Set. Learn more about <a href="#">reading and writing Cal data using COM</a>.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>CalSet.Save</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td><code>CalSet</code></td>
<td><em>(object)</em> - A <code>CalSet</code> object</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>myCalSet.Save</code></td>
</tr>
<tr>
<td>See <a href="#">Copy Method</a> for an example application of this command.</td>
<td></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT Save();</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>ICalSet</code></td>
</tr>
</tbody>
</table>
**SaveCalSets Method**  Superseded

**Description**  This command is replaced by `ICalSet::Save` which saves the data for only the current Cal Set to the disk.

Writes new or changed Cal Sets to disk. All Cal Sets are saved in a single file. This file is updated at the following times:

- When a Cal Set has been deleted.
- When a calibration has been performed through the front panel interface.
- When this method is called.
- When `ICalSet::Save` is called.

Learn more about reading and writing Cal data using COM

**VB Syntax**  
```
object.SaveCalSets```

**Variable**  
`object` - Description

- `object` (object) - A CalManager object or a Calibrator object

**Return Type**  None

**Default**  Not Applicable

**Example**  
`calMgr.SaveCalSets`

**C++ Syntax**  
`HRESULT SaveCalSets();`

**Interface**  
ICalManager
ICalibrator
### SaveCitiDataData Method

**Description**
Saves UNFORMATTED trace data to .cti file. Learn more about citifiles.

**VB Syntax**
```
app.SaveCitiDataData(filename.cti)
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>Application (object)</td>
<td></td>
</tr>
<tr>
<td>filename.cti</td>
<td>string</td>
<td>Full path, file name, and .cti extension of the file. Files are typically stored in &quot;C:\Program Files\Agilent\Network Analyzer\Documents&quot;.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
app.SaveCitiDataData ("C:\Program Files\Agilent\Network Analyzer\Documents\myDDCitifile.cti") 'Saves "myDDCitifile.cti" to the specified folder
```

**C++ Syntax**
```
HRESULT SaveCitiDataData (BSTR bstrFile)
```

**Interface**
IApplication5
### SaveCitiFormattedData Method

**Description**  
Saves FORMATTED trace data to .cti file. [Learn more about citifiles.](#)

**VB Syntax**  
```vb
app.SaveCitiFormattedData(filename.cti)
```

**Variable (Type) - Description**

- **app**  
  An [Application](#) (object)

- **filename.cti**  
  (string) - Full path, file name, and .cti extension of the file.  
  Files are typically stored in "C:\Program Files\Agilent\Network Analyzer\Documents"

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
```vb
app.SaveCitiFormattedData("C:\Program Files\Agilent\Network Analyzer\Documents\Newfolder\myFDCitifile.cti") 'Saves "myFDCitifile.cti" to the specified folder
```

**C++ Syntax**  
HRESULT SaveCitiFormattedData (BSTR bstrFile)

**Interface**  
IApplication5
SaveENRFile Method

**Description**
Saves an ENR table to disk.

**VB Syntax**
`enr.SaveENRFile (filename)`

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enr</td>
<td>An ENRFile (object)</td>
</tr>
<tr>
<td>filename</td>
<td>(String) - Absolute path and filename of the ENR file.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
See example program

**C++ Syntax**

```cpp
HRESULT SaveENRFile(BSTR filename);
```

**Interface**
IENRFile

---

Last Modified:

2-Aug-2007    MX New topic
## SaveFile Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the mixer/convertor test setup to a mixer attributes (.mxr) file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mixer.SaveFile(filename)</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>mixer</code></td>
<td>An <code>IMixer Interface</code> pointer to a <code>measurement</code> (object)</td>
</tr>
<tr>
<td><code>filename</code></td>
<td>(String) Full path, file name, and .mxr extension of the file. Files are typically stored in &quot;C:\Program Files\Agilent\Network Analyzer\Documents&quot;.</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>String</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mixer.SaveFile(&quot;C:\Program Files\Agilent\Network Analyzer\Documents\myMixer.mxr&quot;)</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT put_SaveFile(BSTR newVal)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td><code>IMixer</code></td>
</tr>
</tbody>
</table>
SaveKits Method

**Description**
Saves the cal kits, typically after modifying a calibration kit. To load a cal kit into the analyzer from the hard drive, use `app.RecallKits`.

**VB Syntax**
`app.SaveKits`

**Variable**
(Type) - Description

- `app` An `Application` (object)

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
`app.SaveKits`

**C++ Syntax**
`HRESULT SaveKits()`

**Interface**
`IApplication`
## SearchFilterBandwidth Method

**Description**  
Searches the measurement data with the current BandwidthTarget (default is -3). To continually track the filter bandwidth, use BandwidthTracking. This feature uses markers 1-4. If not already, they are activated. To turn off these markers, either turn them off individually or DeleteAllMarkers.

The bandwidth statistics are displayed on the analyzer screen. To get the bandwidth statistics, use either GetFilterStatistics or FilterBW, FilterCF, FilterLoss, or FilterQ.

The analyzer screen will show either Bandwidth statistics OR Trace statistics; not both.

To search a UserRange with the bandwidth search, first activate marker 1 and set the desired UserRange. Then send the SearchFilterBandwidth command. The user range used with bandwidth search only applies to marker 1 searching for the max value. The other markers may fall outside the user range.

### VB Syntax

```vbnet
meas.SearchFilterBandwidth
```

### Variable (Type) - Description

- **meas**  
  A Measurement (object)

### Return Type
Not Applicable

### Default
Not Applicable

### Examples

```vbnet
meas.SearchFilterBandwidth
```

### C++ Syntax

```cpp
HRESULT SearchFilterBandwidth()
```

### Interface
IMeasurement
## SearchMax Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the marker domain for the maximum value.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.SearchMax</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><em>(Type) - Description</em></td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker <em>(object)</em></td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.SearchMax</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SearchMax()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
## SearchMin Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the marker domain for the minimum value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark/SearchMin</code></td>
</tr>
<tr>
<td>Variable (Type)</td>
<td>(object) A Marker</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark/SearchMin</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SearchMin()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
# SearchNextPeak Method

**Description**  Searches the marker's domain for the next peak value.

**VB Syntax**  

```
mark.SearchNextPeak
```

**Variable**  

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mark</td>
<td>A Marker (object)</td>
</tr>
</tbody>
</table>

**Return Type**  Not Applicable

**Default**  Not Applicable

**Examples**  

```
mark.SearchNextPeak
```

**C++ Syntax**  

```
HRESULT SearchNextPeak()
```

**Interface**  IMarker
SearchPeakLeft Method

**Description**
Searches the marker's domain for the next **VALID** peak to the left of the marker.

**VB Syntax**

mark/SearchPeakLeft

**Variable**

*(Type)* - **Description**

mark A Marker *(object)*

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

mark/SearchPeakLeft

**C++ Syntax**

HRESULT SearchPeakLeft()

**Interface**
IMarker
SearchPeakRight Method

**Description**
Searches the marker's domain for the next **VALID** peak to the right of the marker.

**VB Syntax**
mark.SearchPeakRight

**Variable**
- **mark**
  - A Marker (object)

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
mark.SearchPeakRight

**C++ Syntax**
HRESULT SearchPeakRight()

**Interface**
IMarker
### SearchTarget Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Searches the marker's domain for the target value (specified with <code>mark.TargetValue</code>). Searches to the right; then at the end of the search domain, begins again at the start of the search domain.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.SearchTarget</code></td>
</tr>
<tr>
<td>Variable (Type) - Description</td>
<td><code>mark</code> A Marker (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark.SearchTarget</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SearchTarget()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
**SearchTargetLeft Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Moving to the left of the marker position, searches the marker’s domain for the target value (specified with <code>mark.TargetValue</code>).</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.SearchTargetLeft</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td>mark</td>
<td>A Marker <strong>(object)</strong></td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark/SearchTargetLeft</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT SearchTargetLeft()</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
## SearchTargetRight Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Moving to the right of the marker position, searches the marker's domain for the target value (specified with <code>mark.TargetValue</code>).</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.SearchTargetRight</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td></td>
<td><code>mark</code> A Marker <em>(object)</em></td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark.SearchTargetRight</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT SearchTargetRight()</code></td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
SelectCalSet Method

**Description**
Selects and applies a Cal Set to the specified channel.

**Note:** Error Correction is not automatically applied as a result of this command being issued. If there is more than one Cal Type in the Cal Set, you must explicitly choose the Cal Type you want to apply. (See meas.CalType)

**VB Syntax**

```
channel.SelectCalSet calSet, restore
```

**Variable**

- **channel** *(Type)* - A Channel object

- **calSet** *(string)* - Cal Set to make active. Specify the Cal Set by GUID or Name. Use EnumerateCalSets to list the available Cal Sets.

- **restore** *(boolean)* -
  - True (1) - The stimulus stored with the cal set will be applied to the channel.
  - False (0) - If a conflict is detected between the existing channel settings and the Cal Set stimulus settings, then the following will occur:
    - If interpolation is ON, then interpolation will be attempted. This may fail if the channel frequency is outside the range of the Cal Set.
    - If interpolation is OFF, the selection will be abandoned and an error is returned:
      ```
      E_NA_CAL_STIMULUS_VALUES_EXCEEDED
      ```

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```
channel.SelectCalSet GUID, 1
chan.SelectCalSet "MyCalSet", 0
```

**C++ Syntax**

```
HRESULT SelectCalSet (BSTR strCset, bool bRestore);
```

**Interface**
IChannel

---

Last Modified:

29-Nov-2007  Modified to accept name
SetAllSegments Method

**Description**
Uploads a segment table to the PNA replacing any existing segment table. Segments must be ascending in frequency and non-overlapping. If they are not, the segments are 'adjusted' as they are from the User Interface control. The total number of points for all segments cannot exceed the PNA maximum number of points for a sweep.

**VB Syntax**
```
Segs.SetAllSegments (segdata)
```

**Variable (Type) - Description**

- **segs** A Segments (Collections)
- **segdata** (Variant) A 2-dimensional array of Segment data:
  - dimension 0 is the number of elements in each segment.
  - dimension 1 is the number of segments that will be used.

The following is a list of dimension 0 elements for each segment:

**Note:** All elements must be dimensioned as either ALL Double or ALL Variant.

- 0 = Segment state (Boolean True or False)
- 1 = Number of Points in this segment (Integer)
- 2 = Start Freq (Double)
- 3 = Stop Freq (Double)
- 4 = IFBW (Double)
- 5 = Dwell Time (Double)
- 6 = Power (Double) - see table below.

The IFBandwidthOption and SweepTimeOption settings do NOT affect the array size and order-of-element interpretation. The array must always contain elements that contain IFBW and Sweep Time values. The value is ignored if the corresponding 'option' is set to False.

The number of dimension 0 elements depends on the following two settings:

1. **SourcePowerOption** = True allows segments to have independent power levels.
2. **CouplePorts** = False allows different power levels for each test port.
<table>
<thead>
<tr>
<th>CouplePorts</th>
<th>SourcePowerOption</th>
<th>Number of Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>False</td>
<td>Each port has its own channel-wide power setting, which is set using <code>TestPortPower</code>. Supply exactly 7 elements per segment. The last element (power) is ignored.</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>Supply 6 + total number of ports. The first 7 elements are still interpreted the same. The remaining elements (in-order) are interpreted as the power levels to set on that segment for Ports 2 through N, where N is the total number of ports currently enabled for the PNA or for a PNA with multiport external test set.</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>Supply exactly 7 elements per segment. The last element (power) is ignored.</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>Supply exactly 7 elements per segment. The last element (power) is honored.</td>
</tr>
</tbody>
</table>

**Return Type**  Not Applicable

**Default**  Not Applicable

**Examples**  
See a VB example using this command
See a C++ example using this command

**C++ Syntax**  
```c++
SetAllSegments (VARIANT Segments );
```

**Interface**  
`ISegments2`

---

Last Modified:  
15-Oct-2007  Major edits and link to C++ example
### SetBBPorts Method

**Description**
For a Balanced - Balanced device type, maps the PNA ports to the DUT ports.
Set the Balanced device type using the [DUTTopology Property](#).

**VB Syntax**
```vbnet
balTopology.SetBBPorts p1Pos, p1Neg, p2Pos, p2Neg
```

**Variable**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>balTopology</td>
<td></td>
<td>A BalancedTopology (object)</td>
</tr>
<tr>
<td>p1Pos, p1Neg, p2Pos, p2Neg</td>
<td>Long Integer</td>
<td>PNA port number that connects to each of the following DUT ports:</td>
</tr>
</tbody>
</table>

![Diagram](image)

**Return Type**
Not applicable - To read port mappings, use the [BalancedTopology properties](#).

**Default**
Not Applicable

**Examples**
```
balTop.SetBBPorts 1, 2, 3, 4
```

**C++ Syntax**
```cpp
HRESULT SetBBPorts (long p1Pos, long p1Neg, long p2Pos, long p2Neg)
```

**Interface**
IBalancedTopology
### Description
Specifies the type of Unguided calibration. This method should be the first method called on the calibrator object. It prepares the internal state for the rest of the calibration.

**Note:** You can NOT perform a 3 or 4-port cal using SetCalInfo even though there is `enumCalTypes`. You must use the `GuidedCalibration` object.

Learn more about reading and writing Cal data using COM

The analyzer can measure both ports simultaneously, assuming you have two of each standard type. For a 2-port cal, See `cal.Simultaneous2PortAcquisition`

### VB Syntax
```
cal.SetCalInfo (type,rcvPort,srcPort)
```

### Variable - Description

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cal</code></td>
<td>A Calibrator</td>
<td>(object)</td>
</tr>
<tr>
<td><code>type</code></td>
<td>(enum NACalType)</td>
<td>- Calibration type. Choose from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - naCalType_Response_Open</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 - naCalType_Response_Short</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - naCalType_Response_Thru</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - naCalType_Response_Thru_And_Isol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 - naCalType_OnePort</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 - naCalType_TwoPort_SOLT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 - naCalType_TwoPortTRL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 - naCalType_None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 - naCalType_ThreePort_SOLT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9 - Custom</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 - naCalType_FourPort_SOLT</td>
</tr>
</tbody>
</table>

**Note:** For 1-port cals, the source port = receiver port. For 2, 3, 4-port SOLT and TRL, it doesn't matter which port is specified as source and receiver

<table>
<thead>
<tr>
<th>Variable</th>
<th>(long integer)</th>
<th>Receiver Port</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>rcvPort</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>srcPort</code></td>
<td>(long integer)</td>
<td>Source Port</td>
</tr>
</tbody>
</table>

### Return Type
NACalType

### Default
7-naCalType_None

### Examples
```
cal.setCalInfo (naCalType_Response_Open,1,1)
```
<table>
<thead>
<tr>
<th><strong>C++ Syntax</strong></th>
<th>HRESULT SetCalInfo(tagNACalType calType, long portA, long portB)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interface</strong></td>
<td>ICalibrator</td>
</tr>
</tbody>
</table>
SetCalInfoEx Method (for source power cals)

**Description**
This command replaces `SetCalInfo2 Method`. Specifies the channel and the source port to be used for the source power calibration about to be performed.

**VB Syntax**
```vbnet
powerCalibrator.SetCalInfoEx channel, sourcePort, [powerOffset,] [display]
```

**Variable (Type) - Description**
- `powerCalibrator (object)` - A `SourcePowerCalibrator` object
- `channel (long integer)` - Number of the PNA channel (not power meter channel) on which the source power cal will be performed. If the channel does not already exist, it will be created.
- `sourcePort (long integer)` - Port number on which the source power cal will be performed. Use `GetPortNumber` to return the port number of a source that only has a string name, such as an `External Source`.
- `[powerOffset] (double)` - Optional argument. Sets or returns a power level offset from the PNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT. Following the calibration, the PNA power readouts are adjusted by this value. This argument performs the same function as `chan.SourcePowerCalPowerOffset Property`.
- `[display] (boolean)` - Optional argument. Enables and disables the display of power readings on the PNA screen. After the source power cal data is acquired, this setting is reset to ON. If unspecified, value is set to ON.
  - **True** - Display of power readings is ON
  - **False** - Display of power readings is OFF

**Return Type**
None

**Default**
Not Applicable

**Examples**
```vbnet
powerCalibrator.SetCalInfoEx 1, 1, -10, True
```

**C++ Syntax**
```cpp
HRESULT SetCalInfoEx( long Channel, long SourcePort, double PowerOffset = 0., VARIANT_BOOL bDisplay = VARIANT_TRUE);
```

**Interface**
`ISourcePowerCalibrator4`

_Last Modified:_
30-Apr-2007   Edited for src strings
**SetCenter Method**

**Description**
Changes the center stimulus to the stimulus value of the marker. The start stimulus stays the same and the stop is adjusted.

This command does not work with channels that are in CW or Segment Sweep mode.

**VB Syntax**

```vbnet
mark.SetCenter
```

**Variable (Type) - Description**

- `mark` A Marker (object)

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
mark.SetCenter
```

**C++ Syntax**

```cpp
HRESULT SetCenter()
```

**Interface**
IMarker
### SetCW Method

**Description**
Changes the analyzer to sweep type CW mode and sets the CW frequency to the marker’s frequency. Does not change anything if current sweep type is other than a frequency sweep.

**VB Syntax**
```
mark.SetCW
```

**Variable**
- **(Type)** - Description
  - `mark` A Marker *(object)*

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
mark.SetCW
```

**C++ Syntax**
```cpp
HRESULT SetCW()
```

**Interface**
IMarker
**SetElectricalDelay Method**

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Changes the measurement's electrical delay to the marker's delay value.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>mark.SetElectricalDelay</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type) - Description</strong></td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>mark.SetElectricalDelay</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT SetElectricalDelay()</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>IMarker</td>
</tr>
</tbody>
</table>
### SetFailOnOverRange Method

**Description**
When set TRUE, configures the analyzer to report outOfRange conditions with an error code. Any overrange error will return `E_NA_LIMIT_OUTOFRANGE_ERROR`.

**Note:** This method is for the benefit of VB clients. The analyzer automatically adjusts overrange conditions to the closest acceptable setting. The VB user will not see that an overrange occurred because the HRESULT is not returned if it has a success code. For more information, see Events/OverRange.

**VB Syntax**
```
app.SetFailOnOverRange state
```

**Variable**
- **(Type)** Description
  - **app** An Application (object)
  - **state** (boolean) -
    - True (1) - Overrange conditions report an error code
    - False (0) - Overrange conditions report a success code

**Return Type**
Not Applicable

**Default**
False (0)

**VB Example**
```
app.SetFailOnOverRange TRUE
On Error Goto ERRHANDLER

' the following overrange will cause ERRHANDLER to be invoked
channel.StartFrequency = 9.9 GHZ
exit

ERRHANDLER:
  print "something failed"
```

**C++ Syntax**
`HRESULT put_SetFailOnOverRange(VARIANT_BOOL mode)`

**Interface**
IAplication
SetIsolationPaths Method

**Description**
Adjusts the list of paths (port pairings) for which isolation standards will be measured during calibration.

**VB Syntax**
guidedCal.SetIsolationPaths specifier, pathList

**Variable (Type) - Description**
- **obj**
  Any of the following:
  GuidedCalibration (object)
- **specifier** (Enum) - Choose from:
  0 - naPathsAll - Measure isolation on all pairings of the ports that are to be calibrated.
  1 - naPathsNone - Do not measure isolation on any pairing of the ports to be calibrated.
  2 - naPathsAdd - Add one or more specific pairings of ports to the list of port pairings for which isolation will be measured.
  3 - naPathsRemove - Remove one or more specific pairings of ports from the list of port pairings for which isolation will be measured.
- **pathlist** (Variant) port numbers in pairs. One-dimensional array of Long Integers.
  **Note:** pathList is evaluated only when specifier is naPathsAdd or naPathsRemove.
  For naPathsAll and naPathsNone, pathList is ignored.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
Dim pathList
'selecting to measure isolation on all possible paths for the ports about to be calibrated
guidedCal.SetIsolationPaths naPathsAll, pathList

'now removing the paths 1-to-2, 2-to-3 and 2-to-4 from the set of all paths
pathList = Array(1,2,2,3,2,4)
guidedCal.SetIsolationPaths naPathsRemove, pathList

**C++ Syntax**
HRESULT SetIsolationPaths(enum NAPortPathSpecifier specifier, VARIANT pathList);

**Interface**
IGuidedCalibration3

Last Modified:
16-Apr-2007   MX New topic
### SetPowerAcquisitionDevice Method

**Description**  
Sets the power sensor channel (A or B) to be used. This performs the same function as the **Use this sensor only** checkbox in the Power Sensor Settings dialog.  
**Note:** This method is only necessary when performing an SMC calibration.

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pwrCal</td>
<td>(Object) A SourcePowerCalibrator object</td>
</tr>
<tr>
<td>sensor</td>
<td>(enum NAPowerAcquisitionDevice) The power sensor channel. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 – naPowerSensor_A</td>
</tr>
<tr>
<td></td>
<td>1 – naPowerSensor_B</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**VB Syntax**  
`pwrCal.SetPowerAcquisitionDevice sensor`

**C++ Syntax**  
`HRESULT SetPowerAcquisitionDevice( tagNAPowerAcquisitionDevice enumAcqDevice);`

**Interface**  
ISourcePowerCalibrator3
### SetFrequencyLowPass Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Set the start frequencies when <code>trans.Mode = LowPass</code>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>trans.SetFrequencyLowPass</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(<code>Type</code>) - Description</td>
</tr>
<tr>
<td><code>trans</code></td>
<td>A Transform (<code>object</code>)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>trans.SetFrequencyLowPass</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td><code>HRESULT SetFrequencyLowPass(void)</code></td>
</tr>
<tr>
<td>Interface</td>
<td><code>ITransform</code></td>
</tr>
</tbody>
</table>
SetPortMap Method

**Description**
Set the DUT-to-PNA port mapping for the Gain Compression measurement.

**VB Syntax**
gca.SetPortMap in,out

**Variable**

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>gca</strong></td>
<td>A GainCompression (object)</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>PNA port which is connected to the DUT input.</td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>PNA port which is connected to the DUT output.</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
1,2

**Examples**
gca.SetPortMap 2,1

**C++ Syntax**
HRESULT SetPortMap(long input_port,long output_port);

**Interface**
IGainCompression
### SetReferenceLevel Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Changes the measurement's reference level to the marker's Y-axis value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.SetReferenceLevel</code></td>
</tr>
<tr>
<td>Variable</td>
<td><strong>(Type)</strong> - Description</td>
</tr>
<tr>
<td>mark</td>
<td>A Marker (<strong>object</strong>)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark.SetReferenceLevel</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SetReferenceLevel()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>
Write-only

About Balanced Measurements

SetSBPorts Method

**Description**
For a Single-ended - Balanced device type, maps the PNA ports to the DUT ports. Set the Single-ended - Balanced device type using the [DUTTopology Property](#).

**VB Syntax**
```vbnet
balTopology.SetSBPorts se, bPos, bNeg
```

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>balTopology</code></td>
<td>A</td>
<td>BalancedTopology (object)</td>
</tr>
<tr>
<td><code>se, bPos, bNeg</code></td>
<td></td>
<td>PNA port number that connects to each of the following DUT ports:</td>
</tr>
</tbody>
</table>

![Diagram](image.png)

**Return Type**
Not applicable - To read port mappings, use the [BalancedTopology](#) properties.

**Default**
Not Applicable

**Examples**
`balTop.SetSBPorts 1, 2, 3`

**C++ Syntax**
```cpp
HRESULT SetSBPorts (long se, long bPos, long bNeg)
```

**Interface**
IBalancedTopology
**SetSSBPorts Method**

**Description**
For a Single-ended - Single-ended - Balanced device type, maps the PNA ports to the DUT ports.

Set the Single-ended - Single-ended - Balanced device type using the [DUTTopology Property](#).

**VB Syntax**
```
balTopology.SetSSBPorts se, se2, bPos, bNeg
```

**Variable**

- **balTopology**
  A [BalancedTopology](#) object

- **se, se2, bPos, bNeg**
  PNA port number that connects to each of the following DUT ports:

```
<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-end Port 1</td>
<td>se1, se2</td>
</tr>
<tr>
<td>Bal Port 3</td>
<td>bPos</td>
</tr>
<tr>
<td>Single-end Port 2</td>
<td>bNeg</td>
</tr>
</tbody>
</table>
```

**Return Type**
Not applicable - To read port mappings, use the [BalancedTopology properties](#).

**Default**
Not Applicable

**Examples**
```
balTop.SetSSBPorts 1, 2, 3, 4
```

**C++ Syntax**
```
HRESULT SetSSBPorts (long se, long se2, long bPos, long bNeg)
```

**Interface**
[IBalancedTopology](#)
### SetupMeasurementsForStep Method

**Description**
Show the Cal Window, and optionally one or more other specific windows, before acquiring a Cal standard. This command will cause the Cal Window to display the specific measurements that are to be made for that particular Cal standard. See custom Cal window commands.

**VB Syntax**
`guidedCal.SetupMeasurementsForStep n`

**Variable**

- `Type`: Description
- `guidedCal` A `GuidedCalibration` (object)
- `n` Step number in the calibration process.
  - Use `GenerateSteps` to determine the total number of steps.
  - Use `GetStepDescription` to read the description of each step.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
`guidedCal.SetupMeasurementsForStep 3`
See example using this command

**C++ Syntax**
`HRESULT SetupMeasurementsForStep(long step);`

**Interface**
`IGuidedCalibration4`

---

Last Modified:
8-Nov-2007 MX New topic
## SetStandardsForClass Method

**Description**  
Set the calibration standard numbers for a specified calibration class. To read the calibration standard numbers use the [GetStandardsForClass Method](#).

**VB Syntax**  
```
calKit.SetStandardsForClass (calclassorder, std1, std2, std3, std4, std5, std6, std7)
```

**Variable**  
(Variable) - Description

- **calKit**  
  A CalKit (object)

- **calclassorder**  
  (enum NACalClassOrder) Cal Class. Choose from:

  0 - naRefl_1_S11  
  1 - naRefl_2_S11  
  2 - naRefl_3_S11  
  3 - naTran_1_S21  
  4 - naRefl_1_S22  
  5 - naRefl_2_S22  
  6 - naRefl_3_S22  
  7 - naTran_1_S12  
  8 - naRefl_1_S33  
  9 - naRefl_2_S33  
  10 - naRefl_3_S33  
  11 - naTran_1_S32  
  12 - naTran_1_S23  
  13 - naTran_1_S31  
  14 - naTran_1_S13  
  15 - naTRL_T  
  16 - naTRL_R  
  17 - naTRL_L

- **std1…std7**  
  (long) Calibration Standard Number. Choose from 1 through 30. Std2 through Std7 are optional

**Return Type**  
Not applicable
**Default**  Not applicable

**Examples**
- `calkit.SetStandardsForClass naRefI_3_S11, 3, 5, 6`
- `calkit.SetStandardsForClass naTran_1_S21, 4`

**C++ Syntax**

```
HRESULT SetStandardsForClass(NACalClassOrder calclassorder, long std1, long std2, long std3, long std4, long std5, long std6, long std7)
```

**Interface**
ICalKit
### SetStart Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Changes the start stimulus to the stimulus value of the marker. The stop stimulus stays the same and the span is adjusted. This command does not work with channels that are in CW or Segment Sweep mode.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB Syntax</td>
<td><code>mark.SetStart</code></td>
</tr>
<tr>
<td>Variable</td>
<td>(Type) - Description</td>
</tr>
<tr>
<td><code>mark</code></td>
<td>A Marker (object)</td>
</tr>
<tr>
<td>Return Type</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Examples</td>
<td><code>mark.SetStart</code></td>
</tr>
<tr>
<td>C++ Syntax</td>
<td>HRESULT SetStart()</td>
</tr>
<tr>
<td>Interface</td>
<td>IMarker</td>
</tr>
</tbody>
</table>


### SetStop Method

**Description**  Changes the stop stimulus to the stimulus value of the marker. The start stimulus stays the same and the span is adjusted.

This command does not work with channels that are in **CW** or **Segment Sweep** mode.

**VB Syntax**  `mark.SetStop`

**Variable (Type) - Description**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mark</code></td>
<td>A Marker (object)</td>
</tr>
</tbody>
</table>

**Return Type**  Not Applicable

**Default**  Not Applicable

**Examples**  `mark.SetStop`

**C++ Syntax**  `HRESULT SetStop()`

**Interface**  `IMarker`
# ShowMarkerReadout Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Shows and Hides the Marker readout for the active marker in the upper-right corner of the window.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td><code>win.ShowMarkerReadout state</code></td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td><strong>(Type)</strong> - <strong>Description</strong></td>
</tr>
<tr>
<td><code>win</code></td>
<td>A NAWindow <em>(object)</em></td>
</tr>
<tr>
<td><code>state</code></td>
<td><em>(boolean)</em> -</td>
</tr>
<tr>
<td></td>
<td><strong>True</strong> (1) - Show the Marker readout</td>
</tr>
<tr>
<td></td>
<td><strong>False</strong> (0) - Hide the Marker readout</td>
</tr>
<tr>
<td><strong>Return Type</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td><code>win.ShowMarkerReadout True</code></td>
</tr>
<tr>
<td><strong>C++ Syntax</strong></td>
<td><code>HRESULT ShowMarkerReadout(VARIANT_BOOL bState)</code></td>
</tr>
<tr>
<td><strong>Interface</strong></td>
<td>INAWindow</td>
</tr>
</tbody>
</table>
ShowStatusBar Method

**Description**
Shows and Hides the Status Bar. The Status Bar is located across the bottom of the display. The following information is shown for the active measurement:

- Channel number
- Parameter
- Correction On or Off
- Remote or Local operation

**VB Syntax**

```vbnet
app.ShowStatusBar state
```

**Variable**

- **app** (Type) - Description
  - An Application (object)

- **state** (boolean) -
  - True (1) - Show the Status Bar
  - False (0) - Hide the Status Bar

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**

```vbnet
app.ShowStatusBar True
```

**C++ Syntax**

```cpp
HRESULT ShowStatusBar (VARIANT_BOOL bState)
```

**Interface**
IApplication
ShowStimulus Method

Description  Shows and Hides the Stimulus (X-axis) information located at the bottom of the display. The start and stop stimulus values are shown for the active measurement.

VB Syntax  
```
app.ShowStimulus state
```

Variable  

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>state (boolean)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>True (1)</td>
<td>Show the Stimulus information</td>
</tr>
<tr>
<td>False (0)</td>
<td>Hide the Stimulus information</td>
</tr>
</tbody>
</table>

Return Type  Not Applicable

Default  Not Applicable

Examples  
```
app.ShowStimulus True
```

C++ Syntax  HRESULT ShowStimulus(VARIANT_BOOL bState)

Interface  IApplication
ShowTable Method

**Description**  Shows or Hides the specified table for the window's active measurement in the lower part of the window.

**VB Syntax**  
`win.ShowTable value`

**Variable**  
(Variable - Type) - Description

- **win**  A NAWindow (object)

- **value**  (enum naTable) - The table to show or hide. Choose from:
  - 0 - naTable_None
  - 1 - naTable_Marker
  - 2 - naTable_Segment
  - 3 - naTable_Limit

**Return Type**  Not Applicable

**Default**  Not Applicable

**Examples**  
`win.ShowTable naTable_limit`

**C++ Syntax**  
HRESULT ShowTable (tagNATableType table)

**Interface**  INAWindow
**ShowTitleBars Method**

**Description**
Shows and Hides the Title Bars. The Title Bars are across the top of the Network Analyzer Window and each of the measurement windows. The Window name is shown in the Title Bar.

**VB Syntax**
```vbnet
app.ShowTitleBars state
```

**Variable (Type) - Description**

- **app**
  An Application (object)

- **state**
  (boolean)
  - **True (1)** - Show the Title Bars
  - **False (0)** - Hide the Title Bars

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```vbnet
app.ShowTitleBars True
```

**C++ Syntax**
```cpp
HRESULT ShowTitleBars(VARIANT_BOOL bState)
```

**Interface**
IApplication
# ShowToolbar Method

**Description**
Shows and Hides the specified Toolbar.

**VB Syntax**
```
app.ShowToolbar toolbar, state
```

**Variable**
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application (object)</td>
</tr>
<tr>
<td>toolbar</td>
<td>(enum NAToolbarType) - The toolbar to show or hide. Choose from:</td>
</tr>
<tr>
<td></td>
<td>0 - naToolbar_None</td>
</tr>
<tr>
<td></td>
<td>1 - naToolbar_ActiveEntry</td>
</tr>
<tr>
<td></td>
<td>2 - naToolbar_Markers</td>
</tr>
<tr>
<td></td>
<td>3 - naToolbar_Measurement</td>
</tr>
<tr>
<td></td>
<td>4 - naToolbar_Stimulus</td>
</tr>
<tr>
<td></td>
<td>5 - naToolbar_SweepControl</td>
</tr>
<tr>
<td>state</td>
<td>(boolean) -</td>
</tr>
<tr>
<td></td>
<td>True (1) - Show the specified toolbar</td>
</tr>
<tr>
<td></td>
<td>False (0) - Hide the specified toolbar</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
1 - naToolbar_ActiveEntry showing; all others hiding.

**Examples**
```
app.ShowToolbar 1, 1 'shows the active entry toolbar
```

**C++ Syntax**
```
HRESULT ShowToolbar(tagNAToolbarType toolbar, VARIANT_BOOL bState)
```

**Interface**
IAplication
### Single Method

#### Description
Sets the trigger count to 1, which will cause the channel to respond to exactly one trigger signal from an Internal or External trigger source.

If trigger source is set to Manual:

- with `sync = True`, trigger source is automatically changed to Internal which sends AND allows one trigger signal, then changed back to Manual.
- with `sync = False`, a trigger signal must also be sent using `app.ManualTrigger Method`.

This setting has implications on Calibration. [Learn more.](#)

#### VB Syntax
```
chan.Single [sync]
```

#### Variable *(Type) - Description*

- `chan` A Channel *(object)*
- `[sync]` *(boolean)* - Optional argument.

  - `True` - The PNA waits until the trigger is completed to process subsequent commands.
  - `False` - Subsequent commands are processed immediately (default setting).

#### Return Type
Not Applicable

#### Default
Not Applicable

#### Examples
```
sync = True
chan.Single sync
```

#### C++ Syntax
```
HRESULT Single(VARIANT_BOOL bWait)
```

#### Interface
`IChannel`

---

Last Modified:

10-Jul-2007 Corrected for manual mode with sync = True
## Store Method

<table>
<thead>
<tr>
<th>Description</th>
<th>Saves the path configuration currently associated with channel (ch) to the specified configuration name. This command is identical to PathConfigurationManager.StoreConfiguration Method</th>
</tr>
</thead>
</table>

### VB Syntax

`pathMgr.StoreConfiguration ch, name`

### Variable *(Type)* - Description

<table>
<thead>
<tr>
<th>pathMgr</th>
<th>PathConfigurationManager <em>(object)</em></th>
</tr>
</thead>
</table>

| ch | *(Long)* Channel number of the configuration to be saved. |
| name | *(String)* Configuration name. Factory configurations can NOT be overwritten. Specifying the name of a pre-defined factory configuration will result in an error. |

### Return Type

Not Applicable

### Default

Not Applicable

### Examples

`path.StoreConfiguration(2) "myMixer"`

### C++ Syntax

`HRESULT StoreConfiguration( long channelNum, BSTR configName );`

### Interface

IPathConfigurationManager

---

Last modified:

Dec.12, 2006   MX New Command
StoreConfiguration Method

**Description**
Saves the path configuration currently associated with channel (ch) to the specified configuration name.

**VB Syntax**
`pathMgr.StoreConfiguration ch, name`

**Variable (Type) - Description**
- `pathMgr` **PathConfigurationManager (object)**
- `ch` **(Long)** Channel number of the configuration to be saved.
- `name` **(String)** Configuration name. Factory configurations can NOT be overwritten. Specifying the name of a pre-defined factory configuration will result in an error.

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
`path.StoreConfiguration(2) "myMixer"

**C++ Syntax**
`HRESULT StoreConfiguration( long channelNum, BSTR configName );`

**Interface**
IPathConfigurationManager

---

Last Modified:
14-Dec-2006   MX New topic
StringToNACalClass Method

**Description**  Converts the returned strings from `GetStandardsList` into the enumeration (NACalClass) and the port numbers required for `PutStandard` and `GetStandard` methods that transmit data in and out of the Cal Set.

Learn more about [reading and writing Cal data using COM](#)

**VB Syntax**  
```
CalSet.StringToNACalClass (list, std, rcv, src)
```

**Variable**  

- **(Type)** - Description

  | **CalSet** | (object) - A Cal Set object |
  | **list**   | (string) - a string containing the textual description of the standard. |
  | **std**    | (enum NACalClass) Choose from: |

  1 - naClassA
  2 - naClassB
  3 - naClassC
  4 - naClassD
  5 - naClassE
  6 - naReferenceRatioLine
  7 - naReferenceRatioThru

**SOLT Standards**

  1 - naSOLT_Open
  2 - naSOLT_Short
  3 - naSOLT_Load
  4 - naSOLT_Thru
  5 - naSOLT_Isolation

**TRL Standards**

  1 - naTRL_Reflection
  2 - naTRL_Line_Reflection
  3 - naTRL_Line_Tracking
  4 - naTRL_Thru
  5 - naTRL_Isolation
rcv  (long) - port number of the receiver

src  (long) - port number of the source

**Return Type**  Not Applicable

**Default**  Not Applicable

**Examples**  `guid = CalSet.StringToNACalClass(list, std, rcv, src)`

**C++ Syntax**  `HRESULT StringToNACalClass ( BSTR* str, NACalClass* item, long *rcv, long *src);`

**Interface**  ICalSet
StringToNAErrorTerm2 Method

Description
Converts the returned strings from GetErrorTermList into the enumeration (NAErrorTerm2) and the port numbers required for PutErrorTerm and GetErrorTerm methods that transmit data in and out of the Cal Set.

Learn more about reading and writing Cal data using COM.

VB Syntax
```
Cal Set.StringToNAErrorTerm2 (list, eterm, rcv, src)
```

Variable
- **Cal Set (object)** - A Cal Set object
- **list (string)** - a string containing the textual description of the error term.
- **eterm (enum As NaErrorTerm2)**. Choose from:
  0 - naET_Directivity (rcv = src)
  1 - naET_SourceMatch (rcv = src)
  2 - naET_ReflectionTracking (rcv = src)
  3 - naET_TransmissionTracking (rcv != src)
  4 - naET_LoadMatch (rcv != src)
  5 - naET_Isolation (rcv != src)
- **rcv (long)** - port number of the receiver
- **src (long)** - port number of the source

Return Type
Not Applicable

Default
Not Applicable

Examples
```
CalSet.StringToNAErrorTerm2 str, term, rcv, src
```

C++ Syntax
```
HRESULT StringToNAErrorTerm2 (BSTR* str, NAErrorTerm2* item, long *rcv, long *src);
```

Interface
ICalSet
### SweepOnlyCalChannelDuringCalAcquisition Method

**Description**
Clears ALL flags for channels to sweep during calibration except the Cal channel. To flag a channel, see [AllowChannelToSweepDuringCalAcquisition Method](#).

**VB Syntax**
```vbnet
calMgr.SweepOnlyCalChannelDuringCalAcquisition
```

**Variable**

<table>
<thead>
<tr>
<th>(Type) - Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>calMgr (object) - A CalManager object</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Example**
```vbnet
calMgr.SweepOnlyCalChannelDuringCalAcquisition
```

See example using this command

**C++ Syntax**

```cpp
HRESULT SweepOnlyCalChannelDuringCalAcquisition()
```

**Interface**
ICalManager5

---

Last Modified:
8-Nov-2007    MX New topic
## TestsetCatalog Method

**Description**
Returns a list of supported testsets.

**VB Syntax**
```
data = Tsets.TestsetCatalog
```

**Variable**
- **Type**: Variant array
- **Description**: Variable to store the returned data.

- **Tsets**: Object - An `ExternalTestSets` collection

**Return Type**
Variant

**Default**
Not Applicable

**Examples**
```
value = Tsets.TestsetCatalog
```

**C++ Syntax**
```
HRESULT TestsetCatalog (VARIANT* Data);
```

**Interface**
`IE externaTestSets`
# UserPreset Method

**Description**
Performs a User Preset. There must be an active User Preset state file (see `UserPresetLoadFile` and `UserPresetSaveState`) or an error will be returned.

Regardless of the state of the User Preset Enable checkbox, the `app.Preset` command will always preset the PNA to the factory preset settings, and `app.UserPreset` will always perform a User Preset.

**VB Syntax**
`app.UserPreset`

**Variable**
*Type* - Description

- **app**
  - An `Application` *(object)*

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
`app.UserPreset`

**C++ Syntax**
`HRESULT UserPreset()`

**Interface**
`IAplication6`
UserPresetLoadFile Method

Description
Loads an existing instrument state file (.sta or .cst) to be used for User Preset. Subsequent execution of `app.UserPreset` will cause the PNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the `app.Preset` command will always preset the PNA to the factory preset settings, and `app.UserPreset` will always perform a User Preset.

VB Syntax
`app.UserPresetLoadFile (file)`

Variable (Type) - Description
- `app` An `Application` (object)
- `file` (String) Full path, name, and extension of the file to be loaded.

Return Type
Not Applicable

Default
Not Applicable

Examples
`app.UserPresetLoadFile ("C:\Program Files\Agilent\Network Analyzer\Documents\10MHzto20GHz.sta")`

C++ Syntax
`HRESULT UserPresetLoadFile (BSTR bstrFile)`

Interface `IApplication6`
# UserPresetSaveState Method

## Description
Saves the current instrument settings as UserPreset.sta. Subsequent execution of `app.UserPreset` will cause the PNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the `app.Preset` command will always preset the PNA to the factory preset settings, and `app.UserPreset` will always perform a User Preset.

## VB Syntax

```vb
app.UserPresetSaveState
```

## Variable (Type) - Description

- **app**  
  An `Application` (object)

## Return Type
Not Applicable

## Default
Not Applicable

## Examples

```vb
app.UserPresetSaveState
```

## C++ Syntax

```cpp
HRESULT UserPresetSaveState();
```

## Interface
`IAplication6`
### WriteData Method

**Description**
Writes a 13-bit value to the specified address using the AD0 through AD12 lines of the external test set connector. The PNA generates the appropriate timing signals. It automatically controls timing signals LDS, LAS and RLW to strobe the address, then the data, to the external test set. See the timing diagram for Address and Data I/O read.

**VB Syntax**
```
ExtIO.WriteData (address) = value
```

**Variable**

<table>
<thead>
<tr>
<th>(Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExtIO (object)</td>
<td>An External IO object</td>
</tr>
<tr>
<td>address (variant)</td>
<td>Address to be written to.</td>
</tr>
<tr>
<td>value (variant)</td>
<td>13-bit word to write</td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
ExtIO.WriteData (15) = 12
```

**C++ Syntax**
```
HRESULT WriteData(VARIANT Address,  VARIANT Data);
```

**Interface**
IHWExternaTestSetIO
**WriteRaw Method**

**Description**

Writes a 16-bit value to the external test set connector lines AD0 - AD12, RLW, LAS and LDS. The analyzer does NOT generate the appropriate timing signals. The user has control of all 16 lines using this write method.

**Note:** When RLW (pin 25) is set to 1 (high) it causes lines AD0 - AD12 to float. It disables their output latches and sets the hardware for reading. LDS and LAS are not affected by this behavior.

Below is the format of data that is written with WriteRaw:

* This Output will float if RLW (bit-13) is set high

<table>
<thead>
<tr>
<th>Pin</th>
<th>Bit</th>
<th>Signal name</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>0</td>
<td>AD0*</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>AD1*</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>AD2*</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>AD3*</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>AD4*</td>
</tr>
<tr>
<td>21</td>
<td>5</td>
<td>AD5*</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
<td>AD6*</td>
</tr>
<tr>
<td>19</td>
<td>7</td>
<td>AD7*</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>AD8*</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>AD9*</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>AD10*</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>AD11*</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>AD12*</td>
</tr>
<tr>
<td>25</td>
<td>13</td>
<td>RLW</td>
</tr>
<tr>
<td>24</td>
<td>14</td>
<td>LDS</td>
</tr>
<tr>
<td>8</td>
<td>15</td>
<td>LAS</td>
</tr>
</tbody>
</table>
**VB Syntax**  
`ExtIO.WriteRaw value`

**Variable**  
(Type) - Description

- `ExtIO` (object) - An External IO object
- `value` (variant) - Data to be written

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
`ExtIO.WriteRaw 12`

**C++ Syntax**  
`HRESULT WriteRaw( VARIANT Output );`

**Interface**  
IHWExternalTestSetIO
### WriteSnpFileWithSpecifiedPorts Method

**Description**  
Note: This command replaces `app.Save (.snp)`. This command is more explicit regarding the data to be saved, and works for PNAs with multiport test sets.

Saves SnP data to the specified file.  Learn more about SnP data.

**VB Syntax**  
```
data = meas.WriteSnpFileWithSpecifiedPorts ports, filename
```

**Variable**  
(Type) - Description

- **data** (Variant) array to store the data.
- **meas** A Measurement (object)
- **ports** (Variant Array) One dimensional array containing a list of port numbers for which snp data is requested.
- **filename** (string) - Full path, filename, and suffix to store the data.

  The suffix is not checked for accuracy. If saving 2 ports, specify "filename.s2p"; If saving 3 ports, specify "filename.s3p." and so forth.

SnP data can be output using several data formatting options. See SnPFormat Property

**Return Type** Variant array - automatically dimensioned to the size of the data.

**Default** Not Applicable

**Examples**

'This VBScript example can be pasted into a notepad file and run on the PNA as a macro. Learn how.

Set pna = CreateObject("AgilentPnA835x.application")
Set meas = pna.ActiveMeasurement

'List the port numbers for required data
ports = Array(1,2,4)

'specify where to save the data
filename="C:\Program Files\Agilent\Network Analyzer\Documents\MyData.s3p"
meas.WriteSnpFileWithSpecifiedPorts ports, filename

**C++ Syntax**  
```
HRESULT WriteSnpFileWithSpecifiedPorts(VARIANT portsToMeasure,BSTR filename);
```

**Interface** IMeasurement7

Last modified: 1780
9/18/06  MQ Added for multiport
OnCalEvent

Description
Triggers a calibration event. See a list of CAL Events.

Note: Some Severe Events are also used as Error Messages

VB Syntax
Sub app_OnCalEvent(ByVal eventID As Variant, ByVal chanNum As Variant, ByVal measNum As Variant)

Variable (Type) - Description

app An Application (object)

eventID Code number of the event which occurred

chanNum Channel Number of the event

measNum Measurement Number of the event

Return Type Not Applicable

Default Not Applicable

Examples
Sub pna_OnCalEvent(ByVal eventID As Variant, ByVal channelNumber As Variant, ByVal measurementNumber As Variant)
'
MsgBox ("A Calibration event has occurred")
End Sub

C++ Syntax
HRESULT OnCalEvent(VARIANT eventID, VARIANT channelNumber, VARIANT measurementNumber)

Interface IApplication

Selected Cal Events
512 naEventID_CAL_QUESTIONABLE
513 naEventID_CAL_STD_NEEDED
514 naEventID_CAL_STATE_NOT_HW_COMPATIBLE
515 naEventID_CAL_REQUIRED
516 naEventID_CAL_CORRECTION_TURNED_OFF
517 naEventID_CAL_CORRECTION_TURNED_OFF_INTERPOLATION_OFF
518 naEventID_CAL_CORRECTION_RESTORED
519 naEventID_CAL_CORRECTION_TURNED_OFF_FREQRANGE_EXCEEDED
naEventID_CAL_CALTYPE_SET_TO_NONE
naEventID_CAL_CORRECTION_TURNED_OFF_NOT_AN_SPARAM
naEventID_SOURCE_POWER_CAL_COMPLETED
naEventID_SOURCE_POWER_CAL_NOT_PRESENT
naEventID_SOURCE_POWER_CAL_NOT_COMPLETE
naEventID_SOURCE_POWER_CAL_REMOVE_TRACE
naEventID_SOURCE_POWER_CAL_REMOVE_MEAS
naEventID_SOURCE_POWER_CAL_POWER_CHANGED
naEventID_INSUFFICIENT_SLIDE_MOVEMENT
naEventID_CALSET_NOT_FOUND
naEventID_CALSET_CREATED
naEventID_CALSET_FILE_NOT_VALID
naEventID_CALSET_LOAD_FAILED
naEventID_CALSET_SAVE_FAILED
naEventID_CALSET_DELETED
naEventID_CALSET_FILE_NOT_COMPATIBLE
naEventID_NEW_CALSET_FILE_CREATED
naEventID_CAL_SET_IN_USE
naEventID_CAL_COULD_NOT_TURN_ON
naEventID_ERROR_FIXTURING_S2PFILE_CANNOT_OPEN
naEventID_ERROR_FIXTURING_TURNED_OFF
naEventID_MORE_THRU_PATHS_NEEDED

See Also
Errors and the SCPIStringParser Object
### OnChannelEvent

**About Analyzer Events**

#### Description
Triggered by a channel event.

#### VB Syntax
```vbnet
Sub app_OnChannelEvent(ByVal eventID As Variant, ByVal chanNum As Variant)
```

#### Variable (Type) - Description

- **app**  An Application (object)
- **eventID**  Code number of the event which occurred
- **chanNum**  Channel Number of the event

#### Return Type
Not Applicable

#### Default
Not Applicable

#### Examples
```vbnet
Sub pna_OnChannelEvent(ByVal eventID As Variant, ByVal channelNumber As Variant)
    If eventID=naEventID_CHANNEL_CREATED then
        MsgBox "Channel" + channelNumber + " was created"
    End If
End Sub
```

#### C++ Syntax
```cpp
HRESULT OnChannelEvent(VARIANT eventID, VARIANT channelNumber)
```

#### Interface
IApplication

### Selected Channel Events

- 1792 `naEventID_CHANNEL_SWEEP_COMPLETE`
- 1793 `naEventID_CHANNEL_TRIGGER_COMPLETE`
- 1796 `naEventID_SET_CHANNEL_DIRTY`
- 1797 `naEventID_CLEAR_CHANNEL_DIRTY`
- 1801 `naEventID_ALL_SWEEPS_COMPLETED_AND_PROCESSED`
- 1805 `naEventID_CHANNEL_CREATED`
- 1806 `naEventID_CHANNEL_DELETED`
- 1876 `naEventID_NO_SOURCE_ATTEN`
- 1879 `naEventID_FREQ_OFFSET_OVERRANGE_SO_TURNED_OFF`
- 1883 `naEventID_PORT_NUMBER_OUT_OF_RANGE`

### See Also

Errors and the SCPIStringParser Object
Last modified:

March 2, 2007    Added channel create and delete
Nov. 6, 2006    Added events
OnDisplayEvent

**Description**  
Triggered by a display event.

**VB Syntax**  
Sub **OnDisplayEvent**(ByVal eventID As Variant, ByVal winNum As Variant, ByVal traceNum As Variant)

**Variable**  
*Variable* (**Type**) - **Description**

- **app** An Application **(object)**
- **eventID** Code number of the event which occurred
- **winNum** Window Number of the event
- **traceNum** Trace Number of the event

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
Sub pna_OnDisplayEvent(ByVal eventID As Variant, ByVal windowNumber As Variant, ByVal traceNumber As Variant)  
MsgBox ("A Display event has occurred")  
End Sub

**C++ Syntax**  
HRESULT OnDisplayEvent(VARIANT eventID, VARIANT windowNumber, VARIANT traceNumber)

**Interface**  
IApplication

**Selected Display Events**

- **1541** **naEventID_PRINT_SETUP_FAILURE**
- **1542** **naEventID_PRINT_CANCELED**

**See Also**

- [Errors and the SCPIStringParser Object](#)

**Last modified:**

- Nov. 6, 2006  Added events
OnHardwareEvent

Description  Triggered by a hardware event. See a list of Hardware Events

Note: Some Severe Events are also used as Error Messages

VB Syntax  Sub app_OnHardwareEvent(ByVal eventID As Variant)

Variable  (Type) - Description

app  An Application (object)

eventID  Code number of the event which occurred

Return Type  Not Applicable

Default  Not Applicable

Examples  Private Sub pna_OnHardwareEvent(ByVal eventID As Variant)
MsgBox ("A Hardware event has occurred")
End Sub

C++ Syntax  HRESULT OnHardwareEvent(VARIANT eventID)

Interface  IApplication

Selected Hardware Events

848 naEventID_PHASELOCK
852 naEventID_RFPOWEROFF
853 naEventID_RFPOWERON
855 naEventID_UNLEVELED
857 naEventID_OVERLOAD
914 naEventID_TRIGGER_REQUIRES_EDGE_LEVEL_TRIGGER
915 naEventID_TRIGGER_REQUIRES_TRIGGER_OUT

See Also  Errors and the SCPIStringParser Object

Last modified:
### OnMeasurementEvent

**Description**
Triggered by a measurement event.

**VB Syntax**
Sub app_OnMeasurementEvent(ByVal eventID As Variant, ByVal measNum As Variant)

---

**Variable (Type) - Description**

- `app` An Application (object)
- `eventID` Code number of the event which occurred
- `measNum` Measurement Number of the event

**Return Type**
Not Applicable

**Default**
Not Applicable

---

**Examples**
Private Sub pna_OnMeasurementEvent(ByVal eventID As Variant, ByVal measurementNumber As Variant)
    MsgBox ("A Measurement event has occurred")
End Sub

---

**C++ Syntax**
HRESULT OnMeasurementEvent(VARIANT eventID, VARIANT measurementNumber)

**Interface**
IApplication

### Selected Measurement Events
- 1024 naEventID_NO_VALID_MEMORY_TRACE
- 1028 naEventID_LIMIT_FAILED
- 1029 naEventID_LIMIT_PASSED
- 1034 naEventID_MEMORY_NOT_SAVED
- 1035 naEventID_SET_AVERAGE_COMPLETE
- 1036 naEventID_CLEAR_AVERAGE_COMPLETE
- 1111 naEventID_MARKER_BANDWIDTH_NOT_FOUND
- 1112 naEventID_PEAK_NOT_FOUND
- 1113 naEventID_TARGET_VALUE_NOT_FOUND
See Also

Errors and the SCPIStringParser Object

Last modified:

Nov. 6, 2006  Added events
OnSCPIEvent

**Description**  
Triggered by a SCPI event.  

**Note:** Some Severe Events are also used as Error Messages

**VB Syntax**  
Sub `app_OnSCPIEvent(ByVal eventID As Variant)`

**Variable** (Type) - Description

- `app`  An Application (object)
- `eventID`  Code number of the event which occurred

**Return Type**  
Not Applicable

**Default**  
Not Applicable

**Examples**  
Private Sub pna_OnSCPIEvent(ByVal eventID As Variant)  
    MsgBox ("A SCPI event has occurred")  
    End Sub

**C++ Syntax**  
HRESULT OnSCPIEvent(VARIANT eventID )

**Interface**  
IApplication

**Selected SCPI Parser Events**

- 1281 `naEventID NOTHING_TO_SAY`
- 1284 `naEventID SCPI_STATUS_BYTE_CHANGE`
- 1360 `naEventID BAD_SCPI_EXECUTE`
- 1375 `naEventID CALC_MEASUREMENT_SET_TO_NONE`

**See Also**

Errors and the SCPIStringParser Object

Last modified:

Nov. 6, 2006  Added events
### About Analyzer Events

**OnSystemEvent**

**Description**
Triggered by a system event. See a list of [System Events](#), also known as general events.

See also [EnableSourceUnleveledEvents Property](#)

**Note:** Some Severe Events are also used as Error Messages

**VB Syntax**
```
Sub app_OnSystemEvent(ByVal eventID As Variant)
```

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>An Application</td>
<td>(object)</td>
</tr>
<tr>
<td>eventID</td>
<td>Code number of the event which occurred</td>
<td></td>
</tr>
<tr>
<td>chanNum</td>
<td>Channel Number of the event</td>
<td></td>
</tr>
</tbody>
</table>

**Return Type**
Not Applicable

**Default**
Not Applicable

**Examples**
```
Private Sub pna_OnSystemEvent(ByVal eventID As Variant)
    MsgBox ("A System event has occurred")
End Sub
```

**C++ Syntax**
```
HRESULT OnSystemEvent(VARIANT eventID)
```

**Interface**
IApplication

### Selected System Events

<table>
<thead>
<tr>
<th>Event ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2048</td>
<td>naEventID_OPTION_NOT_INSTALLED</td>
</tr>
<tr>
<td>2049</td>
<td>naEventID_FEATURE_NOT_AVAILABLE</td>
</tr>
<tr>
<td>2050</td>
<td>naEventID_FEATURE_NOT_VALID</td>
</tr>
<tr>
<td>2051</td>
<td>naEventID_SAVEFILE_OK</td>
</tr>
<tr>
<td>2063</td>
<td>naEventID_RECALLFILE_SUCCESS</td>
</tr>
<tr>
<td>2130</td>
<td>naEventID_PRINTER_TROUBLE</td>
</tr>
<tr>
<td>2133</td>
<td>naEventID_TRIGGERDENIED</td>
</tr>
<tr>
<td>2134</td>
<td>naEventID_MACRO_FAILED</td>
</tr>
</tbody>
</table>
2144 naEventID_NO_LICENSE
2163 naEventID_PRESET
2166 naEventID_TRIGGERFAILED

See Also
Errors and the SCPIStringParser Object

Last modified:

Nov. 6, 2006    Added events
<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Reserved for future use.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VB Syntax</strong></td>
<td>Sub app <strong>OnUserEvent</strong></td>
</tr>
</tbody>
</table>
Reading Cal Set Data using COM

This example iterates over the entire collection of Cal Sets that currently reside in the PNA. It reads the entire list of error term strings from each Cal Set and queries the data for each term. It then does the same for the standards data.

Learn more about Reading and Writing Calibration data using COM.

Learn more about Cal Sets.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as CalSets.vbs. Learn how to setup and run the macro.

```vbs
Dim pna
Dim cset
Dim calsets

' create the pna object
' to run on a remote PC, substitute 'name' for the full computer name of your PNA
' to run as a macro on the PNA, remove ,"name"
Set pna = CreateObject("AgilentPNA835x.Application", "name")
wscript.echo pna.IDString

' obtain the calset collection
Set calsets = pna.GetCalManager.calsets

' loop thru the calsets
Dim c
For c = 1 To calsets.count
Set cset = calsets.Item(c)

' wscript.echo prints values to a message box
wscript.echo "calset = ", cset.GetGUID, cset.Description

' iterate through error terms data
Dim vterms
vterms = cset.GetErrorTermList2(0, "")
if (Not IsEmpty(vterms)) then
For i = LBound(vterms) To UBound(vterms)
wscript.echo vterms(i)
vdata = cset.GetErrorTermByString(0, vterms(i) )
wscript.echo vdata(1,0), vdata(1,1)
Next
end if

' iterate through standards data
vterms = cset.GetStandardList2(""")
if (Not IsEmpty(vterms)) then
For i = LBound(vterms) To UBound(vterms)
end if
```

1797
wscript.echo vterms(i)
 vdata = cset.GetStandardByString( vterms(i) )
 wscript.echo vdata(1,0), vdata(1,1)
  Next
 end if
  Next
Getting Trace Data from the Analyzer

This Visual Basic program:

- Retrieves Scalar Data from the Analyzer and plots it.
- Retrieves Paired Data from the Analyzer and plots it.
- Retrieves Complex Data from the Analyzer and plots it.

To use this code, prepare a form with the following:

- Two MSCharts named **MSChart1** and **MSChart2**
- Three buttons named **GetScalar**, **GetPaired**, **GetComplex**

**Note:** You can get MSChart in Visual Basic by clicking **Project / Components / Microsoft Chart Control**

```
'Put this in a module
Public dlocation As NADataStore
Public numpts As Long
Public fmt As NADataFormat
Public app As Application
Public measData As IArrayTransfer
Public chan As Channel

Sub Form_Load()
    'Change analyzerName to your analyzer's full computer name
    Set app = CreateObject("AgilentPNA835x.Application", "analyzerName")
    Set measData = app.ActiveMeasurement
    Set chan = app.ActiveChannel

    'To pick a location to get the data from remove the comment from one of these
dlocation = naRawData
    'dlocation = naCorrectedData
    'dlocation = naMeasResult
    'dlocation = naRawMemory
    'dlocation = naMemoryResult
    'setup MSChart1 and MSChart2
    'right click on the chart and select:
    ' - line chart
    ' - series in rows
End Sub

Sub GetComplex_Click()
    ReDim Data(numpts) As NAComplex
    Dim Real(201) As Single
    Dim Imag(201) As Single
    numpts = chan.NumberOfPoints
```
'You cannot change the format of Complex Data
Call trigger
'get data
measData.GetNAComplex dlocation, numpts, Data(0)
'plot data
Dim i As Integer
For i = 0 To numpts - 1
Real(i) = Data(i).Re
Imag(i) = Data(i).Im
Next i
MSChart1 = Real()
MSChart2.Visible = True
MSChart2 = Imag()
Call Sweep
End Sub

Sub GetPaired_Click()
ReDim Real(numpts) As Single
ReDim Imag(numpts) As Single
numpts = chan.NumberOfPoints
' To pick a format, remove the comment from one of these
fmt = naLogMagPhase
'fmt = naLinMagPhase
Call trigger
'Get data
measData.getPairedData dlocation, fmt, numpts, Real(0), Imag(0)
'Plot Scalar
MSChart1 = Real()
MSChart2.Visible = True
MSChart2 = Imag()
Call Sweep
End Sub

Sub GetScalar_Click()
ReDim Data(numpts) As Single
numpts = chan.NumberOfPoints
'To pick a format remove the comment from one of these
fmt = naDataFormat_LogMag
'fmt = naDataFormat_LinMag
'fmt = naDataFormat_Delay
'fmt = naDataFormat_Phase
'fmt = naDataFormat_Real
'fmt = naDataFormat_Imaginary
Call trigger
'Get data
measData.GetScalar dlocation, fmt, numpts, Data(0)
'Plot Data
MSChart1 = Data()
MSChart2.Visible = False
Call Sweep
End Sub

Sub trigger()
    'The analyzer sends continuous trigger signals
    app.TriggerSignal = naTriggerInternal
    'The channel will only accept one, then go into hold
    'Sync true will wait for the sweep to complete
    sync=True
    chan.Single sync
End Sub

Sub Sweep()
    'The channel goes back to accepting all triggers
    chan.Continuous
End Sub
Perform a Guided Calibration using COM

This example uses the GuidedCalibration interface to perform either a 2-port or 4-port calibration. Learn more about Reading and Writing Calibration data using COM.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as Calibrate.vbs. Learn how to setup and run the macro.

```
Set pna = CreateObject("AgilentPNA835x.Application")
Set calMgr = pna.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = pna.ActiveChannel
chanNum = chan.ChannelNumber
' Initialize guided cal to be performed on the active channel.
' The boolean argument of True indicates to store the cal only
' in the channel's calibration register. If instead you wish
' to create a new calset that the new cal will get stored to,
' comment out this next line and uncomment the three lines below it.
guidedCal.Initialize chanNum, True
'Set calset = calMgr.CreateCalSet(chanNum)
'chan.SelectCalSet calset.GetGUID, True
'guidedCal.Initialize chanNum, False

' To perform 2-port cal, Uncomment the following
' Then comment the 4-port cal

' Do 2-port cal
'TwoPortGuidedCal

' Do 4-port cal
FourPortGuidedCal

Sub TwoPortGuidedCal()
' Select the connectors
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 male"
For i = 3 To pna.NumberOfPorts
guidedCal.ConnectorType(i) = "Not used"
Next
value = MsgBox("Connectors defined for Ports 1 and 2")
' Select the Cal Kit for each port being calibrated.
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"
```

1802
To use an ECal module instead, comment out the above two lines
and uncomment the following two lines.

Replace N4691-60004 with your own ECAL model followed by 'ECal'.
Your ECal module must already be connected to a PNA USB port.

```
guidedCal.CalKitType(1) = "N4691-60004 ECal"
guidedCal.CalKitType(2) = "N4691-60004 ECal"
```

Value = MsgBox("Cal kits defined for Ports 1 and 2")

Initiate the calibration and query the number of steps

```
umSteps = guidedCal.GenerateSteps
```

Measure the standards, compute and apply the cal

MeasureAndComplete(numSteps)

End Sub

Sub FourPortGuidedCal()

Select the connectors

```
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 female"
guidedCal.ConnectorType(3) = "APC 3.5 female"
guidedCal.ConnectorType(4) = "APC 3.5 female"
```

If a PNA which has more than 4 ports

```
For i = 5 To pna.NumberOfPorts
  guidedCal.ConnectorType(i) = "Not used"
Next
```

Value = MsgBox("Connectors defined for Ports 1 to 4")

Select the Cal Kit for each port being calibrated.

```
guidedCal.CalKitType(1) = "85052D"
guidedCal.CalKitType(2) = "85052D"
guidedCal.CalKitType(3) = "85052D"
guidedCal.CalKitType(4) = "85052D"
```

To use an ECal module instead, comment out the above four lines
and uncomment the following four lines.

Replace N4691-60003 with your own ECAL model followed by 'ECal'.
Your ECal module must already be connected to a PNA USB port.

```
guidedCal.CalKitType(1) = "N4431-60003 ECal"
guidedCal.CalKitType(2) = "N4431-60003 ECal"
guidedCal.CalKitType(3) = "N4431-60003 ECal"
guidedCal.CalKitType(4) = "N4431-60003 ECal"
```

Value = MsgBox("Cal kits defined for Ports 1 to 4")

Initiate the calibration

```
guidedCal.GenerateSteps
```

If your selected cal kit is not a 4-port ECal module which can
mate to all 4 ports at once, then you may want to choose which
thru connections to measure for the cal. You must measure at
least 3 different thru paths for a 4-port cal (for greatest
accuracy you can choose to measure a thru connection for all 6
pairings of the 4 ports). If you omit this command, the default
is to measure from port 1 to port 2, port 1 to port 3, and
port 1 to port 4. For this example we select to measure
from port 1 to port 2, port 2 to port 3, and port 2 to port 4.

```
portList = Array(1,2,2,3,2,4)
guidedCal.ThruPortList = portList
```

Re-generate the connection steps to account for the thru changes
numSteps = guidedCal.GenerateSteps
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Sub MeasureAndComplete(ByVal numSteps)
value = MsgBox("Number of steps is " + CStr(numSteps))

' Measure the standards
'The following series of commands shows that standards
can be measured in any order. These steps acquire
measurement of standards in reverse order.
'It is easiest to iterate through standards using
'a For-Next Loop.
For i = NumSteps To 1
step = "Step " + CStr(i) + " of " + CStr(numSteps)
strPrompt = guidedCal.GetStepDescription(i)
value = MsgBox(strPrompt, vbOKOnly, step)
guidedCal.AcquireStep i
Next

' Conclude the calibration
guidedCal.GenerateErrorTerms
MsgBox ("Cal is done!")
End Sub

Last Modified:

20-Jan-2007 Added any order to steps.
Perform an ECal using COM

This example uses the GuidedCalibration interface to perform a 2-port ECal calibration.

Learn more about Reading and Writing Calibration data using COM.

---

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as ECal.vbs. Learn how to setup and run the macro.

```vbs
Set pna = CreateObject("AgilentPNA835x.Application")
Set calMgr = pna.GetCalManager
Set guidedCal = calMgr.GuidedCalibration
Set chan = pna.ActiveChannel
chanNum = chan.ChannelNumber

' Initialize guided cal to be performed on the active channel.
' The boolean argument of True indicates to create a new calset
guidedCal.Initialize chanNum, True

' To perform 3-port cal, Uncomment the following
' Then comment the 2-port cal

' Do 2-port cal
TwoPortGuidedCal

' Do 3-port cal
' ThreePortGuidedCal

Sub TwoPortGuidedCal()

' Change the following to match the connectors on your ECal module
guidedCal.ConnectorType(1) = "APC 3.5 female"
guidedCal.ConnectorType(2) = "APC 3.5 female"
For i = 3 To pna.NumberOfPorts
    guidedCal.ConnectorType(i) = "Not used"
Next

value = MsgBox("Connectors defined for Ports 1 and 2")

' Select the ECal module for each port being calibrated.
' Replace N4691-60004 with your own ECAL model followed by 'ECal'.
' Your ECal module must already be connected to a PNA USB port.

guidedCal.CalKitType(1) = "85092-60010 ECal"
guidedCal.CalKitType(2) = "85092-60010 ECal"

value = MsgBox("ECal Module defined for Ports 1 and 2")

' Initiate the calibration and query the number of steps
numSteps = guidedCal.GenerateSteps

' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub
```
Sub ThreePortGuidedCal()
    'Change the following to match the connectors on your ECal module
    guidedCal.ConnectorType(1) = "APC 3.5 female"
    guidedCal.ConnectorType(2) = "APC 3.5 female"
    guidedCal.ConnectorType(3) = "APC 3.5 female"
    
    ' Select the ECal module for each port being calibrated.
    ' Replace N4691-60003 with your own ECAL model followed by 'ECal'.
    ' Your ECal module must already be connected to a PNA USB port.
    guidedCal.CalKitType(1) = "N4431-60003 ECal"
    guidedCal.CalKitType(2) = "N4431-60003 ECal"
    guidedCal.CalKitType(3) = "N4431-60003 ECal"
    value = MsgBox("Cal kits defined for Ports 1 to 3")
    
    ' Initiate the calibration
    numSteps = guidedCal.GenerateSteps
    
    ' Measure the standards, compute and apply the cal
    MeasureAndComplete(numSteps)
End Sub

Sub MeasureAndComplete(ByVal numSteps)
    value = MsgBox("Number of steps is " + CStr(numSteps))
    
    ' Measure the standards
    For i = 1 To numSteps
        step = "Step " + CStr(i) + " of " + CStr(numSteps)
        strPrompt = guidedCal.GetStepDescription(i)
        value = MsgBox(strPrompt, vbOKOnly, step)
        guidedCal.AcquireStep i
    Next
    
    ' Conclude the calibration
    guidedCal.GenerateErrorTerms
    MsgBox("Cal is done!")
End Sub
Perform a Source Power Cal using COM

This program can be run in either Visual Basic 6 or as a VBScript program. The PNA can run *.vbs programs as macros.

This program demonstrates:

- Performing a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

Learn more about Power Calibrations
See an example that Uploads a Source Power Cal

See Other COM Example Programs

To run this program, you need:

- One of the following power meters connected to the PNA through GPIB: E4416A, E4417A, E4418A/B, E4419A/B, 437B, 438A, EPM-441A, EPM-442A

Note: If your power meter is other than these, you can create your own Power Meter Driver using our template.

- Your PC and PNA both connected to a LAN (for communicating with each other).

To make this program work in VBS, save the following code in a text editor file such as Notepad and save as *.vbs.

To make this program work in Visual Basic 6:

1. Create a new project
2. Click Project, Add New Module, click Open.
3. Paste the following code into the code window.
4. Delete the first two lines (comment and Main)
5. Click Project, Properties. Under Startup Object, select Sub Main
6. Click Project, References, and select the Agilent PNA Series Type Library.

```vbnet
Public Sub Main()
    Dim PNA, chan, pwrcal ' PNA COM objects
    Const naPowerMeter = 0, naPowerMeterAndReceiver = 1 ' enum NASourcePowerCalMethod
    Const naPowerSensor_A = 0 ' enum NAPowerAcquisitionDevice
    Const naCorrectionValues = 0 ' enum NASourcePowerCalBuffer
```
Const port = 2 ' PNA port #2 as source port
Const offset = 0 ' cal power offset value
Const bDisplay = True ' whether to display data during acquire
Dim stimulus, calvalues, strResult

' Instantiate our PNA COM objects
Set PNA = CreateObject("AgilentPNA835x.Application")
Set chan = PNA.Channels(1)
Set pwrcal = PNA.SourcePowerCalibrator

' Set the number of sweep points to 21 on Channel 1.
chan.NumberOfPoints = 21

' Specify the GPIB address of the power meter
' that will be used in performing the calibration.
pwrcal.PowerMeterGPIBAddress = 13

' Turn use of the loss table OFF (this assumes there is
' virtually no loss in the RF path to the power sensor
' due to a splitter, coupler or adapter).
pwrcal.UsePowerLossSegments = False

' Turn frequency checking OFF (so one power sensor is used for the entire cal
' acquisition sweep regardless of frequency span).
pwrcal.UsePowerSensorFrequencyLimits = False

' Specify a nominal power accuracy tolerance (IterationsTolerance) in dB for the
' calibration, and the maximum number of iterations to adjust power at each point,
' attempting to achieve within tolerance of the desired power. If at any stimulus
' point the power fails to reach within the set tolerance of the desired power
' after the maximum number of iterations, the power at that point will be set to the
' value determined by the last iteration (the Source Power Cal dialog box will
' indicate the FAIL, but we can still apply the cal if desired when it's complete).
' Each iteration is based upon a SETTLED power reading (see comments preceding the
' next two properties below).
pwrcal.IterationsTolerance = 0.1
pwrcal.MaximumIterationsPerPoint = 3

' The worst-case window of power uncertainty (for a calibration which meets
' tolerance) is the sum of the iteration tolerance and the power meter settling
' tolerance (which is described below).
' At each stimulus point, the PNA takes power meter readings and determines when
' they have settled by comparing the magnitude difference between consecutive
' readings versus a nominal dB tolerance limit (ReadingsTolerance) on that magnitude
difference. When consecutive readings are within tolerance of each other, or
' if they are not within tolerance but we've taken a maximum number of readings
' (ReadingsPerPoint), the PNA does a weighted average of the readings taken at that
' stimulus point and that is considered our settled power reading.
pwrcal.ReadingsTolerance = 0.1
pwrcal.ReadingsPerPoint = 5
Setup of information pertaining to this specific cal acquisition. Includes the method (type of devices) that will be used to perform the cal -- choose either naPowerMeter or naPowerMeterAndReceiver. naPowerMeterAndReceiver uses the power meter for the first iteration of each point and the PNA's reference receiver for subsequent iterations, so is much faster than using power meter only naPowerMeter).

But the power meter accounts for compression when calibrating at the output of an active device, whereas the reference receiver cannot unless it is coupled to the cal reference plane (on a PNA which allows direct access to the receivers).

'offset' specifies if the cal power level is offset (positive value for a gain, negative value for a loss) from the PNA port power setting on the channel when no source power cal is active. This is to account for components between the PNA test port and cal reference plane. In this example, we will calibrate at the PNA test port, so there is no offset (it is zero).

'bDisplay' indicates whether to display the source power cal dialog during the source power cal acquisition (the dialog will chart the corrected power readings).

\[
pwrcal.SetCalInfo2 naPowerMeter, chan.channelNumber, port, offset, bDisplay
\]

Perform synchronous source power cal acquisition sweep using the sensor attached to Channel A of the power meter. This assumes that the power sensor is already connected to Port 2 of the PNA.

\[
pwrcal.AcquirePowerReadings naPowerSensor_A, True
\]

Conclude the calibration. This applies the cal data to PNA channel memory, and turns the correction ON for Port 2 on Channel 1, but does NOT save the calibration.

\[
pwrcal.ApplyPowerCorrectionValues
\]

At this point, if you choose to save the instrument state as a '.CST' file, the calibration will be saved with the instrument state in that file.

Read the stimulus values from Channel 1.

\[
stimulus = chan.GetXAxisValues
\]

Read the source power correction data.

\[
calvalues = chan.getSourcePowerCalDataEx(naCorrectionValues, port)
\]

Print the data using a message box (here, Chr returns the ASCII characters for Tab (9) and Linefeed (10)).

\[
strResult = "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
\]

\[
For i = 0 To UBound(stimulus)
\]

\[
strResult = strResult & stimulus(i) & Chr(9) & calvalues(i) & Chr(10)
\]

\[
Next
\]

\[
MsgBox strResult
\]

End Sub
Perform an Unguided Cal using COM

This example uses the ICalibrator interface to do the following:

- perform a two port calibration
- retrieve the error term data
- retrieve the standard data (cal acquisition data)

Learn more about Reading and Writing Calibration data using COM.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as Calibrate.vbs. Learn how to setup and run the macro.

dim pna
' To run from an external PC, substitute your PNA Name and use the following command.
' set pna = CreateObject("AgilentPNA835x.Application", "PNA Name")
set pna = CreateObject("AgilentPNA835x.Application")
dim calibrator
set calibrator=pna.activechannel.calibrator

wscript.echo "setcalinfo for two port cal"
calibrator.setcalinfo 5,1, 2

' only have one set of standards
calibrator.Simultaneous2PortAcquisition = false

'first acquire forward reflection standards, then reverse
dim p
for p = 1 to 2
    if (p = 1) then
        calibrator.AcquisitionDirection = 0
    else
        calibrator.AcquisitionDirection = 1
    end if
    wscript.echo "connect open to port ", p
    calibrator.acquirecalstandard 1
    wscript.echo "connect short to port ", p
    calibrator.acquirecalstandard 2
wscript.echo "connect load to port ", p
calibrator.acquirecalstandard 3
next
wscript.echo "connect a thru1"
calibrator.acquirecalstandard 4

'Optional - perform isolation
wscript.echo "connect loads to both ports"
calibrator.acquirecalstandard 5
wscript.echo "calculating"
calibrator.CalculateErrorCoefficients

'Calibration complete
' Now read error terms and standard data
dim termName
termName = Array("Directivity", "SourceMatch", "ReflectionTracking")
dim vardata

' iterate over error terms
dim t
for t = 0 to 2 ' per error term
    for p = 1 to 2 ' per port
        wscript.echo "Requesting ", termName(t), p, p
        vardata = calibrator.GetErrorTerm(t, p, p)
    next
next

' now get the path terms: iterator each one request
termName = Array("Isolation", "LoadMatch", "TransmissionTracking")
for t = 0 to 2
    wscript.echo "Requesting Forward term", termName(t), 1, 2
    vardata = calibrator.GetErrorTerm(t, 1, 2)
    wscript.echo "Requesting Reverse Term", termName(t), 2, 1
    vardata = calibrator.GetErrorTerm(t, 2, 1)
next
dim stdname
stdname = Array("", "Open", "Short", "Load", "Thru", "Isolation")

' iterate over the port standards
for t = 1 to 3
    for p = 1 to 2
        wscript.echo "Requesting", stdname(t), p, p
        vardata = calibrator.GetStandard(t, p, p)
next

next

' now get the path standards: iterator each one request
for t = 4 to 5

wscript.echo "Requesting Forward", stdname(t), 1, 2
vardata = calibrator.GetStandard(t, 1, 2)

wscript.echo "Requesting Reverse", stdname(t), 2, 1
vardata = calibrator.GetStandard(t, 2, 1)

next
Perform an Unknown Thru or TRL Cal

The following program performs either a 2-port SOLT Unknown Thru Cal or a 2-port TRL Cal. The 85052C Cal Kit used in this program contains both types of standards. This program can be run on 2-port or 4-port PNAs. When run on a multiport (4 or more ports) PNA, which does not have a reference receiver per port, a Delta Match Cal is required. See example of Delta Match Cal.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unknown.vbs. Learn how to setup and run the macro.

```
' PerformUnknownThruOrTRLCal
Sub PerformUnknownThruOrTRLCal()
  ' Create / Get the PNA application.
  Set app = CreateObject("AgilentPNA835x.Application")
  ' Get the cal manager object
  Set calMgr = app.GetCalManager
  ' Get the guided cal object
  Set guidedCal = calMgr.GuidedCalibration
  Set chan = app.ActiveChannel
  chanNum = chan.ChannelNumber

  ' Initialize guided cal to be performed on the active channel.
  ' The boolean argument of True specifies the creation of a new calset
  ' for storing the new calibration.
  guidedCal.Initialize chanNum, True

  ' Specify connectors for Ports 1 and 2
  guidedCal.ConnectorType(1) = "APC 3.5 female"
  guidedCal.ConnectorType(2) = "APC 3.5 male"

  ' If your PNA has more than 2 ports, uncomment one or both of
  ' these next two lines, to explicitly specify this is
  ' just a 2-port cal.
  ' guidedCal.ConnectorType(3) = "Not used"
  ' guidedCal.ConnectorType(4) = "Not used"

  ' Specify cal kit for Ports 1 and 2
  guidedCal.CalKitType(1) = "85052C"
  guidedCal.CalKitType(2) = "85052C"

  ' Since the 85052C cal kit contains SOLT standards and also TRL
  ' standards, these next lines determine whether the cal becomes
  ' unknown thru (SOLT), or TRL.
  ' Specify cal and Thru method
  guidedCal.PathCalMethod (1,2) = "SOLT"
  guidedCal.PathThruMethod (1,2) = "Undefined Thru"

  ' To set up the cal as TRL, comment the previous line and uncomment this next line.
  ' The Thru method is set by default.
  guidedCal.PathCalMethod (1,2) = "TRL"
```

1813
' Initiate the calibration and query the number of steps
numSteps = guidedCal.GenerateSteps
MsgBox "Number of steps is " + CStr(numSteps)

' Query the list of ports that need delta match
portList = guidedCal.PortsNeedingDeltaMatch
' If portList contains just one element and it's value is 0, then that indicates
' none of the ports being calibrated require delta match data.
' If each testport on the PNA has it's own reference receiver (R channel),
' then delta match is never needed, so portList will always be just 0.
lowerBound = LBound(portList)
If (UBound(portList) <> lowerBound) Or (portList(lowerBound) <> 0) Then
  ' Delta match data is required for at least one port.
  ' For this example, we assume a Global Delta Match Cal has previously been
  ' performed so the Global Delta Match CalSet exists.
  ' Supplying an empty string to ApplyDeltaMatchFromCalSet indicates to use
  ' the Global Delta Match CalSet.
  MsgBox "here"
guidedCal.ApplyDeltaMatchFromCalSet ""
End If

' Measure the standards
For i = 1 To numSteps
  step = "Step " + CStr(i) + " of " + CStr(numSteps)
  strPrompt = guidedCal.GetStepDescription(i)
  retVal = MsgBox(strPrompt, vbOKCancel, step)
  If retVal = vbCancel Then Exit Sub
  guidedCal.AcquireStep i
Next

' Conclude the calibration
guidedCal.GenerateErrorTerms
MsgBox "Cal is done!"

End Sub
Perform Global Delta Match Cal

The following program performs a Global Delta Match Calibration. This is required when performing an Unknown Thru cal or TRL cal on PNAs without a reference receiver for each test port. See example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Delta.vbs. Learn how to setup and run the macro.

Sub PerformGlobalDeltaMatchCal()
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
' Get cal manager object
Set calMgr = app.GetCalManager
' Get guided cal object
Set guidedCal = calMgr.GuidedCalibration

' Initiate a Global Delta Match calibration, choosing connector and cal kit
numSteps = guidedCal.GenerateGlobalDeltaMatchSequence("APC 3.5 female", "85033D/E")
MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards
For i = 1 To numSteps
    step = "Step " + CStr(i) + " of " + CStr(numSteps)
    strPrompt = guidedCal.GetStepDescription(i)
    retVal = MsgBox(strPrompt, vbOKCancel, step)
    If retVal = vbCancel Then Exit Sub
    guidedCal.AcquireStep i
Next

' Conclude the calibration
guidedCal.GenerateErrorTerms
MsgBox "Cal is done!"
End Sub
Writing Cal Set Data using COM

This example creates a Cal Set and then writes data to the Cal Set.
Learn more about Reading and Writing Calibration data using COM.
Learn more about Cal Sets.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as CalSetsWrite.vbs. Learn how to setup and run the macro.

dim pna
dim v
Set pna=CreateObject("AgilentPNA835x.Application")
InitPhonyData
PutPhonyData
' This sub creates phony data
Sub InitPhonyData()
Dim i
Dim numpts
wscript.echo "init phony"
umpts = pna.ActiveChannel.NumberOfPoints
ReDim v(numpts - 1, 1)
For i = 0 To numpts - 1
 v(i, 0) = i
 v(i, 1) = 0
Next
End Sub

' This sub creates a Cal Set, then writes the phony data to it
Sub PutPhonyData()
Dim cmgr
Dim cset
wscript.echo "putphony"
Set cmgr = pna.GetCalManager
Set cset = cmgr.CreateCalSet(1)
cset.OpenCalSet naCalType_OnePort, 1
cset.putErrorTerm directivity,1, 1, v
cset.putErrorTerm sourcematch,1, 1, v
cset.putErrorTerm reflectiontracking ,1, 1, v
cset.CloseCalSet
cset.Description = "Phony One Port"
cset.save
End Sub
Upload a Source Power Cal using COM

This program can be run in either Visual Basic 6 or as a VBScript program. The PNA can run *.vbs programs as macros.

This program demonstrates:

- Uploading a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

Learn more about Power Calibrations

See Other COM Example Programs

To run this program you need:

- Your PC and PNA both connected to a LAN (for communicating with each other).

To make this program work in VBS, save the following code in a text editor file such as Notepad and save as *.vbs.

To make this program work in Visual Basic 6:

1. Create a new project
2. Click Project, Add New Module, click Open.
3. Paste the following code into the code window.
4. Delete the first two lines (comment and Main)
5. Click Project, Properties. Under Startup Object, select Sub Main
6. Click Project, References, and select the Agilent PNA Series Type Library.

Public Sub Main()
    Dim PNA, chan ' PNA COM objects
    Const naCorrectionValues = 0 ' enum NASourcePowerCalBuffer
    Const port = 2 ' PNA port #2 as source port
    Dim stimulus, calvalues
    Dim power, calpower, strResult
    ' Instantiate our PNA COM objects
    Set PNA = CreateObject("AgilentPNA835x.Application")
    Set chan = PNA.Channels(1)
' Set the number of sweep points to 2 on Channel 1.
chan.NumberOfPoints = 2

' Ensure there's currently no source power cal on for this channel and port.
chan.SourcePowerCorrection(port) = False

' Specify if the cal power level is offset (positive value for a gain, negative
' value for a loss) from the PNA port power setting on the channel when
' no source power cal is active. This is to account for components
' between the PNA test port and cal reference plane.
' In this example, let's set up our calibration
' at the output of an amplifier with 15 dB gain.
chan.SourcePowerCalPowerOffset(port) = 15

' Send our source power correction data to the PNA. For purpose of simplicity
' in this example, we'll set up for no correction (0) at our start stimulus and
' 0.5 dB at our stop stimulus (recall that our sweep currently has just 2 points).
' calvalues = Array(0, 0.5)
chan.putSourcePowerCalDataEx naCorrectionValues, port, calvalues

' Set the number of sweep points to 21 on Channel 1.
chan.NumberOfPoints = 21

' Read the fixed power level for this port on Channel 1.
power = chan.TestPortPower(port)

' Turn the source power cal on.
chan.SourcePowerCorrection(port) = True

' Again read the fixed power level for this port on Channel 1
' (with our calibration turned on, this should now include the 15 dB offset
' we indicated our power amplifier provides).
calpower = chan.TestPortPower(port)

' Read the stimulus values from Channel 1.
stimulus = chan.GetXAxisValues

' Read back the source power correction data, now interpolated for 21 points
calvalues = chan.getSourcePowerCalDataEx(naCorrectionValues, port)

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).
strResult = "PNA port power = " & power & Chr(10)
strResult = strResult & "Power at reference plane = " & calpower & Chr(10) & hr(10)
strResult = strResult & "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(stimulus)
strResult = strResult & stimulus(i) & Chr(9) & calvalues(i) & Chr(10)
Next
MsgBox strResult
End Sub
Upload Segment Table

This example program uses the SetAllSegments_Method to do the following:

- creates a 2-dimensional array (7 x 10) 7 data elements that define each segment x 10 segments
- uploads the data to the PNA

This program does not make sweep type = segment or show the segment table.
The comments indicate the order in which the segment elements are specified: Index 0 - segment state, Index 4 is IFBW, and so forth.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as *.vbs. Learn how to setup and run the macro.

```vbscript
' Create the application instance, and preset the application
Set app = CreateObject("AgilentPNA835x.Application")
app.Preset

Dim chan
Set chan = app.ActiveChannel
chan.sweeptype = 4

Dim segs
Set segs = chan.Segments

Dim win
Set win = app.NAWindows(1)
win.ShowTable 2

' Multipliers
kHz = 1000
MHz = kHz*1000
GHz = MHz*1000

' Create segments from 10MHz to 3GHz
StartFreq = 10 * MHz
StopFreq = 3 * GHz
'*
'* Create 10 segments between StartFreq and StopFreq
'*
' Create a 2-D array of segments.
' 1st dimension is size 7 (6 is max index)
' to hold all the data per segment.
' 2nd dimension is size 10 (9 is max index)
```
' to hold 10 total segments.
Dim segdata(6, 9)
' Width of frequency segment, used below
SegmentWidth = (StopFreq - StartFreq) / 10
' Fill up all 10 segments (indices 0 to 9) with data
For i = 0 To 9
' element 0=segment state (on or off)
segdata(0, i) = True
' element 1=Num Points in this segment
segdata(1, i) = 500
' element 2=Start Freq
segdata(2, i) = StartFreq + i * SegmentWidth
' element 3=Stop Freq
segdata(3, i) = segdata(2, i) + SegmentWidth
' element 4=IFBW
segdata(4, i) = 35000
' element 5=Dwell Time
segdata(5, i) = 0
' element 6=Power
segdata(6, i) = 0
Next
' Configure Independent segment settings
segs.IFBandwidthOption = 1
segs.SourcePowerOption = 1
' Push the segment data into the PNA's Active Channel
segs.SetAllSegments segdata
Create and Cal an SMC Measurement

This example creates and calibrates an SMC measurement. To run this example without modification you need the following:

- A Mixer setup file saved on the PNA: C:\Program Files\Agilent\Network Analyzer\Documents\Mixer\MyMixer.mxr.
- If the mixer file uses an external LO source, it must be connected and configured.
- An ECal module that covers the frequency range of the measurement.
- A power meter must be connected to the PNA using GPIB.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as SMC.vbs. Learn how to setup and run the macro.

```vbs
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
Dim Meas
Set Meas = App.ActiveMeasurement
Meas.Delete
App.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","SC21"
'Other valid strings that can be specified to create a measurement with a parameter 'other than"SC21" are: "S11", "S22", "IPwr", and "OPwr"
Set Meas = App.ActiveMeasurement
'You can perform mixer setup here or
'recall a previous mixer setup from the PNA Hard drive
Dim mix
Set mix = Meas
'Load your own SMC measurement from the PNA Hard drive.
Meas.LoadFile "C:\Program Files\Agilent\Network Analyzer\Documents\Mixer\MyMixer.mxr"
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim SMC
Set SMC = CalMgr.CreateCustomCal("SMC")
SMC.Initialize 1, 1
SMC.Do2PortEcal = 1 'specify 0 for mechanical cal, 1 for ecal
'use Factory Characterization
SMC.ECALCharacterization(1) = 0
```
only specify the ThruCalMethod if you have a non-insertable DUT

' SM.C.ThruCalMethod = "Adapter Removal"
'
If you specify Adapter Removal or Unknown Thru calibration
then you need to specify the connector types of your DUT.

'Specify a connector that is the same type as your selected
ECAL characterization. The characterization selected in
this case is APC 2.4 female APC 2.4 male so the connectors
specified for my DUT have to be APC 2.4 but can be any sex.

SM.C.ConnectorType(1) = "APC 2.4 female"
SM.C.ConnectorType(2) = "APC 2.4 female"
SM.C.OmitIsolation = 1
SM.C.AutoOrient = 1

' 1- forward, 2-reverse, or Both
SM.C.CalibrationPort = "1"

Dim steps
steps = SM.C.GenerateSteps
For i = 1 To steps
MsgBox SM.C.GetStepDescription(i)
SM.C.AcquireStep i
Next

Dim calset
calset = SM.C.GenerateErrorTerms
MsgBox("SMC Cal Complete!")
Create and Cal a VMC Measurement

The following VMC calibration options are presented in VB Script examples:

- **Full 2-port ECAL**
- **For Mixer Characterization ONLY**
- **Mechanical Calibration**

**Note:** Each of the following programs load a mixer setup file. Substitute "Mixer009.mxr" with your own .mxr file.

### Full 2-port ECAL

```vbnet
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
Dim Meas
Set Meas = App.ActiveMeasurement
Meas.Delete
App.CreateCustomMeasurementEx ( 2, "Vector Mixer/Converter", "VC21" )
'Other valid strings that can be specified to create a measurement with a parameter other than "VC21" are: "S11", and "S22"
Set Meas = App.ActiveMeasurement
Meas.LoadFile "C:\Program Files\Agilent\Network Analyzer\Documents\mixer\Mixer009.mxr"

Dim CalMgr
Set CalMgr = App.GetCalManager
Dim VMC
Set VMC = CalMgr.CreateCustomCal("VMC")
VMC.Initialize 1, 1
VMC.Do2PortEcal = 1
VMC.Do1PortEcal = 1
VMC.ECALCharacterization(1, 1) = 0 'APC 2.4 male APC 2.4 female
VMC.ECALCharacterization(1, 0) = 0 
' could be Default, Flush Thru, Unknown Thru, or Adapter Removal
VMC.ThruCalMethod = "Adapter Removal"
VMC.OmitIsolation = 1
VMC.AutoOrient = 1
VMC.EcalOrientation1Port(1) = "A1"
VMC.CharacterizeMixerOnly = 0
VMC.LoadCharFromFile = 0
' to load mixer characterization from file, specify LoadCharFromFile = 1,
' then VMC.CharFileName = "C:\Program Files\Agilent\Network Analyzer\Documents\YourFile.s2p" (specify your own .s2P filename)
VMC.ConnectorType(1) = "APC 2.4 female"
```
VMC.ConnectorType(2) = "APC 2.4 female"

Dim steps
steps = VMC.GenerateSteps
For i = 1 To steps
MsgBox VMC.GetStepDescription(i)
VMC.AcquireStep i
Next

Dim calset
calset = VMC.GenerateErrorTerms

**Mixer Characterization ONLY**

Dim App
Set App = CreateObject("AgilentPNA835x.Application")
Dim Meas
Set Meas = App.ActiveMeasurement
Meas.Delete
app.CreateCustomMeasurementEx ( 2, "Vector Mixer/Converter", "VC21" )
'Other valid strings that can be specified to create a measurement with a parameter other than "VC21" are: "S11", and "S22"

Set Meas = App.ActiveMeasurement
Meas.LoadFile "C:\Program Files\Agilent\Network Analyzer\Documents\mixer\Mixer009.mxr"

Dim CalMgr
Set CalMgr = App.GetCalManager
Dim VMC
Set VMC = CalMgr.CreateCustomCal("VectorMixerCal.VMCTYPE")
VMC.Initialize 1, 1
VMC.Do2PortEcal = 1
VMC.Do1PortEcal = 1
VMC.ECALCharacterization(1, 1) = 0
VMC.ECALCharacterization(1, 0) = 0
VMC.ThruCalMethod = "Default"
VMC.OmitIsolation = 1
VMC.AutoOrient = 1
VMC.EcalOrientation1Port(1) = "A1"
VMC.CharacterizeMixerOnly = 1
VMC.LoadCharFromFile = 0
VMC.CharFileName = "C:\Program Files\Agilent\Network Analyzer\Documents\MyMixerChar.s2p"

Dim steps
steps = VMC.GenerateSteps
For i = 1 To steps
MsgBox VMC.GetStepDescription(i)
VMC.AcquireStep i
Next
Dim calset
calset = VMC.GenerateErrorTerms
MsgBox VMC.CharFileName

**Mechanical Calibration**

Dim App
Set App = CreateObject("AgilentPNA835x.Application")
Dim Meas
Set Meas = App.ActiveMeasurement
Meas.Delete
app.CreateCustomMeasurementEx ( 2, "Vector Mixer/Converter", "VC21" )
'Other valid strings that can be specified to create a measurement with a parameter other than "VC21" are: "S11", and "S22"

Set Meas = App.ActiveMeasurement
Meas.LoadFile "C:\Program Files\Agilent\Network Analyzer\Documents\mixer\Mixer009.mxr"
Dim CalMgr
Set CalMgr = App.GetCalManager
Dim VMC
Set VMC = CalMgr.CreateCustomCal("VectorMixerCal.VMCTYPE")
VMC.Initialize 1, 1
VMC.Do2PortEcal = 1
VMC.Do1PortEcal = 0
VMC.ECALCharacterization(1, 1) = 0
VMC.ThruCalMethod = "Default"
VMC.OmitIsolation = 1
VMC.AutoOrient = 1
VMC.CharacterizeMixerOnly = 0
VMC.LoadCharFromFile = 0
VMC.ConnectorType(1) = "APC 3.5 female"
VMC.ConnectorType(2) = "APC 3.5 male"
VMC.CalKitType(1) = "85033D/E"
VMC.CalKitType(2) = "85033D/E"

Dim steps
steps = VMC.GenerateSteps
For i = 1 To steps
MsgBox VMC.GetStepDescription(i)
VMC.AcquireStep i
Next

Dim calset
calset = VMC.GenerateErrorTerms
Create an SMC Fixed Output Measurement with COM

This VBScript example creates a calibrated SMC fixed output measurement using a controlled LO. Then a single sweep is taken and data is retrieved.

Fixed output measurements are only supported on SMC (not VMC) measurements. Fixed output measurements require that an external LO source be sweeping and synchronized with the PNA source. FCA can perform this synchronization using the external source configuration settings. The fastest, and recommended, method of controlling the LO source is Hardware List (BNC) triggering mode. However, in this mode, FCA channels will not respond to manual triggers. Therefore, the following example uses the "Single" IChannel::Single mechanism to trigger a sweep.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as SMC.vbs. Learn how to setup and run the macro.

```vb
option explicit
dim app
set app = createobject("agilentpna835x.application")
app.preset
' Put the channel in hold (highly recommended)
app.ActiveChannel.Hold 1

' Delete the standard measurement
app.ActiveMeasurement.Delete

' Create an SC21 measurement
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","SC21"

' Set the number of points to 11
app.ActiveChannel.NumberOfPoints = 11

' Setup the mixer parameters for a swept LO, fixed output measurement

dim mixer
set mixer = app.ActiveMeasurement
mixer.InputStartFrequency = 200e6
mixer.InputStopFrequency = 700e6
mixer.LORangeMode(1) = 0 ' 0 = Swept mode
mixer.OutputFixedFrequency = 3.4e9
mixer.InputPower = -17
mixer.LOPower(1) = 10

' The CALC method calculates the LO frequency from the other parameters, It also applies ALL mixer parameters to the channel.
mixer.Calculate 3 ' Calculate the LO range

' Create an S11 in the same channel
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter","S11"
dim S11Meas
set S11Meas = app.ActiveMeasurement
```
Create an IPwr in the same channel
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter", "S22"

Create an OPwr in the same channel
app.CreateCustomMeasurementEx 1, "Scalar Mixer/Converter", "OPwr"

Perform a single sweep synchronously.
app.ActiveChannel.Single 1

function ToString(complexDataArray)
dim dataAsString

dim point
for point = 0 to UBound(data)
dataAsString = dataAsString & "(" & data(point,0) & "," & data(point,1) & ")" 
next
ToString = dataAsString
end function

Retrieve the SC21 data
dim data
'Get the calibrated real/imaginary values
data = mixer.GetData(1,3)
wscript.echo ToString(data)

Retrieve the S11 data
'Get the calibrated real/imaginary values
data = S11Meas.GetData(1,3)
wscript.echo ToString(data)
Create a Pulsed Measurement

**Note:** This example applies only to the E836x Opt H11. For PNA-X see [PNA-X Create a Narrowband Pulsed Measurement](#).

The following example demonstrates how to create a narrowband pulsed measurement using the Pulsed Application DLL. It first gets valid configuration settings and then uses those settings to configure the PNA and external pulsed generators.

To run this program, you need:

- **Pulsed Application** (Option H08)
- External Pulse Generators
- External Pulse Modulator / Pulse Bias

**See Also**

- Learn how to install and register the pulsed dll on your PC
- See the [ConfigureNarrowBand3](#) Method for sending and returning parameters to the .dll.
- See the documentation for the following [COM IF Configuration](#) commands.
- See the [SCPI IF Configuration](#) commands.
- Learn about the [Pulsed Application](#).

```vba
' Interfaces
Dim OApp As AgilentPNA835x.Application
Dim OIF As AgilentPNA835x.IFConfiguration

' Pulsed parameters
Dim DPRF As Double
Dim DOffset As Double
Dim DSampleRate As Double
Dim LNumTaps As Long
Dim LBW As Long
Dim IPrecision as Integer
Dim BPG81110 As Boolean
Dim BFixedPRF As Boolean

' Pulsed DLL interface
Dim OPulsed As New AgilentPNAPulsed.Application
DPRF=5123 'Hz
LBW=100 'Hz
BPG81110=True 'Using the Agilent 81110A Pulse Generator
BFixedPRF=True 'Do not change the PRF during filter alignment. Only adjust the IFBW.

'Calculate precision of pulse generators so that the config function returns the correct precision with the right filter. For example, DPRF=5000 Hz with a pulse
```
generator that will only take a total of four numeric values
' (5.123 kHz)
'–log10(DPRF)=3.709
'–int(3.709)=3
'–3–3=0
'The algorithm will use a 10^x value for decrementing the PRF for null computation.
   This means that the first numeric digit from the right should be the one that is
decremented by the pulsed algorithm (i.e. 5.122 kHz) to compute the filter nulls.
This ensures that the pulse generators receive a PRF that is within their precision
with the associated nulling IFBW.
IPrecision = Int(Flog10(CSng(DPRF))) - 3
'Send desired pulsed parameters to the pulsed configuration DLL. The DLL will return
a new set of pulse parameters that provide the proper filter nulling.
OPulsed.ConfigNarrowBand3 DPRF, LNumTaps, LBW, DOffset, DSampleRate, IPrecision,
BFixedPRF, BPG81110
'Send configuration to PNA
'Set receivers to medium gain setting
OIF.IFGainMode("ALL") = naMANUAL
OIF.IFGainLevel("A") = 1
OIF.IFGainLevel("B") = 1
OIF.IFGainLevel("R1") = 1
OIF.IFGainLevel("R2") = 1
Public Function Flog10(SGNum As Single) As Single
Flog10 = Log(SGNum) / Log(10)
End Function
'Enter Code here to send configuration to external pulse generators
Create a Balanced Measurement using COM

The following program creates several Balanced measurements in separate windows, generates markers, calculates statistics, and sets limit lines and queries results.

**Note:** By their nature, balanced measurements are extremely sensitive to phase differences between the two RF paths that make up the balanced port, especially at higher frequencies. A good calibration (not performed in this example) is critical to achieving good balanced measurement results.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as BalancedCOM.vbs. Learn how to setup and run the macro.

```vb
' PNA application object
Dim app

' Channel 1 object
Dim chan1

' start of marker/limit testing range
Dim minTestStimulus

' end of marker/limit testing range
Dim maxTestStimulus

' Set to true if you want additional balanced measurements.
Dim AdditionalMeasurements
AdditionalMeasurements = 1

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")

' Preset the instrument
app.Preset

' Get the Channel 1 object
Set chan1 = app.Channels(1)

' Stop data taking for now.
chan1.Hold true

' Set up the start / stop frequency for Channel 1 sweep.
MHZ = 1000000
GHZ = 1000*MHZ
chan1.StartFrequency = 10  *MHZ
chan1.StopFrequency = 1  *GHZ
chan1.NumberOfPoints = 801

' Define our test frequency range
minTestStimulus = 100*MHZ
maxTestStimulus = 900*MHZ

' This example uses DUT topology Bal-Bal -
' a DUT with a balanced input and balanced output.

' Port mapping for our DUT:
' logical port 1 = physical ports 1 and 4
```
' logical port 2 = physical ports 2 and 3
' The default is:
' logical port 1 = physical ports 1 and 2
' logical port 2 = physical ports 3 and 4

' logical 1 logical 2

1 ------|     ------ 2 +
| DUT
|-------|--------|----- 3 -

chan1.BalancedTopology.SetBBPorts 1, 4, 2, 3

' Now we create some Bal-Bal measurements.
' By creating Bal-Bal measurements ("BBAL:..."),
' the channel is set to Bal-Bal topology,
' so it is not necessary to do this explicitly
' with the BalancedTopology.DUTTopology command.
' We do it here just for clarity:

chan1.BalancedTopology.DUTTopology = 2
0 == SE-Bal, 1 == SE-SE-Bal, 2 == Bal-Bal

' Create four windows, each showing one category of balanced measurement:
' Create Forward Transmission Measurements in Bal-Bal topology on Channel 1, window 1

differential mode transmission
app.CreateMeasurement 1, "BBAL:SDD21",1,1
Set sdd21_1 = app.ActiveMeasurement

differential to common mode conversion
app.CreateMeasurement 1, "BBAL:SCD21",1,1
Set scd21_1 = app.ActiveMeasurement

common to differential mode conversion
app.CreateMeasurement 1, "BBAL:SDC21",1,1
Set sdc21_1 = app.ActiveMeasurement

common mode transmission
app.CreateMeasurement 1, "BBAL:SCC21",1,1
Set scc21_1 = app.ActiveMeasurement

' Optionally create some additional measurements
If AdditionalMeasurements Then

Create (logical) Port 1 reflection measurements, channel 1, window 2
app.CreateMeasurement 1, "BBAL:SDD11",1,2 ' differential mode reflection
app.CreateMeasurement 1, "BBAL:SDC11",1,2 ' C to D mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCD11",1,2 ' D to C mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCC11",1,2 ' common mode reflection
app.CreateMeasurement 1, "BBAL:SDD12",1,3 ' differential mode transmission
app.CreateMeasurement 1, "BBAL:SCD12",1,3 ' differential to common mode conversion
app.CreateMeasurement 1, "BBAL:SDC12",1,3 ' common to differential mode conversion
app.CreateMeasurement 1, "BBAL:SCC12",1,3 ' common mode transmission

app.CreateMeasurement 1, "BBAL:SDD22",1,4 ' differential mode reflection
app.CreateMeasurement 1, "BBAL:SDC22",1,4 ' C to D mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCD22",1,4 ' D to C mode conversion reflection
app.CreateMeasurement 1, "BBAL:SCC22",1,4 ' common mode reflection

End If

sdd21_1.LimitTest(1).BeginStimulus = minTestStimulus
sdd21_1.LimitTest(1).EndStimulus = maxTestStimulus
sdd21_1.LimitTest(1).BeginResponse = -2
sdd21_1.LimitTest(1).EndResponse = -2
sdd21_1.LimitTest(1).Type = 2 ' minimum limit
sdd21_1.LimitTest.State = 1

sdc21_1.LimitTest(1).BeginStimulus = minTestStimulus
sdc21_1.LimitTest(1).EndStimulus = maxTestStimulus
sdc21_1.LimitTest(1).BeginResponse = -20
sdc21_1.LimitTest(1).EndResponse = -20
sdc21_1.LimitTest(1).Type = 1 ' maximum limit
sdc21_1.LimitTest.State = 1

chan1.Single true

sdd21_1.ShowStatistics = true

chan1.UserRangeMin(0,1) = minTestStimulus
chan1.UserRangeMax(0,1) = maxTestStimulus

sdc21_1.MarkerState(1) = true

sdc21_1.Marker(1).UserRange = 1
sdc21_1.Marker(1).SearchMax

sdd21_1.MarkerState(1) = true

sdd21_1.Marker(1).UserRange = 1
sdd21_1.Marker(1).SearchMin
If sdd21_1.LimitTestFailed Then
End If
If sdc21_1.LimitTestFailed Then
Wscript.Echo "Common to differential conversion failed: " &
sdc21_1.Marker(1).Stimulus/MHZ & "MHz, " & _sdc21_1.Marker(1).Value(1) & " dB"
End If
ECAL Confidence Check

This Visual Basic program:

- Initializes the PNA objects.
- Performs a complete ECAL confidence check

Before using this code:

- The active channel must contain an S11 measurement with a 1-port or N-port calibration
- Prepare a form with two buttons named `cmdRun` and `cmdQuit`

```vbnet
Private oPNA As AgilentPNA835x.Application
Private oChan As Channel
Private oCal As Calibrator
Private oMeas As Measurement

Private Sub cmdRun_Click()
    Dim iMeasIndex As Integer
    Set oPNA = CreateObject("AgilentPNA835x.Application", "MachineName")
    Set oChan = oPNA.ActiveChannel
    Set oCal = oChan.Calibrator
    iMeasIndex = 1

    ' Loop through measurements until an S11 on the active channel
    ' is found, or the end of the measurement collection is reached.
    Do
        Set oMeas = oPNA.Measurements(iMeasIndex)
        If oMeas.Parameter = "S11" And _
            oMeas.channelNumber = oChan.channelNumber Then Exit Do
        iMeasIndex = iMeasIndex + 1
        If iMeasIndex > oPNA.Measurements.Count Then
            MsgBox "No S11 measurement found on the active channel." _
            & "Create an S11 measurement, then try again."
            Exit Sub
    End If
End Sub
```
Loop

' Set up trace view so we are viewing only the data trace.
oMeas.View = naData

' Acquire the S11 confidence check data from ECal Module A
' into the memory buffer.
oCal.AcquireCalConfidenceCheckECALEx "S11", 1

' Turn on trace math so the trace shows data divided by memory.
' You can be confident the S11 calibration is reasonably good if
' the displayed trace varies no more than a few tenths of a dB
' from 0 dB across the entire span.
oMeas.TraceMath = naDataDivMemory

End Sub

Sub cmdQuit_Click()

' Turn off trace math
' in case someone clicks Quit without having clicked Run
If oMeas <> Nothing Then oMeas.TraceMath = naDataNormal

' Conclude the confidence check to set the ECal module
' back to it's idle state.
If oCal <> Nothing Then oCal.DoneCalConfidenceCheckECAL

' End the program
End

End Sub
Limit Line Testing with COM

This Visual Basic program:

- Turns off existing Limit Lines
- Establishes Limit Lines with the following settings:
  - Frequency range - 4 GHz to 8 GHz
  - Maximum value - (10dB)
  - Minimum value - (-30dB)
- Turns on Lines, Testing, and Sound

If using **Global Pass/Fail** to report limit results, trigger the PNA after configuring and enabling Limit lines.

```vbnet
Public limts As LimitTest
Set limts = meas.LimitTest
' All Off
For i = 1 To 20
  limts(i).Type = naLimitSegmentType_OFF
Next i

' Set up Limit Lines
limts(1).Type = naLimitSegmentType_Maximum
limts(1).BeginResponse = 10
limts(1).EndResponse = 10
limts(1).BeginStimulus = 4000000000#
limts(1).EndStimulus = 8000000000#
limts(2).Type = naLimitSegmentType_Minimum
limts(2).BeginResponse = -30
limts(2).EndResponse = -30
limts(2).BeginStimulus = 4000000000#
limts(2).EndStimulus = 8000000000#

'Turn on Lines, Testing, and Sound
limts.LineDisplay = 1
limts.State = 1
limts.SoundOnFail = 1
```
E5091 Testset Control

The following VB Script example exercises the COM commands used to control the E5091A testset.
For a description of each command, see E5091 Testsets collection.

```vbnet
Sub Main()
    Set pna = CreateObject("AgilentPNA835x.Application")
    Dim testsets As E5091Testsets
    Set testsets = pna.E5091Testsets
    Dim tset1 As E5091Testset
    Set tset1 = testsets(1)
    tset1.OutputPort(1, 3) = naE5091PortR2
    tset1.ControlLines(1) = 5
    tset1.ShowProperties = True
    tset1.Enabled = True
    MsgBox tset1.ID
    MsgBox tset1.Enabled
    MsgBox tset1.ShowProperties
    ' NumberOfPorts property returns 0 when testset not connected
    MsgBox tset1.NumberOfPorts
    MsgBox tset1.OutputPort(1, 3)
    MsgBox tset1.ControlLines(1)

    Dim tset2 As E5091Testset
    Set tset2 = testsets(2)
    tset2.Enabled = True
    tset2.ShowProperties = True
    MsgBox tset2.Enabled
    MsgBox tset2.ShowProperties
End Sub
```
Errors and the SCPIStringParser Object

This C++ program uses the SCPIStringParser.Parse command to detect the failed HRESULT and interrogate the errorInfo object for more details.

// scpierrors.cpp : Defines the entry point for the console application.
//

#include <iostream>
#include "afx.h"
#include "atlbase.h"
#import "C:\program files\common files\agilent\pna\835x.tlb" raw_interfaces_only, no_namespace, named_guids
using namespace std;
HRESULT SendScpiCommand( IScpiStringParser* parser, CComBSTR& cmd, CComBSTR& response)
{
    CComBSTR bstr;
    HRESULT hr = parser->Parse(CComBSTR(cmd), &response);
    if (FAILED(hr))
    {
        // see if this interface supports ErrInfo
        CComPtr<ISupportErrorInfo> spSupportsErrInfo;
        if (SUCCEEDED(parser->QueryInterface(&spSupportsErrInfo)))
        {
            // it does, so let's get the errorinfo object
            CComPtr<IErrorInfo> spErrorInfo;
            if (SUCCEEDED(GetErrorInfo(0, &spErrorInfo)))
            {
                CComBSTR errStr;
                spErrorInfo->GetDescription(&errStr);
                std::cout << "ERROR: " << CString(errStr) << std::endl;
            }
        }
    }
    return hr;
}
int main()
{
    CoInitialize(NULL);
    {
        CComBSTR response;
        CComPtr<IApplication> spPNA;
        CComPtr<IScpiStringParser> spSCPI;
        if (SUCCEEDED(spPNA.CoCreateInstance(CLSID_Application)))
        {
            spPNA->get_ScpiStringParser(&spSCPI);
            SendScpiCommand(spSCPI, CComBSTR("SYSTEM:PRESET"), response);
            SendScpiCommand(spSCPI, CComBSTR("CALC:PAR:CAT?"), response);
            std::cout << CString(response) << std::endl;
            SendScpiCommand(spSCPI, CComBSTR("THIS:IS:A:SYNTAX:ERROR"), response);
        }
    }
    CoUninitialize();
    return 0;
}
External Testset Control

The following VB Script example exercises the COM commands used to control the Z5623AK64 testset. For a description of each command, see TestsetControl Object.

See Other COM Example Programs

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save on the PNA hard drive as Testset.vbs. Learn how to setup and run the macro.

```vbs
' Demonstrate some COM commands for external testsets.
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Sub DemoTestset(na)
Dim testsets, tset1
Dim portNum
Dim chNum, address
Set testsets = na.ExternalTestsets
chNum = 1

' Load a configuration file.
' NOTE: the K64 testset is only compatible with 4-port analyzers.
address = 0
testsets.Add "Z5623AK64", address

' Get the testset object
' in the testsets collection.
Set tset1 = testsets(1)

' Show the selections available for each port.
For portNum = 1 To 4
MsgBox("Port " & CStr(portNum) & " catalog: " & tset1.PortCatalog(portNum))
Next

' Set port mappings on channel 1.
tset1.OutputPorts(chNum) = "5 ext R,2 int R,3 int R,6 int R"
```
' Set control lines.
tset1.ControlLines(chNum) = 85

' Set label.
tset1.Label(chNum) = "Some label"

' Enable external testset control. This automatically enables status bar display as well.
tset1.Enabled = True
End Sub

' The testset used in this demo is only usable on 4-port analyzers
If (pna.NumberOfPorts <> 4) Then
    MsgBox("This program only runs on 4-port analyzers.")
Else
    DemoTestset(pna)
End If
PathConfiguration Example

Note: These commands are accessible only for PNA-X models.
This Visual Basic Program exercises various commands on the:

- PathConfigurationManager Object
- PathConfiguration Object
- PathElement Object

```vbnet
' Create / Get the PNA application
Dim app
Set app = CreateObject("AgilentPNA835x.Application")
' Preset the instrument
app.Preset
' Get a channel interface on which to operate
Dim chan
Set chan = App.ActiveChannel
' Modify the Default configuration, and save it as "My Config"
chan.PathConfiguration = "Default"
' Set the "Combiner" element to value "Reversed"
chan.PathConfiguration.Element("Combiner").Value = "Reversed"
' Set the "Src1" element to value "High Power"
chan.PathConfiguration.Element("Src1").Value = "High Power"
' Change the description text
chan.PathConfiguration.DescriptionText = "Connect J8 to J9."
' Store the modified configuration
chan.PathConfiguration.Store("My Config")
' Set the instrument’s path config back to the default (req. 8)
chan.PathConfiguration = "Default"
' Load a previously saved configuration onto channel 2
App.PathConfigurationManager.Load 2, "My Config"
```
COM Events Example

This Visual Basic program shows how to monitor the end of sweep. The program will set sweep time to various amounts and BEEPs when sweep is completed. This method allows other processes to continue while waiting for end-of-sweep. This program stops after 10 loops.

Note: To avoid Permission Denied problems, this should be run on the PNA and not a PC. To run it from a PC both units must be "trusted" and on the same domain/workgroup.

Option Explicit
Dim na As AgilentPNA835x.Application
Dim WithEvents naEvnt As AgilentPNA835x.Application
Dim ch As AgilentPNA835x.Channel
Dim sweepComplete As Boolean

Private Sub Form_Load()

Dim N As Integer
Set na = CreateObject("AgilentPNA835x.application")
na.preset
Set ch = na.ActiveChannel
na.DisallowAllEvents ' Turn off all events
Set naEvnt = na ' Enable event interrupts
Do
N = N + 1 ' Loop counter
ch.sweepTime = 1 + (Rnd * 9) ' Set random sweep-time from 1-10 sec
sweepComplete = False ' Trigger sweep
ch.Single False
naEvnt.AllowEventCategory naEventCategory_CHANNEL, True ' Enable Channel event
Do
DoEvents ' Allows other processes to continue
Loop Until sweepComplete = True
naEvnt.AllowEventCategory naEventCategory_CHANNEL, False ' Disable event until ready for next one
Beep ' Do end-of-sweep processing here;
Loop Until N > 10
End
End Sub

Private Sub naEvnt_OnChannelEvent(ByVal eventID As Variant, ByVal chNumber As Variant)
' In this example we don't care about the channel info
If eventID = naEventID_CHANNEL_TRIGGER_COMPLETE Then sweepComplete = True
End Sub
Configure for COM-DCOM Programming

Before developing or running a COM program, you should first establish communication between your PC and the analyzer. This process is referred to as gaining Access to the analyzer. You should then register the PNA type library on your PC.

**DCOM** (Distributed Component Object Model) refers to accessing the PNA from a remote PC.

**COM** refers to accessing the PNA application from the analyzer PC.

- **Access Concepts**
- **Access Procedures**
- **Register the PNA Type Library on Your PC**
- **Problems?**

**Note:** After performing a Firmware Upgrade you must copy the new type library to your development PC to get access to new COM commands. See Register the analyzer on your PC.

**Other Topics about COM Concepts**

For detailed information on this subject, see [http://na.tm.agilent.com/pna/DCOMSecurity.html](http://na.tm.agilent.com/pna/DCOMSecurity.html)

**Access Concepts**

PNAs are shipped from the factory such that Everyone has permission to launch and access the PNA application via COM/DCOM. The term Everyone refers to a different range of users depending on whether the PNA is a member of a Domain or Workgroup (it must be one or the other; not both). By default, the PNA is configured as members of a workgroup. Therefore, Everyone includes only those users who have been given logon accounts on the PNA.

**Workgroup**

A workgroup is established by the PNA administrator declaring the workgroup name and declaring the PNA as a member of the workgroup. A workgroup does not require a network administrator to create it or control membership.

Everyone includes only those users who have been given logon accounts on the PNA.

By default, the PNA is configured as members of a workgroup named WORKGROUP.

**Note:** To setup a logon account for a new user, see Additional Users. The easiest method of gaining DCOM access, is to make the user's account name and password on the PNA to EXACTLY match their PC logon account name and password.

**Domain**

A domain is typically a large organizational group of computers. Network administrators maintain the domain and control which machines have membership in it.
Everyone includes those people who have membership in the domain. In addition, those with logon accounts can also access the analyzer.

**Summary**

- A Workgroup requires no maintenance, but allows DCOM access to only those users with a log-on account for the PNA.
- A Domain requires an administrator, but all members of the domain and those with logons to the analyzer are allowed DCOM access to the PNA.

The following section "Access Procedures" provides a tighter level of security allowing only selected (not Everyone) domain and workgroup users DCOM Access and Launch capability of the PNA.

**Access Procedures**

Perform this procedure for the following reasons:

- To allow only selected users (not everyone) remote Access and remote Launch capability to the PNA. Launch capability is starting the PNA application if it is not already open.
- To verify that you have DCOM access to the analyzer.

**Note:** Before doing this procedure, you must first have a logon account on the PNA. See [PNA User Accounts](#).

The following procedure grants specific users DCOM access and launch capability of the PNA application:

To perform this procedure, you must first [minimize the PNA](#) application.

**How do I know which Operating System I have?**

<table>
<thead>
<tr>
<th>Windows 2000</th>
<th>Windows XP</th>
</tr>
</thead>
<tbody>
<tr>
<td>On the PNA, click the Windows <strong>Start</strong> button</td>
<td>On the PNA, click the Windows <strong>Start</strong> button</td>
</tr>
<tr>
<td>Click <strong>Run</strong></td>
<td>Click <strong>Run</strong></td>
</tr>
<tr>
<td>In the <strong>Open:</strong> box, type <strong>dcomcnfg</strong></td>
<td>In the <strong>Open:</strong> box, type <strong>dcomcnfg</strong></td>
</tr>
<tr>
<td>Click <strong>OK</strong></td>
<td>Click <strong>OK</strong></td>
</tr>
</tbody>
</table>
| In the Distributed COM Configuration Properties window, Click on **Agilent PNA Series** in the Applications list. Then click **Properties**... | Open the following folder sequence:  
  - Component Services Window  
  - Component Services  
  - Computers  
  - My Computer  
  - DCOM Config  
  - Right click **Agilent PNA Series** |
<table>
<thead>
<tr>
<th><strong>Click the Security tab</strong></th>
<th>Click Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Click Use custom access permissions then click Edit</strong></td>
<td>Under Access Permissions, click Customize, then click Edit</td>
</tr>
<tr>
<td><strong>In Registry Value Permissions, select Everyone, then click Remove</strong></td>
<td>Select Everyone, then click Remove</td>
</tr>
<tr>
<td><strong>Click Add</strong></td>
<td>Click Add</td>
</tr>
<tr>
<td><strong>You could either select one or more of these groups to have access to the PNA, or specific users. To give specific users access, click Show users or Members, then select the name from the list.</strong></td>
<td>Type a group name or user account name</td>
</tr>
<tr>
<td><strong>Click Add, then click OK</strong></td>
<td>Click OK</td>
</tr>
</tbody>
</table>

**Launch Permission**

<table>
<thead>
<tr>
<th><strong>Click Use custom launch permissions, then click Edit</strong></th>
<th>Under Launch Permissions, click Customize, then click Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>In Registry Value Permissions, select Everyone then click Remove</strong></td>
<td>Select Everyone, then click Remove</td>
</tr>
<tr>
<td><strong>Click Add</strong></td>
<td>Click Add</td>
</tr>
<tr>
<td><strong>You could either select one or more of these groups to have launch permission of the PNA, or specific users.</strong></td>
<td>Type a group name or user account name</td>
</tr>
<tr>
<td></td>
<td>- To give groups launch permission, select the group from the list.</td>
</tr>
<tr>
<td></td>
<td>- To give specific users launch permission, click Show users or Members, then select the name from the list.</td>
</tr>
<tr>
<td><strong>Click Add, then click OK</strong></td>
<td>Click OK</td>
</tr>
</tbody>
</table>

**Register the PNA Type Library on Your PC**

The type library contains the PNA object model. On your PC, there is a Registry file that keeps track of where object models are located. Therefore, you must register the type library on the PC that will be used to develop code and run the program. It is much more efficient to have the type library registered at design time (BEFORE running your COM program).

Do the following two items before proceeding:
1. Connect your PC and the PNA to LAN.

2. Either map a drive to the analyzer or copy the type library files on a floppy disk or other media. See Drive Mapping.

**Note:** To register the type library on your PC, you must be logged on as an administrator of your PC. Learn about User Accounts.

This procedure will do the following:

- Register the Network Analyzer application on your PC.
- Copy and register the proxystub (835xps.DLL) onto the PC.
- Copy and register the PNA type library (835x.tlb) onto the PC.
- Copy and register the FCA type library (fca.tlb) onto the PC.

1. Using Windows Explorer on your PC, find the Analyzer’s C: drive. The drive will not be named "C:" on your PC, but a letter you assigned when mapping the drive.

2. Navigate to **Program Files \ Agilent \ Network Analyzer \ Automation**

3. Double-click **pnaproxy.exe** and follow the prompts to Install PNA Proxy. If the installation offers a choice of Modify, Repair, or Remove, then select **Remove**. Then double-click on **pnaproxy.exe** again.

4. When prompted, type the Computer name of the PNA (Learn how to find this).

5. After the install program runs, the PNA and FCA type library should be registered on your PC.

6. Your programming environment may require you to set a reference to the PNA type library now located on your PC. In Visual Basic, click **Project, References**. Then browse to **C:\Program Files\Common Files\Agilent\PNA** Select **835x.tlb**

**Problems?**

- These procedures will fail if there are any programs using the PNA type library  (for example: Visual basic, VEE, Visual Studio, or any other application program that may communicate with the PNA).

- Perform the following procedure if the previous procedure did not return an error, but you cannot connect to the PNA.

- If you received an error, check that both the account name and password used on both the PNA and PC match EXACTLY.


1. **Map a drive** from your remote PC to the PNA. Note the drive letter your PC assigns to the PNA. Substitute this drive letter for **PNA** in the following procedure.

2. On your PC, go to a DOS prompt c:>
3. Type **PNA:** (for example o:)

4. Type `cd program files\agilent\network analyzer\automation`

5. Type `copy 835xps.dll c:\program files\common files\agilent\pna`

6. Type `copy 835x.tlb c:\program files\common files\agilent\pna`

   If you will NOT be using **FCA commands**, skip steps 7,.8, and 9.

7. Type `cd..`

8. Type `cd extensions\fca`

9. Type `copy fca.tlb c:\program files\common files\agilent\pna`

10. If it is not already there, copy `regtlib.exe` from **PNA**:\WINNT to your C:\<windows>\system32 directory  
    (<windows> is OS-dependent- it is either windows or WINNT)

11. Type `regtlib C:\program files\common files\agilent\pna\835x.tlb`

12. Type `regsvr32 C:\program files\common files\agilent\pna\835xps.dll`

13. Type `regtlib C:\program files\common files\agilent\pna\fca.tlb`

   Perform the [Access Procedure](#) after doing these steps.
COM Fundamentals

The following terms are discussed in this topic:

- Objects
- Interfaces
- Collections
- Methods
- Properties
- Events
- Visual Basic Syntax

**Note:** The information contained in this topic is intended to help an experienced SCPI programmer transition to COM programming. This is NOT a comprehensive tutorial on COM programming.

Other Topics about COM Concepts

**Visual Basic Syntax**

The examples in PNA Help use Visual Basic as the programming environment for COM, which uses 'dot' notation.

To set a property, follow the object reference with:

- a period (.)
- property or method
- an equal sign (=)
- the new value

For example:

```
object.property = value
```

To read a property, a variable to contain the returned value is followed with:

- an equal sign (=)
- an object, or reference to an object
- a period (.)
- property

For example:
variable = object.property

To execute a method, an object, or reference to an object is followed with:

- a period (.)
- the method
- a blank space
- any required parameters

For example:

object.method parameters

Some methods return values, such as methods that return data. To return data from a method, a variable to contain the returned data is followed with:

- an equal sign (=)
- an object, or reference to an object
- a period (.)
- the method
- any required parameters enclosed in parenthesis

variable = object.method (parameters)

**Objects**

The objects of the Network Analyzer (Application) are arranged in a hierarchical order. The PNA object model lists the objects and their relationship to one another.

In SCPI programming, you must first select a measurement before making settings. With COM, you first get a handle to the object (or collection) and refer to that object in order to change or read settings (properties).

For more information on working with objects, see Getting a Handle to an Object.

**Interfaces**

A COM Interface is the connection to an object. When you get a handle to an object, you are actually using an interface to an object. This is important if you are developing PNA code that will run on multiple code versions. For more information, see PNA Interfaces.

**Collections**

A collection is an object that contains several other objects of the same type. For example, the Channels collection contains all of the channel objects.

**Note:** In the following examples, the collections are referred to as a variable. Before using a collection object, you must first get an instance of that object. For more information, see Getting a Handle to an Object

Generally, items in a collection can be identified by number or by name. The order for objects in a collection cannot be assumed. They are always unordered and begin with 1. For example, in the following procedure, chans(1) is used to set averaging on the first channel in the Channels collection (not necessarily channel 1).
Sub SetAveraging()
    chans(1).AveragingFactor = 10
End Sub

The following procedure uses the measurement string name to set the display format for a measurement in the measurements collection.

meass("CH1_S11_1").Format = 1

You can also manipulate an entire collection of objects if the objects share common methods. For example, the following procedure sets the dwell time on all of the segments in the collection.

Sub setDwell()
For Each seg In segs
    segs.DwellTime = 0.03
Next
End Sub

**Methods**

A method is an action that is performed on an object. For example, `CreateSParameter` is a method on the `Application` object. The following procedure uses that method to create a new S21 measurement in channel 1 in a new window.

Sub CreatMeas
    app.CreateSParameter 1,2,1,1
End Sub

**Properties**

A property is an attribute of an object that defines one of the object's characteristics, such as size, color, or screen location. A property can also change an aspect of the object's behavior, such as whether the object is visible. In either case, to change the characteristics of an object, you change the values of its properties.

For example, the following statement sets the IF Bandwidth of a channel to 1 KHz.

Chan.IFBandwidth = 1e3

You can also read the current value of a property. The following statement reads the current IF Bandwidth of a channel into the variable `Ifbw`.

`Ifbw = Chan.IFBandwidth`

Some properties cannot be set and some cannot be read. The Help topic for each property indicates if you can:

- Set and read the property (Write/Read)
- Only read the property (Read-only)
- Only set the property (Write-only)

**Events**

A COM event is an action recognized by an object, such as clicking the mouse or pressing a key. Using events, your program can respond to a user action, program code, or triggered by the analyzer.

The SCPI equivalent of an event is a Service Request (SRQ).

For example:

`OnChannelEvent`
For more information, see Working with the Analyzer's Events.
**Getting a Handle to an Object**

The following are discussed in this topic:

- **What Is a Handle**
- **Declaring an Object Variable**
- **Assigning an Object Variable**
- **Navigating the Object Hierarchy**
- **Getting a Handle to a Collection**

---

**Other Topics about COM Concepts**

**What Is a Handle**

In SCPI programming, you must first select a measurement before changing or reading settings. With COM, you first get a handle to the object (or collection) and refer to that object to change or read its settings. The following analogy illustrates this:

A CAR could be called an object. More precisely, CAR is a class of objects. For example, one of the properties of the CAR class is "**Color**". You can read (by looking) or set (by painting) the color property of a car object. In other words, you can only read or set properties of a specific car object; not the entire car class. Therefore, to read or set a property, you need to get "a handle", or an instance of the object.

This process is also called "accessing an object", "getting an instance of an object", "returning an object", or "referring to an object". You can have handles to many instances of an object at the same time.

**Accessing PNA Objects**

The PNA Application object is the highest object in the PNA object model hierarchy. Because of that, it is the only object that must be 'created' before it, or any other objects, can be accessed and used. During the creation process, the application object assigned to a variable name, or handle. Throughout your program, that object is used by referring to that variable. All PNA objects can be assigned to a variable, and subsequently referred to, in this same manner.

The following example shows how to create the PNA Application object, as well as illustrate the general steps of get a handle to an object.

There are two steps in the process of getting a handle to analyzer objects:

1. Declaring a Variable
2. Assigning an Object to the Variable

**1. Declaring a Variable**

**Note:** The examples in these topics use the Visual Basic Programming Language. See the short section regarding Visual Basic syntax.

Use the Dim statement or one of the other declaration statements (Public, Private, or Static) to declare a variable.
The type of variable that refers to an object must be a Variant, an Object, or a specific type of object. Some programming languages, such as VBScript and Agilent VEE, do not allow you to specify variable types. The following examples ALL declare the variable `pna`. Each subsequent statement is more specific than the previous:

- `Dim pna 'Variant data type.
- `Dim pna As Object 'Object data type.
- `Dim pna As AgilentPNA835x.Application ' Specific Application type
- `Dim pna As AgilentPNA835x.IApplication ' Interface type

1. If you use a variable without declaring it first, the data type of the variable is Variant. If you don't care about using automatic type checking, and willing to run code less efficiently, this method is very safe and is useable on all programming environments.

2. If you know the specific object type, and your programming environment allows it, you can declare the variable as an object.

3. Declaring a specific object type provides automatic type checking (Intellisense), faster code, and improved readability.

4. Declaring the interface is the most specific way and is beneficial when developing code for multiple firmware revisions. Learn more about Interfaces.

**2. Assigning an Object to a Variable**

To assign an object instance to a variable, use the `Set` keyword before the object variable that was declared previously. In the following line of code, we SET the current AgilentPNA835x Application to "pna".

```vba
Set pna = AgilentPNA835x.Application
```

As mentioned earlier, the AgilentPNA835x object is unique because it is the highest level of object in the PNA object model hierarchy. Therefore, we must use the `CreateObject` keyword with the (`classname`, server name) parameters.

- The `classname` for the analyzer object is always "AgilentPNA835x.Application".
- To find your analyzer's `server name`, see View or change full computer name

The following statements create an instance of the Analyzer object.

```vba
Dim pna AS AgilentPNA835x.Application
Set pna = CreateObject("AgilentPNA835x.Application", "Analyzer46")
```

**Note:** These statements will start the PNA application if it is not already running on your instrument.

**Navigating the Object Hierarchy**

Once an instance of the PNA Application is "created", you access all of the PNA objects by navigating the object hierarchy. Navigating the object model hierarchy can be tricky. In addition, you also need to know how to refer to a specific instance of that object. For example, if you have three measurements present on the PNA, how do you refer to the channel 1 measurement? Each object on the PNA Object Model image is linked to an object page. At
the top of each object page is a **Description** section and another called "**Accessing the ... Object**". These sections together explain how to navigate the PNA hierarchy to access a specific instance of that object.

From the previous discussion, you may think that you must always declare and assign variables to an object before setting or reading its properties. While this method is best for objects that you will continue to reuse, such as a measurement, it is not always necessary. You can also refer to an object directly.

The **TriggerSetup** object, which is a child of the Application object. Because we will only need to refer to this object once to set a couple of properties, and it is easy to access, we will refer to it directly. From the previous example, we already have a handle to the Application object in the variable `pna`. The following example uses **Visual basic 'dot' notation** to refer to the TriggerSetup object, and then the Scope property.

```vbnet
pna.TriggerSetup.Scope = naChannelTrigger
```

By referring to the TriggerSetup object directly, we must type the same path whenever we refer to properties on the TriggerSetup object. The following method assigns the `pna.TriggerSetup` object to a variable that can be reused.

```vbnet
Dim trig As Object
Set trig = pna.TriggerSetup
```

Once created, you can treat an object variable exactly the same as the object to which it refers. For example:

```vbnet
trig.Scope = naChannelTrigger
trig.Source = naTriggerSourceInternal
```

**Getting a Handle to a Collection**

The analyzer has several collections of objects which provide a convenient way of setting or reading all of the objects in the collection with a single procedure. Also, there are objects (limit lines for example) that can only be accessed through the collection.

To get a handle to an item in a collection, you can refer to the object by item number or sometimes by name. However, you first have to get a handle to the collection. To assign the collection to a variable, use the same two step process (1. declare the variable, 2. assign the variable using 'Set').

```vbnet
Dim meass As Measurements
Dim meas As Measurement
```

You can then iterate through the entire collection of measurements to read or set properties

```vbnet
Sub setFormat()
For Each meas In meass
meas.Format = naDataFormat_LinMag
Next
End Sub
```

Or you can read or set a property on an individual object in the collection:

```vbnet
meass(1).Format = naLinMag
```

**Note:** Each object and collection has its own unique way of dealing with item names, and numbers. Refer to the **Analyzer Object Model** for details.
Collections in the Analyzer

Collections are a gathering of similar objects. They are a convenience item used primarily to iterate through the like objects in order to change their settings. Collections generally provide the following generic methods and properties:

- `Item(n)`
- `Count`
- `Add(n)`
- `Remove(n)`

where `(n)` represents the number of the item in the collection. Some collections may have unique capabilities pertinent to the objects they collect.

Other Topics about COM Concepts

Collections are Dynamic

A collection does not exist until you ask for it. When you request a Channels object (see Getting a Handle to an Object / Collection), handles to each of the channel objects are gathered and placed in an array.

For example, if channels 2 and 4 are the only channels that exist, then the array will contain only 2 items. The command 'channels.Count' will return the number 2, and:

- Channels(1) will contain the channel 2 object.
- Channels(2) will contain the channel 4 object.

The ordering of objects within the collection should not be assumed. If you add a channel to the previous example, as in:

```
Pna.Channels.Add(3)
```

'channels.Count' will now return 3 and:

- Channels(1) will contain the channel 2 object.
- Channels(2) will contain the channel 3 object.
- Channels(3) will contain the channel 4 object.

Primarily, collections are useful for making this type of iteration possible:

```
Dim ch as Channel
For each ch in pna.Channels
    Print ch.Number
    Print ch.StartFrequency
    Print ch.StopFrequency
Next ch
```

As soon as this for-each block has been executed, the Channels object goes out of scope.
COM Data Types

The PNA uses several data types to communicate with the host computer. Before using a variable, it is best to declare the variable as the type of data it will store. It saves memory and is usually faster to access. The following are the most common data types:

- **Long Integer**
- **Single Precision (Real)**
- **Double Precision (Real)**
- **Boolean**
- **String**
- **Object**
- **Enumeration**
- **Variant**

Other Topics about COM Concepts

**Long** (long integer) variables are stored as signed 32-bit (4-byte) numbers ranging in value from -2,147,483,648 to 2,147,483,647.

**Double** (double-precision floating-point) variables are stored as IEEE 64-bit (8-byte) floating-point numbers ranging in value from -1.79769313486232E308 to -4.94065645841247E-324 for negative values and from 4.94065645841247E-324 to 1.79769313486232E308 for positive values.

**Single** (single-precision floating-point) variables are stored as IEEE 32-bit (4-byte) floating-point numbers, ranging in value from -3.402823E38 to -1.401298E-45 for negative values and from 1.401298E-45 to 3.402823E38 for positive values.

**Boolean** variables are stored as 16-bit (2-byte) numbers, but they can only be True or False. Use the keywords True and False to assign one of the two states to Boolean variables.

When other numeric types are converted to Boolean values, 0 becomes False and all other values become True. When Boolean values are converted to other data types, False becomes 0 and True becomes -1.

In PNA release 5.26, the following properties were changed to return True rather than 1 to conform with this definition. This change may affect the functionality of your COM program:

- **Bandwidth Tracking Property**
- **ErrorCorrection Property**
- **IFGateEnable Property**
**String** variables hold character information. A String variable can contain approximately 65,535 bytes (64K), is either fixed-length or variable-length, and contains one character per byte. Fixed-length strings are declared to be a specific length. Variable-length strings can be any length up to 64K, less a small amount of storage overhead.

**Object** variables are stored as 32-bit (4-byte) addresses that refer to objects within the analyzer or within some other application. A variable declared as Object is one that can subsequently be assigned (using the Set statement) to refer to any actual analyzer object.

**Enumerations (Enum)** are a set of named constant values. They allow the programmer to refer to a constant value by name instead of by number. For example:

```plaintext
Enum DaysOfWeek
    Sunday = 0
    Monday = 1
    Tuesday = 2
    Wednesday = 3
    Thursday = 4
    Friday = 5
    Saturday = 6
End Enum
```

Given this set of enumerations, the programmer can then pass a constant value as follows:

```plaintext
SetTheDay(Monday)
```

rather than

```plaintext
SetTheDay(1)
```

where the reader of the code has no idea what the value 1 refers to.

However, the analyzer RETURNS a long integer, not the text.

```plaintext
Day = DaysofWeek(today) 'Day = 1
```

**Variant** - If you don't declare a data type ("typed" data) the variable is given the Variant data type. The Variant data type is like a chameleon — it can represent many different data types in different situations.

The PNA provides and receives Variant data because there are programming languages that cannot send or receive "typed" data. Variant data transfers at a slower rate than "typed" data.
PNA Interfaces

A COM interface is the connection to an object. When you get a handle to an object, you are actually using an interface to an object. This subtle distinction is relevant to the COM programmer for the following two reasons:

- **Interface Inheritance (Coding for Multiple PNA Versions)**
- **Custom Interfaces.**

**Other Topics about COM Concepts**

**Interface Inheritance (Coding for Multiple PNA Versions)**

The PNA continues to evolve and release new firmware / software versions that provide more functionality and features. New commands are added to existing objects, and with them new interfaces are added to support those commands. For example, new commands were added to the Measurement object in PNA release 3.0. These commands are accessible from the new IMeasurement2 interface. This can be important if you develop code using the type library in release 3.0, and run the code on a PNA with an older release, such as 2.0.

When you use a command that was new with release 3.0, and you run that code on a PNA with release 2.0 firmware, errors will occur because that PNA does not recognize the new commands. However, even if you do NOT utilize new commands, errors can still occur. The following example shows how this occurs and how to avoid it.

The following Visual Basic statement dimensions the `meas` variable as an object.

```vb
Dim meas As Measurement
```

When the program compiles, Visual Basic figures out what interface to use to access that object. When dimensioning as an object, VB will use the default interface. As new interfaces are added to an object, they become the default interface. If this program was developed and compiled using the PNA 3.0 type library, the default Interface of the Measurement Object was IMeasurement2. However, if this program is run on an instrument with PNA 2.0 firmware, there was no IMeasurement2 Interface, and an E_NOINTERFACE error will occur.

Therefore, the more robust approach would be to specify the interface instead of the object when declaring a variable.

```vb
Dim meas As IMeasurement
```

This code will ONLY use the IMeasurement interface; not the default interface.

However, regardless of how you declare a variable, errors will always occur if you use new commands, and run the code on an older instrument.

**Custom Interfaces**

The PNA object model contains three "custom" interfaces use "typed" variables, which is more efficient than using variant type variables. However, these interfaces are only usable from VB6, C, & C++. All other programming languages must use the other standard interfaces.

The custom interfaces are:

- **IArrayTransfer** - Measurement object
- **ICalData** - Calibrator object
- `ISourcePowerCalData` - Channel object
Working with Events

- What are Events?
- Using the Analyzer's Events
- Event ID's
- Filtering Events
- List of Events
- Out of Range Errors
- Troubleshooting Problems with Events

See Also

- Events Example
- Errors and the SCPIStringParser Object

### Other Topics about COM Concepts

#### What are Events?

Windows applications work from user-initiated events such as mouse moves and mouse clicks. A mouse-click produces an event that the programmer can either ignore or "handle" by providing an appropriate subroutine like this:

```vb
Sub DoThis_onClick
    Perform something
End Sub
```

If this subroutine were in your program and the mouse-click event occurs on your PC, it would generate a "Callback" to the client and interrupt whatever it was doing and handle the event.

A more practical example of an event in the analyzer is Limit test. If limit test is on and the measurement fails, the analyzer produces a "Limit-failed" event. If the measurement passed, the analyzer produces a "Limit-succeeded" event.

The Analyzer has a very sophisticated Event structure. Your program can be notified when one or more events occur. However, it may not be necessary.

For example, the analyzer has an event that will notify your program when a sweep is complete. A simpler alternative is to use a synchronous command which waits for the sweep to complete.

```vb
sync = True
app.ManualTrigger sync
chan.StartFrequency = 4.5E6
```

This would NOT work if you want the controller to do other things while waiting, like setup a power meter or sort some data. In this case you would like a "callback" from the analyzer to let your program know that the sweep has completed. For an example of this see Events Example.
Another reason to use events is when you want to be notified of several conditions when they occur, such as errors or source unlock conditions. It would not be practical to routinely poll these conditions while executing your program.

**Using Events**

If you decide to use the COM events to get a callback, your program must do two things:

1. **Subscribe to events:**

   All events in the analyzer are a child of the Application object through the INetworkAnalyzerEvents Interface. You must tell the Application object that you are interested in receiving event callbacks. This process is called subscription.

   In Visual Basic, this is done by including " WithEvents" in the declaration statement. The declaration below dimensions an Application object (myPNA) and subscribes to the events produced by the Application.

   ```vba
   Dim WithEvents myPNA as AgilentPNA835x.Application
   ```

   In C++, this is a bit more involved. You must queryInterface for the IconnectionPointContainer interface, locate the INetworkAnalyzerEvents interface via a call to FindConnectionPoint and call Advise().

2. **Implement the Event Handler**

   When an event occurs, the Application object will "callback" to the client through the INetworkAnalyzerEvents interface.

   In VB, click on the object window (upper left pane). Find the Application object and click it. The event interfaces will appear in the upper right pane. As you click on them, VB supplies the first line of code. You fill in the rest of the handler routine to service the event. The following is an example of a event handler subroutine.

   ```vba
   Private Sub OnChannelEvent( eventID as Variant, channelNumber as Variant)
   Select Case (eventID)
   Case naEventID_CHANNEL_TRIGGER_COMPLETE:
     GetData( channelNumber )
   Case naEventID_CHANNEL_TRIGGER_ABORTED:
     MsgBox( "Hey don't touch the front panel!"
   End Select
   End Sub
   ```

   When the trigger is complete, the application object "fires" the event by making a callback to the event handler Sub OnChannelEvent().

**Filtering Events**

There are over 140 different events that you subscribe to when you "Dim WithEvents..." (or the equivalent in your programming language). Monitoring all of these conditions slows the speed of the analyzer significantly. The following methods allow you to filter the events so that you only monitor specific conditions.
- **AllowEventMessage** - monitor a specific event
- **AllowAllEvents** - monitor ALL events
- **DisallowAllEvents** - monitor NO events
- **AllowEventCategory** - monitor specific event categories (discussed later)
- **AllowEventSeverity** - monitor events having one or more of the following severity levels associated with them.

<table>
<thead>
<tr>
<th>Code</th>
<th>Severity Enumeration</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>naEventSeveritySUCCESS - the operation completed successfully</td>
</tr>
<tr>
<td>01</td>
<td>naEventSeverityINFORMATIONAL - events that occur without impact on the measurement integrity</td>
</tr>
<tr>
<td>10</td>
<td>naEventSeverityWARNING - events that occur with potential impact on measurement integrity</td>
</tr>
<tr>
<td>11</td>
<td>naEventSeverityERROR - events that occur with serious impact on measurement integrity</td>
</tr>
</tbody>
</table>

**List of Events**

The following is a list of categories and the general types of events they include. Click the link view the event details.

<table>
<thead>
<tr>
<th>Category Enumeration</th>
<th>Callback</th>
</tr>
</thead>
<tbody>
<tr>
<td>naEventCategory_PARSER</td>
<td>OnSCPIEvent</td>
</tr>
<tr>
<td>naEventCategory_MEASURE</td>
<td>OnMeasurementEvent</td>
</tr>
<tr>
<td>naEventCategory_CHANNEL</td>
<td>OnChannelEvent</td>
</tr>
<tr>
<td>naEventCategory_HW</td>
<td>OnHardwareEvent</td>
</tr>
<tr>
<td>naEventCategory_CAL</td>
<td>OnCalEvent</td>
</tr>
<tr>
<td>naEventCategory_USER</td>
<td>OnUserEvent</td>
</tr>
<tr>
<td>naEventCategory_DISPLAY</td>
<td>OnDisplayEvent</td>
</tr>
<tr>
<td>naEventCategory_GENERAL</td>
<td>OnSystemEvent</td>
</tr>
</tbody>
</table>

**Note:** Use the **MessageText** Method to get a text message describing the event.

**Out of Range Errors**

When you attempt to set a value on an active function that is beyond the range (min or max) of the allowable values, the analyzer limits that value to an appropriate value (min or max) and sets the function to the limited value. From the front panel controls this is visually evident by the limited value in the edit box or by the annotation on the display. An example would be attempting to set the start frequency below 300kHz. The edit control doesn't allow the number to fall below 300kHz.

When the automation user programs a setting (such as start frequency below the allowable limits) the same behavior takes place. The analyzer accepts the limited value. However, in order to learn what setting took place, you have to read the HRESULT.
All automation calls return HRESULTs. By default the HRESULT returned when an overlimit occurs is S_NA_LIMIT_OUTOF_RANGE. This value is a success code, meaning that bit 31 in this 32 value is 0. Programmers should check the return code from all automation calls to determine success or failure.

Some C++ macros (like SUCCEEDED(hr) or FAILED(hr) ) only check bit 31. So if you are interested in trapping this outOfRange error you will have to check for S_NA_LIMIT_OUTOF_RANGE explicitly.

Alternatively, you can configure the analyzer to report outOfRange conditions with an error code. Use the method: App.SetFailOnOverRange (true). With this method set TRUE, any overrange error will return E_NA_LIMIT_OUTOF RANGE_ERROR.

This method is provided for the benefit of VB clients. VB users can't detect specific success codes because the VB runtime strips off the HRESULT and only raises a run time error if bit 31 is set, indicating a fail code.

---

**Troubleshooting Problems with Callbacks**

When you do callbacks, the client PC becomes the server and the analyzer (server) becomes the client. Callbacks can only take place when both server and client are in the same workgroup or in the same domain. See Configure for COM.
Read and Write Calibration Data using COM

Calibration data in the PNA is stored in Cal Sets. Learn more about Cal Sets

You can read or write two types of Calibration data:

- **Error Terms** - calculated data using standard measurement data and the algorithms for the specified cal type.
- **Standard Measurement data** - raw data resulting from the measurement of a calibration standard.

Each of these data are available in the PNA in either variant data or typed data. Learn more about variant and typed data

Other Topics about COM Concepts

**Calibration / Cal Set Interfaces**

There are several interfaces associated with Calibration.

**ICalibrator**

This interface is the original interface provided with the first version of the PNA. It provides remote access to the "Unguided" Calibration wizard. This interface can perform 1 and 2 port calibrations as well as response cals.

This interface can also read and write error terms from/to a Cal Set. However, ICalibrator is NOT recommended for this purpose. The ICalSet2 Interface is better suited for reading and writing error terms.

See a vbscript example of how to perform a 2-port Cal and read the cal data.

**IGuidedCalibration**

This interface provides the methods and properties used by the Guided Calibration wizard. With this interface you can perform multi-port calibrations (1 to 4 port cals), but no response cals.

**ICalSet2** and **ICalData3**

These interfaces provide access to the Cal Set contents. You can read and write error terms with both of these interfaces.

- ICalSet2 uses Variant data, which means it is usable from vbscript.
- ICalData3 uses "typed" data, which means it can be used from any automation engine that can read the type library (VEE, VB, C++, etc.). Typed arguments (such as float or single) are more efficient than variants, so use the ICalData3 interface where better performance is needed.

See a vbscript example of how to read Cal Set data.

**ICalSet3**

This interface provides access to the stimulus attributes of the Cal data: frequency, power, number of points.

These are the stimulus conditions under which the Cal Set was created.
Programming the PNA with C++

The programming information contained in this Help system is aimed at the Visual Basic programmer. VB does a lot of work for the programmer when it comes to managing and accessing components. Using a lower level language like C++ requires a more thorough understanding of the underlying tenets of COM. It is not the intent of this section to teach COM programming. The following is intended to acquaint you with some of the basic concepts you need to know in order to program against COM.

- Initializing COM
- Importing the Type Library
- Creating the Application Object
- Errors
- Events
- Additional Reading
- Example

**Note:** The information in this section assumes development on a Windows OS using Microsoft tools.

---

**Initializing COM**

The first thing you must do before performing any COM transactions is to initialize the COM library. You can do this in a number of ways. The most basic of these is a call to `CoInitialize()` or `CoInitializeEx()`. Alternatively you can use the MFC (Microsoft Foundation Classes) `AfxOleInit()`.

Conversely, before your program exits you must uninitialize COM. You can accomplish this with `CoUninitialize()` or the MFC routine `AfxOleTerm()`.

---

**Importing the Type Library**

To make a component available to the client, the server exports what is called the type library. For the PNA, this file is 835x.tlb. It is located on the PNA's hard drive at `C:\Program Files\Agilent\Network Analyzer\Automation`. See [Configure for COM-DCOM Programming](#).

The type library can be read and deciphered using another COM interface called ITypeLib. VB uses this interface to present, for example, its object browser. Visual C++ can also read type libraries. This is done by importing the type library into your project with a compiler directive:

```cpp
#import "C:\Program Files\Common Files\Agilent\Pna\835x.tlb", named_guids
```

When you compile your program with this statement in it, the compiler creates two other files: `835x.tlh` and `835x.tli`. The first is a header file that contains the type definitions for the PNA's COM interfaces and their methods. The second file contains inline functions that wrap the PNA's interface methods. The wrappers are beneficial in that they contain error reporting for each of the method calls.

`835x.tlh` file defines a smart pointer which you can use to access the PNA's objects. The smart pointer definition
A smart pointer is a term used for a C++ object that encapsulates a pointer used to refer to a COM object. All COM objects derive from the interface IUnknown. This interface has three methods: QueryInterface(), AddRef(), and Release(). The function of the AddRef and Release methods is to maintain a reference count on the object and thus control the object’s lifetime. Anytime you copy or create a reference to a COM object, you are responsible for incrementing its reference count. And likewise, when you are finished using that reference, it is your responsibility to Release it. Smart pointers do this work for you, as shown in the example program. In addition, smart pointers will also perform the QueryInterface call when required. QueryInterface is a method that requests a specific interface from an object. In the example program we gain access to the IArrayTransfer interface of the Measurement object. In the ReadMethod routine, we see this:

```cpp
PTransferData = pMeas;
```

The assignment operator is overloaded for the smart pointer and in reality, this simple statement does this:

```cpp
HRESULT hr = pMeas->QueryInterface( IID_IArrayTransfer, (void**)&pTransferData);
```

Using the existing interface pointer (pMeas) to the object, this call asks the object if it supports the IArrayTransfer interface, and if so to return a pointer to it in pTransferData. Smart pointer makes life easier for the C++ programmer. Read more about smart pointers in Microsoft Developer's Network Library (MSDN).

### Creating the Application Object

The only createable object exported by the PNA is the Application object. Typically this would be done with a call to CoCreateInstance:

```cpp
STDAPIS CoCreateInstance(
   CLSID__IApplication, //Class identifier (CLSID) of the object
   NULL, //Pointer to controlling IUnknown
   CLS_CTX_SERVER, //Context for running executable code
   IID_IApplication, //Reference to the IID of the interface
   (void**)&pNA //Address of output variable that receives
   // the interface pointer requested in riid
);
```

With the smart pointer, this is taken care of with the following call:

```cpp
IApplicationPtr pNA; // declare the smart pointer
pNA = IApplicationPtr("AgilentPNA835x.Application.1");
```

### Errors

All COM method calls are required to return an HRESULT. This is 32 bit long with a specific format.

- The most significant bit indicates success(0) or failure(1).
- The lower 16 bits indicate the specific failure.

Visual Basic strips off the returned HRESULT and raises an error object for non-successful returns. The C++ programmer must himself be diligent about handling errors. You must check the return value of each COM call to ensure its success.

### Events
The Application object sources the INetworkAnalyzerEvents interface. This object is the source for all events. To use events in C++, you must do two things:

1. Implement the INetworkAnalyzerEvents interface - derive an object from INetworkAnalyzerEvents and implement the methods described there.
2. Subscribe to the IConnectionPoint interface of the Application object. - obtain a pointer to the IConnectionPointContainer interface of the Application object and making the following request:

```
FindConnectionPoint( IID_InetworkAnalyzerEvents, &pConnection );
```

A successful call to this interface will return a valid pointer in pConnection. Use this pointer to subscribe to the Application object:

```
pConnect->Advise( IUnknown* punk, DWORD dwCookie);
```

This call provides the server object with a callback address. The IUnknown pointer in this call is the IUnknown pointer of the object that implements the INetworkAnalyzerEvents interface. This is the event sink. The application object needs a pointer to this object in order to call your interface when an event occurs. The `dwCookie` is your subscription key. Use it to unsubscribe (see Unadvise( ) ).

---

**Additional Reading**

- "MSDN" - Microsoft Developer's Network Library
- "Learning DCOM", by Thuan L. Thai, published by O'Reilly(1999)
- "Understanding ActiveX and OLE", by David Chappell, also published by Microsoft Press (1996)

---

**Example**

The example uses the smart pointer created by Microsoft Visual Studio. The calls to CoInitialize and CoUninitialize open and close the COM libraries. In the example, notice that the pointers local to the main routine are explicitly released. When smart pointers go out of scope, they will perform this duty implicitly. However, we are calling CoUninitialize before they have the chance to be destroyed, so we are obliged to release them.

See the example program.
Using COM from .NET

To communicate with the PNA from Microsoft .NET enabled languages such as C# and Visual Basic.NET perform the following steps:

1. **Configure your PC and PNA for COM-DCOM Programming.**

2. Reference the type library within the development environment (see the following exception for managed C++ projects). In the process of referencing the type library, a .NET assembly is created that wraps the PNA type library with a .NET friendly interface. This .NET assembly is called an Interop Assembly.

To run a .NET program on the PNA, you will need to install the .NET framework on the PNA. This can be done by running the dotnetfx.exe program, located at: [http://www.microsoft.com/downloads/details.aspx?FamilyID=262d25e3-f589-4842-8157-034d1e7cf3a3&DisplayLang=en](http://www.microsoft.com/downloads/details.aspx?FamilyID=262d25e3-f589-4842-8157-034d1e7cf3a3&DisplayLang=en)

If you only intend to run .NET programs on a remote computer, then it is not necessary to install the .NET framework on the PNA."

**Exception for managed C++ projects:** To generate the Interop Assembly for managed C++ projects, you must use the tlbimp.exe utility. This utility is described in the MSDN documentation. On your PC, click Start then Run then type: tlbimp.exe 835x.tlb and click OK. After doing this you can use the #using directive to include the Interop Assembly on managed C++ projects.

**Registering the PNA Primary Interop Assembly (PIA) (OPTIONAL)**

The PIA is NOT necessary to communicate with the PNA. The following procedure is useful only when there are two .NET programs that want to share the same PNA interface definitions. Without the PIA, each .NET application would use its own Interop Assembly.

To register the PIA on a machine, you need to have the common language runtime (CLR) installed. This is included with Visual Studio.NET. Then perform the following steps:

**Note:** In the following steps, replace <local directory> with the full path name of the specified file on your PC.

1. Run the PNAProxy.exe program as described in **Configure for COM-DCOM Programming.**

2. On the PNA, copy C:\Program Files\Agilent\Network Analyzer\Automation\AgilentPNA835x.dll to a local directory on your PC. Make a note of this directory.

3. On your PC, click **Start**, then **Run**, then type: regasm <local directory>\AgilentPNA835x.dll and click **OK** to register the dll.

4. Again, click **Start**, then **Run**, then type: gacutil /i <local directory>\AgilentPNA835x.dll and click **OK** to add the assembly to the Global Assembly Cache (GAC).

To **Uninstall the PIA**, perform the following:

1. On your PC, click **Start**, then **Run**, then type: gacutil /u <local directory>\AgilentPNA835x.exe and click **OK** to remove the assembly from the GAC.

2. On your PC, click **Start**, then **Run**, then type: regasm/unregser <local directory>\agilentpna835x.dll and click **OK** to unregister the assembly.
3. To uninstall PNA Proxy.exe use the Add/Remove Programs utility in the control panel.
SCPI Command Tree

See Also

- Example Programs
- IEEE-488.2 Common Commands
- Local Lockout
- New See Calibrating the PNA Using SCPI
- Synchronizing the PNA and Controller

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABORT</td>
<td>Stops all sweeps</td>
</tr>
<tr>
<td>+ CALCulate</td>
<td>Click to hide and show CALC branches</td>
</tr>
<tr>
<td>:CORRection</td>
<td>Electrical Delay and Phase Offset</td>
</tr>
<tr>
<td>:CUSTom</td>
<td>Custom measurements</td>
</tr>
<tr>
<td>:DATA</td>
<td>Sends and queries data.</td>
</tr>
<tr>
<td>:EQUation</td>
<td>Equation Editor</td>
</tr>
<tr>
<td>:FILTER</td>
<td>Time domain gating</td>
</tr>
<tr>
<td>:FORMat</td>
<td>Display format</td>
</tr>
<tr>
<td>:FSIMulator</td>
<td>Balanced measurements and Fixturing</td>
</tr>
<tr>
<td>:FUNCtion</td>
<td>Trace Statistics</td>
</tr>
<tr>
<td>:GCData</td>
<td>Read Gain compression data</td>
</tr>
<tr>
<td>:LIMIT</td>
<td>Limit lines for pass / fail testing</td>
</tr>
<tr>
<td>:MARKer</td>
<td>Marker settings</td>
</tr>
<tr>
<td>:MATH</td>
<td>Math / Memory</td>
</tr>
<tr>
<td>:MIXer</td>
<td>X-axis display for FCA measurements</td>
</tr>
<tr>
<td>:NORMalize</td>
<td>Receiver power cal (Obsolete)</td>
</tr>
<tr>
<td>:OFFSet</td>
<td>Mag and Phase offset</td>
</tr>
<tr>
<td>:PARameter</td>
<td>Create and delete measurements</td>
</tr>
<tr>
<td>:RDATA?</td>
<td>Queries receiver data</td>
</tr>
<tr>
<td>:SMOothing</td>
<td>Point-to-point smoothing</td>
</tr>
<tr>
<td>:TRANsform</td>
<td>Time domain transform</td>
</tr>
<tr>
<td>CONTrol</td>
<td>Interface control and Rear-panel connector control.</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>DISPlay</td>
<td>Display settings</td>
</tr>
<tr>
<td>FORMat</td>
<td>Format for data transfer</td>
</tr>
<tr>
<td>HCOPY</td>
<td>Hardcopy printing</td>
</tr>
<tr>
<td>INITiate</td>
<td>Continuous or manual triggering</td>
</tr>
<tr>
<td>MMEMory</td>
<td>Saves and recalls instrument states</td>
</tr>
<tr>
<td>OUTPut</td>
<td>Turns RF power ON and OFF</td>
</tr>
<tr>
<td>ROUTe</td>
<td>Controls internal switch to reference receiver. (Opt 81)</td>
</tr>
</tbody>
</table>

**SENSe**  Click to hide and show SENSE branches

- **:AVErage**  Sweep Averaging
- **:BANDwidth**  IF Bandwidth
- **:CORRection**  Calibration and other correction settings
- **:COUPle**  Chopped or Alternate sweep
- **:FOM**  Frequency Offset (opt 080)
- **:FREQuency**  Frequency sweep settings
- **:GCSetup**  Gain Compression App (opt 086)
- **:IF**  IF Access settings
- **:IF (PNA-X)**  IF Access settings
- **:MIXer**  FCA measurements (opts 082 and 83)
- **:MULTiplexer**  Controls external test sets.
- **:NOISe (PNA-X)**  Noise Figure (opt 029)
- **:PATH**  Provides access to hardware configuration
- **:POWer**  Receiver attenuation and overpower protection
- **:PULSe (PNA-X)**  Configure internal pulse generators
- **:ROSCillator**  Returns the source of the reference oscillator.
- **:SEGMent**  Segment sweep settings.
- **:SWEeep**  Sweep types
- **:X:VALues**  Returns X-axis values

**SOURce**  Source power to the DUT

**SOURce:POWer:CORR**  Source power Calibration

**STATus**  Reads the PNA status registers

**SYSTem**  Misc PNA capabilities

**TRIGger**  Trigger measurements
IEEE 488.2 Common Commands

*CLS - Clear Status
*ESE - Event Status Enable
*ESE? - Event Status Enable Query
*ESR? - Event Status Enable Register
*IDN? - Identify
*OPC - Operation complete command
*OPC? - Operation complete query
*OPT? - Identify Options Query
*RST - Reset
*SRE - Service Request Enable
*SRE? - Service Request Enable Query
*STB? - Status Byte Query
*TST? - Result of Self-test Query
*WAI - Wait

See Also

- Example Programs
- Synchronizing the PNA and Controller
- SCPI Command Tree

---

*CLS - Clear Status

Clears the instrument status byte by emptying the error queue and clearing all event registers. Also cancels any preceding *OPC command or query. See Status Commands and Reading the Analyzer's Status Registers.

---

*ESE - Event Status Enable

Sets bits in the standard event status enable register. See Status Commands and Reading the Analyzer's Status Registers.

---

*ESE? - Event Status Enable Query

Returns the results of the standard event enable register. The register is cleared after reading it. See Status Commands and Reading the Analyzer's Status Registers.
*ESR - Event Status Enable Register
Reads and clears event status enable register. See Status Commands and Reading the Analyzer's Status Registers.

*IDN? - Identify
Returns a string that uniquely identifies the analyzer. The string is of the form "Agilent Technologies",<model number>,<serial "number">,<software revision>".

Note: Beginning with Rev 6.01, this command now returns the software revision with 6 digits instead of 4. For example, A.06.01.02.

*OPC - Operation complete command
Generates the OPC message in the standard event status register when all pending overlapped operations have been completed (for example, a sweep, or a Default). See Understanding Command Synchronization.

*OPC? - Operation complete query
Returns an ASCII "+1" when all pending overlapped operations have been completed. See Understanding Command Synchronization

*OPT? - Identify Options Query
Returns a string identifying the analyzer option configuration.

*RST - Reset
Executes a device reset and cancels any pending *OPC command or query, exactly the same as a SYSTem:PRESet. The contents of the analyzer's non-volatile memory are not affected by this command.

*SRE - Service Request Enable
Before reading a status register, bits must be enabled. This command enables bits in the service request register. The current setting is saved in non-volatile memory. See Status Commands and Reading the Analyzer's Status Registers.

*SRE? - Service Request Enable Query
Reads the current state of the service request enable register. The register is cleared after reading it. The return value can be decoded using the table in Status Commands. See also Reading the Analyzer's
Status Registers.

*STB? - Status Byte Query

Reads the value of the instrument status byte. The register is cleared only when the registers feeding it are cleared. See Status Commands and Reading the Analyzer's Status Registers.

*TST? - Result of Self-test Query

Returns the result of a query of the analyzer hardware status. An 0 indicates no failures found. Any other value indicates one or more of the following conditions exist. The value returned is the Weight (or sum of the Weights) of the existing conditions. For example:

- If 4 is returned from *TST?, an Overpower condition exists.
- If 6 is returned, both Unleveled and Overpower conditions exist.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>Phase Unlock</td>
<td>The source has lost phaselock. This could be caused by a reference channel open or a hardware failure.</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>Unleveled</td>
<td>The source power is unleveled. This could be a source is set for more power than it can deliver at the tuned frequency. Or it could be caused by a hardware failure.</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>Not used</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>EE Write Failed</td>
<td>An attempted write to the EEPROM has failed. This is possibly caused by a hardware failure.</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>YIG Cal Failed</td>
<td>The analyzer was unable to calibrate the YIG. Either the phaselock has been lost or there has been a hardware failure.</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>Ramp Cal Failed</td>
<td>The analyzer was unable to calibrate the analog ramp generator due to a possible hardware failure.</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>Not used</td>
<td></td>
</tr>
</tbody>
</table>

*WAI - Wait

Prohibits the instrument from executing any new commands until all pending overlapped commands have been completed. See Understanding Command Synchronization
About Triggering

Abort Command

ABORt

(Write-only) Stops all sweeps - then resume per current trigger settings. This command is the same as INITiate:IMMediate (restart) except if a channel is performing a single sweep, ABORt will stop the sweep, but not initiate another sweep.

Learn about Synchronizing the PNA and Controller

<table>
<thead>
<tr>
<th>Examples</th>
<th>ABOR abort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Syntax</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>
Calculate:Correction Commands

Controls error correction functions.

CALC:CORRection:

EDELay [STATe] TYPE OFFSet

TIME MEDium WGCutoff [MAGNitude] PHASE

Click on a blue keyword to view the command details.

See Also

- Example Programs
- New Calibrating the PNA Using SCPI
- Synchronizing the PNA and Controller
- SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

**CALCulate<cnun>:CORRection:EDELay:MEDium <char>**

(Read-Write) Sets the media used when calculating the electrical delay.

**Parameters**

- `<cnun>` Any existing channel number. If unspecified, value is set to 1.
- `<num>` Choose from: COAX for coaxial medium, WAVEguide for waveguide medium.

**Examples**

CALC:CORR:EDEL:MED COAX
calc3:corr:edelay:medium waveguide

**Query Syntax** CALCulate<cnun>:CORRection:EDELay:MEDium?

**Return Type** Character

**Default** COAX
CALCulate<nun>:CORRection:EDELay:TIME <num>

(Read-Write) Sets the electrical delay for the selected measurement.

See Critical Note

Parameters

<nun>  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <nun> is set to 1.

<num>  Electrical delay in seconds. Choose any number between: -10.00 and 10.00

Use SENS:CORR:RVEL:COAX <num> to set Velocity factor.

This parameter supports MIN and MAX as arguments. Learn more.

Examples

CALC:CORR:EDEL:TIME 1NS
calculate2:correction:time 0.5e-12

Query Syntax

CALCulate<nun>:CORRection:EDELay:TIME?

Return Type  Numeric

Default  0 seconds

CALCulate<nun>:CORRection:EDEL:WGCutoff <num>

(Read-Write) Sets the waveguide cutoff frequency used when the electrical delay media is set to WAVEguide. (See CALCulate:CORRection:EDELay:MEDium <char>.)

Parameters

<nun>  Any existing channel number. If unspecified, value is set to 1.

<num>  Waveguide cutoff frequency used with the electrical delay calculation.

This parameter supports MIN and MAX as arguments. Learn more.

Examples

CALC:CORR:EDEL:WGC 18.067 GHz

calculate3:correction:edelay:wgcutoff 14.047 ghz

Query Syntax  CALCulate<nun>:CORRection:EDELay:WGCutoff?

Return Type  Numeric

Default  45 MHz

CALCulate<nun>: CORRection: [STATe] <bool>
(Read-Write) Turns error correction ON or OFF for the selected measurement on the specified channel.

To turn error correction ON or OFF for a channel, use SENS:CORR:STATe.

**See Critical Note**

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` Correction state. Choose from:
  - 0 - Correction OFF
  - 1 - Correction ON

**Examples**

```
CALC:CORR ON
calculate:correction:state off
```

**Query Syntax**
CALCulate<cnum>: CORRection: STATe?

**Return Type**
Boolean

**Default**
Not Applicable

---

**CALCulate<cnum>:CORRection:TYPE <string>**

(Read-Write) Sets the Cal Type for the selected measurement on the specified channel. This is used when a Cal Set is applied. Learn more about applying Cal Types.

- Use SENS:CORR:TYPE:CAT? to list the Cal Types in the PNA.
- Use SENS:CORR:CSET:TYPE:CAT? to list the Cal Types contained in the active Cal Set for the channel.
- Use SENS:CORR:COLL:METH to set the Cal type to perform a new calibration.

**See Critical Note**

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<string>` (String) Cal type. Case sensitive. Use one of the following:

  **For Full Calibrations:**

  This command does not distinguish between TRL and SOLT. The same number of error terms is applied for both Cal Types.
"Full n Port(x,y,z...)"

where

n = the number of ports to calibrate

x,y,z = the port numbers to calibrate

For example:

"Full 7 Port(2,3,4,5,6,7,8)"

For Response Calibrations:

"Response(param)" OR

"ResponseAndIsolation(param)"

Where param =

- S-parameter. For example"

  "Response(S21)"
  "ResponseAndIsolation(A/R)"

- Single or ratioed receivers using either logical receiver notation or physical receiver notation. For example:

  "Response(A)"
  "ResponseAndIsolation(a3/b4)"

For FCA Calibrations:

- "Scalar Mixer Cal"
- "SMC with NO Output Match Correction"
- "SMC with NO Input Match Correction"
- "SMC with NO Match Correction"
- "Vector Mixer Cal"
- "Characterize Mixer Only"

For Gain Compression Cal
where r = receive port; s = source port

- "GCA 2P (r,s)" - full 2-port cal
- "GCA Enh Resp (r,s)" - Enhanced Response Cal

Use a ClassID or GUID.

For example:

- CLSID - "VectorMixerCal.VCMCType"
- GUID - "{2061767B-0FE2-4F6F-86D0-9AB332B18DA5}"

<table>
<thead>
<tr>
<th>Examples</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:CORR:TYPE</td>
<td>&quot;Vector Mixer Cal&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>CALCulate&lt;cnum&gt;:CORRection:TYPE?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>String</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**CALCulate<cnum>:CORRection:OFFSet[:MAGNitude] <num> Superseded**

Note: This command is replaced with SENS:CORR:RPOWer:OFFSet[:AMPLitude].
To set data trace magnitude offset, use CALC:OFFS:MAGN.
This command does NOT function for FCA measurements.

See an example of a Receiver Power Calibration.

(Read-Write)

**For Receiver Power Calibration**, specifies the power level to which the selected (unratioed) measurement data is to be adjusted. This command applies only when the selected measurement is of unratioed power.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Cal power level in dBm. No limits are enforced on this value, but the PNA receivers themselves have maximum and minimum power specifications (that may differ between PNA models) which this value must comply with for a valid receiver power cal.

<table>
<thead>
<tr>
<th>Examples</th>
<th>CALC:CORR:OFFS 10DBM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>calculates1:correction:offset:magnitude maximum</td>
</tr>
</tbody>
</table>
**Query Syntax**  
CALCulate<cnum>:CORRection:OFFSet[:MAGNitude]?

**Return Type**  
Numeric

**Default**  
0dBm

---

**CALCulate<cnum>:CORRection:OFFSet:PHASe <num>[<char>]**  
*Superseded*

**Note:** This command is replaced with **CALC:OFFS:PHASE**

(Read-Write) Sets the phase offset for the selected measurement.

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>`: Offset phase value. Choose any number between: -360 and 360
- `<char>`: Units for phase. OPTIONAL. Choose either:  
  - **DEG** - Degrees (default)  
  - **RAD** - Radians

**Examples**

- **CALC:CORR:OFFS:PHAS 10**  
- **calculate:correction:offset:phase 20rad**

---

**Query Syntax**  
CALCulate:CORRection:OFFSet:PHASe?

**Return Type**  
Numeric, returned value always in degrees

**Default**  
0 degrees

---

Last modified:

- **12-Feb-2008**  
  - Fixed typo
- **9/12/06**  
  - MQ Modified Calc:Corr for multiport.
**Calculate:Custom Commands**

Provides capability to create and modify the following measurements:

- **Frequency Converter Application** (opt 082 and 083)
- **Noise Figure Application** (opt 029) measurements
- **Gain Compression Application** (opt 086)

```
CALCulate:CUSTom:
    DEFine
    MODify
```

**See Also**

- **Example Programs**
- **Synchronizing the PNA and Controller**
- **SCPI Command Tree**

**CALCulate<cnum>:CUSTom:DEFine <Mname>, <type> [,param]**

*(Write-only)* Creates a custom measurement. The custom measurement is not automatically displayed. You must also do the following:

- Use `DISP:WIND:STATe` to create a window if it doesn't already exist.
- Use `DISP:WIND:TRAC:FEED` to display the measurement
- Select the measurement (`CALC:PAR:SEL`) before making additional settings.

See an example using this command to create a VMC and SMC measurement

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<Mname>` Name of the measurement. Any non-empty, unique string, enclosed in quotes.
- `<type>` String. The type of custom measurement. Choose from:
  - "Vector Mixer/Converter"
  - "Scalar Mixer/Converter"
  - "Noise Figure Cold Source"
  - "Gain Compression"
String. Optional parameter specifies the measurement parameter to create.

<table>
<thead>
<tr>
<th>ID</th>
<th>Measurement Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector Mixer/Converter</td>
<td>&quot;S11&quot;&lt;br&gt;&quot;VC21&quot;&lt;br&gt;&quot;S22&quot;</td>
<td>Learn about VMC parameters</td>
</tr>
<tr>
<td>Scalar Mixer/Converter</td>
<td>&quot;S11&quot;&lt;br&gt;&quot;SC21&quot;&lt;br&gt;&quot;SC12&quot;&lt;br&gt;&quot;S22&quot;&lt;br&gt;&quot;Ipwr&quot;&lt;br&gt;&quot;RevIPwr&quot;&lt;br&gt;&quot;Opwr&quot;&lt;br&gt;&quot;RevOPwr&quot;</td>
<td>Learn about SMC parameters</td>
</tr>
<tr>
<td>Gain Compression</td>
<td>&quot;CompIn21&quot;</td>
<td>Input power at the compression point.</td>
</tr>
<tr>
<td></td>
<td>&quot;CompOut21&quot;</td>
<td>Output power at the compression point.</td>
</tr>
<tr>
<td></td>
<td>&quot;CompGain21&quot;</td>
<td>Gain at the compression point.</td>
</tr>
<tr>
<td></td>
<td>&quot;CompS11&quot;</td>
<td>Input Match at the compression point</td>
</tr>
<tr>
<td></td>
<td>&quot;RefS21&quot;</td>
<td>Linear Gain</td>
</tr>
<tr>
<td></td>
<td>&quot;DeltaGain21&quot;</td>
<td>CompGain21 - Linear Gain</td>
</tr>
<tr>
<td></td>
<td>&quot;NF&quot;</td>
<td>Noise figure</td>
</tr>
<tr>
<td></td>
<td>&quot;T-Eff&quot;</td>
<td>Effective noise temperature.</td>
</tr>
<tr>
<td></td>
<td>&quot;DUTRNP&quot;</td>
<td>DUT noise power ratio. (Noise power expressed in Kelvin divided by 290).</td>
</tr>
<tr>
<td></td>
<td>&quot;SYSRNP&quot;</td>
<td>System noise power ratio</td>
</tr>
<tr>
<td>Noise Figure Cold Source</td>
<td>&quot;DUTNPD&quot;</td>
<td>DUT noise power density. (Noise power expressed in dBm/Hz).</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------</td>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>&quot;SYSNPD&quot;</td>
<td>System noise power density.</td>
</tr>
<tr>
<td></td>
<td>&quot;OvrRng&quot;</td>
<td>Indication that the noise receiver is being over powered.</td>
</tr>
<tr>
<td></td>
<td>&quot;T-Rcvr&quot;</td>
<td>Temperature reading (in Kelvin) of the noise receiver board.</td>
</tr>
<tr>
<td>&quot;A,1&quot;, &quot;A,2&quot; ...and so forth</td>
<td>Unratioed parameters; with notation: &quot;receiver, source port&quot;</td>
<td></td>
</tr>
</tbody>
</table>

**Examples**

```
CALC:CUST:DEF 'My VC21', 'Vector Mixer/Converter','S22'
calculate2:custom:define 'MyNF', 'NoiseFigure', 'NF'
```

**Query Syntax**

Not applicable

**Overlapped?**

No

**Default**

Not applicable

```
CALCulate<cnum>:CUSTom:MODify <param>
```
(Write-only) Changes the selected custom measurement to a different parameter.

See an example using this command for a VMC and SMC measurement

**Parameters**

- `<cnum>` Channel of the custom measurement to be changed. First, select the measurement using `CALC:PAR:SEL`.

- `<param>` Parameter to change the custom measurement to. Select a parameter that is valid for the type of measurement. Choose from the same arguments as `Calc:Cust:Def`.

**Examples**

```
SYST:PRES
CALC2:CUST:DEF 'My VC21', 'Vector Mixer/Converter'
CALC:PAR:SEL 'My VC21'
CALC2:CUST:MOD 'S22'
```

**Query Syntax** Not applicable

**Overlapped?** No

**Default** Not applicable

---

Last Modified:

23-Aug-2007  Added Noise and GC Arguments
Calculate:Data Commands

Controls writing and reading PNA measurement data.

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select.

(Write) CALCulate<cnum>:DATA <char>,<data>

(Read) CALCulate<cnum>:DATA? <char>

Reads or writes Measurement data, Memory data, or Normalization Divisor data from the Data Access Map location.

- For Measurement data, use FDATA or SDATA
- For Memory data, use FMEM or SMEM. When querying memory, you must first store a trace into memory using CALC:MATH:MEMorize.
- For Normalization Divisor (Receiver Power Cal error term) data, use SDIV
For write operations, data type depends on FORM:DATA command.

**Note:** The Calc:Data SCORR command to read / write error terms is **Superseded** with SENS:CORR:CSET:DATA. SCORR commands do NOT accommodate greater than 12 error terms.

See Critical Note

**Parameters**

<cnm> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnm> is set to 1.

<char> FDATA Formatted measurement data to or from Data Access Map location Display (access point 2).

- Corrected data is returned when correction is ON.
- Uncorrected data is returned when correction is OFF.
- Returns TWO numbers per data point for Polar and Smith Chart format.
- Returns one number per data point for all other formats.
- Format of the read data is same as the displayed format.

SDATA Complex measurement data.

**Writes** data to Data Access Map location Raw Measurement (access point 0).

- When writing corrected data, and correction is ON, it will be corrected again, resulting in meaningless data.

**Reads** data from Apply Error Terms (access point 1).

- Returns TWO numbers per data point.
- Corrected data is returned when correction is ON.
- Uncorrected data is returned when correction is OFF.

FMEM Formatted memory data to or from Data Access Map location Memory result (access point 4).

- Returns TWO numbers per data point for Polar and Smith Chart format.
- Returns one number per data point for all other formats.
- Format of the read data is same as the displayed format.
- Returned data reflects the correction level (On|OFF) when the data was
SMEM  Complex measurement data to or from Data Access Map location Memory (access point 3).

- Returns TWO numbers per data point.
- Returned data reflects the correction level (On|OFF) when the data was stored into memory.
- Returned data reflects the correction level (On|OFF) when the data was stored into memory.

SDIV  Complex data from Data Access Map location Normalization (5).

- Returns TWO numbers per data point.
- If normalization interpolation is ON and the number of points changes after the initial normalization, the divisor data will then be interpolated.
- When querying the normalization divisor, you must first store a divisor trace using CALC:NORMalize[:IMMediate].

The following Calc:Data SCORR command to read / write error terms is Superseded with SENS:CORR:CSET:DATA. These SCORR commands do NOT accommodate greater than 12 error terms.

<table>
<thead>
<tr>
<th>Specity this</th>
<th>to get or put this Error Term...</th>
</tr>
</thead>
<tbody>
<tr>
<td>For 2-Port SOLT and TRL calibrations</td>
<td>&lt;char&gt;</td>
</tr>
<tr>
<td>SCORR1</td>
<td>Forward Directivity</td>
</tr>
<tr>
<td>SCORR2</td>
<td>Forward Source Match</td>
</tr>
<tr>
<td>SCORR3</td>
<td>Forward Reflection Tracking</td>
</tr>
<tr>
<td>SCORR4</td>
<td>Forward Isolation</td>
</tr>
<tr>
<td>SCORR5</td>
<td>Forward Load Match</td>
</tr>
<tr>
<td>SCORR6</td>
<td>Forward Transmission Tracking</td>
</tr>
<tr>
<td>SCORR7</td>
<td>Reverse Directivity</td>
</tr>
<tr>
<td>SCORR8</td>
<td>Reverse Source Match</td>
</tr>
<tr>
<td>SCORR9</td>
<td>Reverse Reflection Tracking</td>
</tr>
<tr>
<td>SCORR10</td>
<td>Reverse Isolation</td>
</tr>
<tr>
<td>SCORR11</td>
<td>Reverse Load Match</td>
</tr>
<tr>
<td>SCORR12</td>
<td>Reverse Transmission Tracking</td>
</tr>
</tbody>
</table>
EXAMPLE

CALC:DATA FDATA, Data(x)
calculate2: data sdata, data(r,i)

See another example using this command.

Return Type: Block data

Default - Not Applicable

CALCulate<cnum>:DATA:CUSTom <name>,<data>

(Read-Write) Reads or writes data from a custom-named measurement buffer.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <name> Name of the buffer to be read or written
- <data> Data to be read or written to the custom buffer. Format as one number per data point.

Examples

CALC:DATA:CUST 'VectorResult0', 0, 1, 2, 3, 4, 5' Write
CALC:DATA:CUST? 'VectorResult0' 'Read

Query Syntax
CALCulate:DATA:CUSTom? <name>

Return Type
Depends on Form: Data

Default Not Applicable

CALCulate<cnum>:DATA:CUSTom:CATalog?
(Read-only) Reads the list of buffer names (comma separated list of string values) available from the selected parameter. Specify the measurement using \texttt{CALCulate:PARameter:SELect}.

See Critical Note

**Parameters**

- \texttt{<cnum>} Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \texttt{<cnum>} is set to 1.

**Examples**

- \texttt{CALC:DATA:CUST:CAT?}
- \texttt{calculate:data:custom:catalog?}

**Return Type** String

**Default** Not Applicable

---

\texttt{CALCulate<cnum>:DATA:SNP? <n>} Superseded

**Note:** This command has been replaced by \texttt{CALC:DATA:SNP:PORTs}.

(Read-only) Reads SnP data from the selected measurement. Learn more about SnP data.

**Note:** This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude data, then all Sx1 phase data, and so forth.

See Critical Note

**Parameters**

- \texttt{<cnum>} Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \texttt{<cnum>} is set to 1.

- \texttt{<n>} Amount of data to return. If unspecified, \texttt{<n>} is set to 2. The number you specify must be less than or equal to the number of available ports on the PNA.

Choose from:

- \textbf{1} (S1P) returns data for the active measurement.

- \textbf{2} (S2P) returns data for the 2 port parameters associated with the current measurement. Default.

- \textbf{3} (S3P) returns data for the 3 port parameters associated with the current measurement.

- \textbf{4} (S4P) returns data for the 4 port parameters associated with the current
SnP data can be output using several data formatting options. See MMEM:STORe:TRACe:FORMat:SNP.

See also MMEM:STOR <file>,<snp>

Examples

```
CALC:PAR:DEF "MyMeasurement", S11
CALC:PAR:SEL "MyMeasurement"
CALC::DATA:SNP? 1
```

Return Type

**Default** Not Applicable

---

**CALCulate<cnum>:DATA:SNP:PORTs? <"x,y,z".>

**Note:** This command replaces CALC::DATA:SNP?. This command is more explicit regarding the data to be returned, and works for PNAs with multiport test sets.

(Read-only) Reads SNP data from the selected measurement for the specified ports. Learn more about SnP data.

**Note:** This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude data, then all Sx1 phase data, and so forth.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<"x,y,z">` Comma or space delimited port numbers for which data is requested, enclosed in quotes.

SnP data can be output using several data formatting options. See MMEM:STORe:TRACe:FORMat:SNP.

**Examples**

```
CALC::DATA:SNP:PORTs? "1,2,4,5,7" 'read data for these ports
```

**Return Type**

**Default** Not Applicable
CALCulate<cnum>:DATA:SNP:PORTs:SAVE <"x,y,z">,<filename>

**Note:** This command replaces MMEM:STOR sNp. This command is more explicit regarding the data to be saved, and works for PNAs with multiport test sets.

*(Write-only)* Saves sNp data from the selected measurement for the specified ports.

Learn more about SNP data.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<"x,y,z">` **String** Comma or space delimited port numbers for which data is requested, enclosed in quotes.
- `<filename>` **String** Path, filename, and suffix of location to store the sNp data. The suffix is not checked for accuracy. If saving 2 ports, specify "filename.s2p"; If saving 4 ports, specify "filename.s4p.", and so forth.

SnP data can be output using several data formatting options. See MMEM:STORe:TRACe:FORMat:SNP.

**Examples**

```
CALC:DATA:SNP:PORTs:Save '1,2,4','C:/Program Files/Agilent/Network Analyzer/Documents/MyData.s3p'
```

**Return Type** Depends on FORMat:DATA

**Default** Not Applicable

---

Last modified:

- April 26, 2007 Added clarification to Calc:Data SDATA
- 9/18/06 MQ Added two SNP Ports commands for multiport
**Calculate: Equation Commands**

Controls Equation Editor capabilities.

![CALCulate:EQUation](image)

Click on a blue keyword to view the command details.

**See Also**

- Example Programs
- Learn about Equation Editor
- Synchronizing the PNA and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using **Calc:Par:Select**

---

**CALCulate<cnum>:EQUation:STATe <bool>**

*(Read-Write)* Turns ON and OFF the equation on selected measurement for the specified channel. If the equation is not valid, then processing is not performed. Use **CALC:EQUation:VALid?** to ensure that the equation is valid.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<bool>`
  - **ON** (or 1) - turns equation ON.
  - **OFF** (or 0) - turns equation OFF.

**Examples**

- **CALC:EQU:STAT 1**
- **calculate2:equation:state 0**

**Query Syntax**

CALCulate<cnum>:EQUation:STATE?
CALCulate<cnum>:EQUation:TEXT <string>

(Read-Write) Specifies an equation or expression to be used on the selected measurement for the specified channel.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <string> Any valid equation or expression. See Equation Editor.

Examples

'|Equation (includes '=')
CALC:EQU:TEXT "foo=S11/S21"

'|Expression
calculate2:equation:text "S11/S21"

Query Syntax
CALCulate<cnum>:EQUation:TEXT?

Return Type String
Default Not Applicable

CALCulate<cnum>:EQUation:VALid?
(Read-Write) Returns a boolean value to indicate if the current equation on the selected measurement for the specified channel is valid. For equation processing to occur, the equation must be valid and ON (CALC:EQU:STAT 1).

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

**Examples**

```
CALC:EQU:VAL?
calculate2:equation:valid?
```

**Return Type** Boolean

- 1 - equation is valid
- 0 - equation is NOT valid

**Default** Not Applicable
**Calculate:Filter Commands**

Controls the gating function used in time domain measurements. The gated range is specified with either (start / stop) or (center / span) commands.

```
CALCulate:FILTER[:GATE]

COUPLE

TIME

CENTer  SPAN  STATE  [TYPE]

PARameters  SHAPe  START  STOP
```

Click on a blue keyword to view the command details.

**see Also**

- [Example Programs](#)
- [Learn about Gating](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using **Calc:Par:Select**

```
CALCulate<cnum>:FILTER[:GATE]:COUPLE:PARameters <num>
```
(Read-Write) Specifies the time domain gating parameters to be coupled. The settings for those parameters will be copied from the selected measurement to all other measurements on the channel.

- To enable Trace Coupling, use `SENS:COUP:PAR`
- To specify Transform parameters to couple, use `CALC:TRAN:COUP:PAR`

Learn more about **Time Domain Trace Coupling**

**See Critical Note**

**Parameters**

| <cnum> | Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1. |
| <num>  | (Numeric) Parameters to couple. To specify more than one parameter, add the numbers. |

1 - Gating Stimulus (Start, Stop, Center, and Span TIME settings.)
2 - Gating State (ON / OFF)
4 - Gating Shape (Minimum, Normal, Wide, and Maximum)
8 - Gating Type (Bandpass and Notch)

**Examples**

'To couple all parameters:
`CALC:FILT:COUP:PAR 15`

'To couple Stimulus and Type:
`calculate2:filter:gate:couple:parameters 9`

**Query Syntax**

`CALCulate<cnum>:FILTer[:GATE]:TIME:CENTer <num>`
(Read-Write) Sets the gate filter center time.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Center time in seconds; Choose any number between: ± (number of points-1) / frequency span

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

```
CALC:FILT:GATE:TIME:CENT -5 ns
calculate2:filter:time:center maximum
```

Query Syntax

CALCulate<cnum>:FILTer[:GATE]:TIME:CENTer?

Return Type

Numeric

Default

0

CALCulate<cnum>:FILTer[:GATE]:TIME:SHAPe <char>

(Read-Write) Sets the gating filter shape when in time domain.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<char>` Choose from
  - MAXimum - the widest gate filter available
  - WIDE -
  - NORMal -
  - MINimum - the narrowest gate filter available

Examples

```
CALC:FILT:GATE:TIME:SHAPE MAX
calculate2:filter:time:shape normal
```

Query Syntax

CALCulate<cnum>:FILTer[:GATE]:TIME:SHAPe?

Return Type

Character

Default

NORMal
CALCulate<cnum>:FILTER[:GATE]:TIME:SPAN <num>

(Read-Write) Sets the gate filter span time.

See Critical Note

Parameters

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<num>** Time span in seconds; Choose any number between: 0 and \(2^*\frac{\text{number of points}-1}{\text{frequency span}}\)

**Note:** This command will accept **MIN** or **MAX** instead of a numeric parameter. See SCPI Syntax for more information.

Examples

```
CALC:FLT[:GATE]:TIME:SPAN 5 ns
calculate2:filter:time:span maximum
```

Query Syntax

CALCulate<cnum>:FILTER[:GATE]:TIME:STATE?

Return Type

Numeric

**Default**

20 ns

CALCulate<cnum>:FILTER[:GATE]:TIME:STATE <boolean>

(Read-Write) Turns gating state ON or OFF.

See Critical Note

**Note:** Sweep type must be set to **LInear Frequency** in order to use Transform Gating.

Parameters

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<boolean>**
  - **ON** (or 1) - turns gating ON.
  - **OFF** (or 0) - turns gating OFF.

Examples

```
CALC:FLT:TIME:STAT ON
calculate2:filter:gate:time:state off
```

Query Syntax

CALCulate<cnum>:FILTER[:GATE]:TIME:STATE?

Return Type

Boolean (1 = ON, 0 = OFF)

**Default**

OFF
CALCulate<cnum>:FILTer[:GATE]:TIME:STARt <num>

(Read-Write) Sets the gate filter start time.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Start time in seconds; any number between: ± (number of points-1) / frequency span

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

CALC:FILT:TIME:STAR 1e-8
calculate2:filter:gate:time:start minimum

Query Syntax

CALCulate<cnum>:FILTer[:GATE]:TIME:STARt?

Return Type

Numeric

Default

10 ns

---

CALCulate<cnum>:FILTer[:GATE]:TIME:STOP <num>

(Read-Write) Sets the gate filter stop time.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Stop time in seconds; any number between: ± (number of points-1) / frequency span

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

CALC:FILT:TIME:STOP -1 ns
calculate2:filter:gate:time:stop maximum

Query Syntax

CALCulate<cnum>:FILTer[:GATE]:TIME:STOP?

Return Type

Numeric

Default

10 ns
CALCulate\(<cnum>\):FILTer\([:GATE]\):TIME\([:TYPE]\) \<char>\

(Read-Write) Sets the type of gate filter used.

See Critical Note

Parameters

\(<cnum>\) Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<cnum>\) is set to 1.

\(<char>\) Choose from:

**BPASs** - Includes (passes) the range between the start and stop times.

**NOTCh** - Excludes (attenuates) the range between the start and stop times.

Examples

```
CALC:FILT:TIME BPAS
calculate2:filter:gate:time:type notch
```

Query Syntax

CALCulate\(<cnum>\):FILTer\([:GATE]\):TIME\([:TYPE]\)?

Return Type

Character

Default

BPAS
Calculate:Format Command

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

See Also

- Example using this command.
- Learn About Data Format
- Synchronizing the PNA and Controller

CALCulate<cnum>:FORMat <char>

(Read-Write) Sets the display format for the measurement.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <char> Choose from:
  - MLINear
  - MLOGarithmic
  - PHASE
  - UPHase 'Unwrapped phase
  - IMAGinary
  - REAL
  - POLar
  - SMITh
  - SADMittance 'Smith Admittance
  - SWR
  - GDELay 'Group Delay
  - KELVin
  - FAHRenheit
  - CELSius

Examples

CALC:FORM MLIN
calculate2:format polar
**Query Syntax**  CALCulate<cnunm>:FORMat?

**Return Type**  Character

**Default**  MLINear

---

Last Modified:

1-Oct-2007  Added temperature formats
**Calculate:FSimulator Commands**

Specifies settings and fixturing for Balanced Measurements.

```
CALCulate:FSIMulator:STATe <bool>
```

(Read-Write) Turns all three fixturing functions (de-embedding, port matching, impedance conversion) ON or OFF for all ports on the specified channel. Does not affect port extensions.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<bool>` Choose from:
  - **ON or 1** - Turns Fixturing ON
  - **OFF or 0** - Turns Fixturing OFF

**Examples**

```
CALC:FSIM:STAT 1
```

```
calculate2:fsimulator:state 0
```

**Query Syntax**

```
CALCulate<cnum>:FSIMulator:STATe?
```

**Return Type**

Boolean

**Default**

OFF
**Calculate:Function Commands**

CALC : FUNCTION

DATA? DOMEin EXECute STATistics TYPE

USER [STATE]

[RANGE] START STOP

Click on a blue keyword to view the command details.

**see Also**

- Example Programs
- Learn about Trace Statistics
- Synchronizing the PNA and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

---

**CALCulate<nnum>:FUNCTION:DATA?**

*(Read-only)* Returns the trace statistic data for the selected statistic type for the specified channel. Select the type of statistic with CALC:FUNC:TYPE.

**See Critical Note**

**Parameters**

- `<nnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<nnum>` is set to 1.

**Return Type** Depends on FORM:DATA

**Example**

CALCulate2:FUNCTION:DATA?

**Default** Not applicable

---

**CALCulate<nnum>:FUNCTION:DOMain:USER[:RANGE] <range>**
(Read-Write) Sets the range used to calculate trace statistics. Each channel has 16 user ranges. The x-axis range is specified with the `CALC:FUNC:DOM:USER:START` and `STOP` commands.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<range>` Range number. Choose from: 0 to 16
  - 0 is Full Span of the current x-axis range
  - 1 to 16 are user-specified ranges

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FUNC:DOM:USER 4</td>
<td>Choose channel 4</td>
</tr>
<tr>
<td>calculate2:function:domain:user:range 0</td>
<td>Set range to 0</td>
</tr>
</tbody>
</table>

**Query Syntax**

`CALCulate<cnum>:FUNCtion:DOMain:USER[:RANGe]?

**Return Type**

Numeric

**Default**

0 - Full Span

---

**CALCulate<cnum>:FUNCtion:DOMain:USER:START <range>, <start>**

(Read-Write) Sets the start of the specified user-domain range.

To apply this range, use `CALC:FUNC:DOM:USER`.

To set the stop of the range, use `CALC:FUNC:DOM:USER:STOP`.

**See Critical Note**

**Note:** This command does the same as `CALC:MARK:FUNC:DOM:USER:STAR`

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<range>` Range number that will receive the start value. Choose an integer between 1 and 16
- `<start>` Start value of the specified range. Choose a real number between: the analyzer's Minimum and Maximum x-axis value.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FUNC:DOM:USER:STAR 1,1e9</td>
<td>Set start value to 1e9</td>
</tr>
<tr>
<td>calculate2:function:domain:user:start 2,2e9</td>
<td>Set start value to 2e9</td>
</tr>
</tbody>
</table>
**Query Syntax**  
CALCulate<cnum>:FUNCtion:DOMain:USER:STARt? <range>

**Return Type**  
Numeric

**Default**  
The analyzer's **Minimum** x-axis value

---

**CALCulate<cnum>:FUNCtion:DOMain:USER:STOP <range>, <stop>**

*(Read-Write)* Sets the stop value of the specified user-domain range.

To apply this range, use **CALC:FUNC:DOM:USER**.

To set the start of the range, use **CALC:FUNC:DOM:USER:START**

**See Critical Note**

**Note:** This command does the same as **CALC:MARK:FUNC:DOM:USER:STOP**

**Parameters**

- **<cnum>**  
  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

- **<range>**  
  Range number that will receive the stop value. Choose an integer between 1 and 16

- **<stop>**  
  Stop value of the specified range. Choose a real number between: the analyzer's **Minimum** and **Maximum** x-axis value.

**Examples**

- **CALC:FUNC:DOM:USER:STOP** 4,5e9
- **calculate2:function:domain:user:stop 3,8e9**

**Query Syntax**  
CALCulate<cnum>:FUNCtion:DOMain:USER:STOP? <range>

**Return Type**  
Numeric

**Default**  
The analyzer's **Maximum** x-axis value

---

**CALCulate<cnum>:FUNCtion:EXECute**
(Write-only) For the active trace of specified channel, executes the statistical analysis specified by the \texttt{CALC:FUNC:TYPE} command.

See Critical Note

**Parameters**

- \texttt{<cnum>} Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \texttt{<cnum>} is set to 1.

**Examples**

- \texttt{CALC:FUNC:EXEC}
- \texttt{calculate2:function:execute}

**Query Syntax** Not Applicable

**Default** Not Applicable

\textit{CALCulate<cnum>:FUNCTION:STATistics[:STATE]<ON|OFF>}

(Read-Write) Displays and hides the trace statistics (peak-to-peak, mean, standard deviation) on the screen.

The analyzer will display either measurement statistics or Filter Bandwidth statistics; not both.

See Critical Note

**Parameters**

- \texttt{<cnum>} Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \texttt{<cnum>} is set to 1.

- \texttt{<ON|OFF>} ON - Displays trace statistics
  
  OFF - Hides trace statistics

**Examples**

- \texttt{CALC:FUNC:STAT ON}
- \texttt{calculate2:function:statistics:state off}

**Query Syntax** \texttt{CALCulate<cnum>:FUNCTION:STATistics[:STATE]?}

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF (0)

\textit{CALCulate<cnum>:FUNCTION:TYPE <char>
(Read-Write) Sets statistic TYPE that you can then query using CALC:FUNCtion:DATA?.

**Note:** In PNA releases 4.2 and prior, this command applied the statistic type to all measurements. Now, this command affects only the selected measurement on the specified channel.

**See Critical Note**

**Parameters**

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <char> Choose from:
  
  **PTPeak** - the difference between the max and min data points on the trace.
  
  **STDEV** - standard deviation of all data points on the trace
  
  **MEAN** - mean (average) of all data points on the trace
  
  **MIN** - lowest data point on the trace
  
  **MAX** - highest data point on the trace

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:FUNC:TYPE PTP</td>
</tr>
<tr>
<td>calculate2:function:type stdev</td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cnum>:FUNCtion:TYPE?

**Return Type**

Character

**Default**

PTPeak
Calc:GCDate Commands

Reads 2-dimensional Gain Compression data.

<table>
<thead>
<tr>
<th>CALCulate:GCData:</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA</td>
</tr>
<tr>
<td>IMAG</td>
</tr>
<tr>
<td>REAL</td>
</tr>
</tbody>
</table>

Click on a blue keyword to view the command details.

Other Gain Compression commands

The calibration commands listed in this topic are supplemental to the Guided Cal commands. See an example: ?????

- **CALC:CUStom:DEFine** - creates a gain compression measurement.
- **SENS:GCSetup** - Most Gain Compression settings.
- Calibrate a GCA measurement
- Gain compression data can also be saved to a *.csv file. Learn how.

See Also

- Example Programs
- Learn about Gain Compression Application
- Synchronizing the PNA and Controller
- SCPI Command Tree

CALCulate<ch>:GCData:DATA? <param>
(Read-Only) Returns measurement data at all frequency and power data points for the current Gain Compression channel.

The format of the data is the same as the format of the measurement that you select using Calc:Par:Select. If the measurement is scalar, than one number is returned per data point. If complex (such as Smith Chart format) than both real and imaginary numbers are returned.

If correction is on, corrected data are returned. Otherwise, raw data are returned.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<param>` (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed or selected. Choose from:
  - "pin" - input power at specified compression level.
  - "pout" - output power at specified compression level.
  - "gain" - device gain (S21) at specified compression level.
  - "inputmatch" - input match (S11) at specified compression level.

**Examples**

```
data = CALC:GCD:DATA? "pin"
data = calculate:gcdata:data? "pout"
```

**Return Type**

Array of data

**Default**

Not Applicable

**CALCulate<ch>:GCDData:IMAG? <char>, <dpoint>, <param>**
(Read-Only) Returns the imaginary part of the specified Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
  - **FREQuency** - for the specified frequency data point, returns all of the measured data for each power stimulus.
  - **POWer** - for the specified power data point, returns all of the measured data for each frequency stimulus.

- `<dPoint>` Data point (FREQ or POWer) for which data is returned.
- `<param>` (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed.
  - "pin" - input power at each data point.
  - "pout" - output power at each data point.
  - "gain" - device gain (S21) at each data point.
  - "inputmatch" - input match (S11) at each data point.

**Examples**

For the fifth frequency data point, returns 'Power Output' imaginary (phase) data from all power stimulus values. If there are 30 power sweep points, 30 values are returned.

\[
data = \text{CALC:GCD:IMAG? FREQ,5,"pout"}
\]

For the 30th stimulus power data point, returns 'Power Output' imaginary (phase) data from all frequency stimulus values. If there are 201 power sweep points, 201 values are returned.

\[
data = \text{calculate:gcdata:imag? power,30,"pout"}
\]

**Return Type**

Array of data

**Default**

Not Applicable

**CALCulate<ch>:GCData:REAL? <char>, <dpoint>, <param>**
(Read-Only) Returns the real part of the specified Gain Compression data. If correction is on, corrected data are returned. Otherwise, raw data are returned.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1
<char> Choose from:
  
  - **FREQuency** - for the specified frequency data point, returns all of the measured data for each power stimulus.
  
  - **POWer** - for the specified power data point, returns all of the measured data for each frequency stimulus.

<dPoint> Data point (FREQ or POWer) for which data is returned.

<param> (String) Parameter to read. NOT Case-sensitive. The specified parameter need NOT be displayed.
  
  - "pin" - input power at each data point.
  
  - "pout" - output power at each data point.
  
  - "gain" - device gain (S21) at each data point.
  
  - "inputmatch" - input match (S11) at each data point.

**Examples**

For the fifth frequency data point, returns 'Power Output' real data from all power stimulus values. If there are 30 power sweep points, 30 values are returned.

```
data = CALC:GCD:REAL? FREQ,5,"pout"
```

For the 30th stimulus power data point, returns 'Power Output' real data from all frequency stimulus values. If there are 201 power sweep points, 201 values are returned.

```
data = calculate:gcdata:real? power,30,"pout"
```

**Return Type** Array of data

**Default** Not Applicable

---

Last Modified:

18-Oct-2007   MX New topic
**Calc:Limit Commands**

Controls the limit segments used for pass / fail testing.

![Limit Commands Diagram]

Click on a blue keyword to view the command details.

**see Also**

- Example Programs
- Learn about Limit Lines
- Synchronizing the PNA and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

---

**CALCulate<cnun>:LIMIT:DATA <block>**

*(Read-Write)* Sets data for limit segments.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<block>` Data for all limit segments in REAL,64 format. The following is the data format for 1 segment:
  - `Type,BegStim, EndStim, BegResp,EndResp`
**Type**  Type of limit segment. Choose from
0 - Off
1 - Max
2 - Min

**BegStim**  Start of X-axis value (freq, power, time)

**EndStim**  End of X-axis value

**BegResp**  Y-axis value that corresponds with Start of X-axis value

**EndResp**  Y-axis value that corresponds with End of X-axis value

**Examples**  The following writes three max limit segments for a bandpass filter.
"CALC:LIM:DATA 1,3e5,4e9,-60,0,1,4e9,7.5e9,0,0,1,7.5e9,9e9,0,-30"

**Query Syntax**  CALCulate<cnum>:LIMit:DATA?

**Return Type**  Depends on FORM:DATA - All 100 predefined limit segments are returned.

**Default**  100 limit segments - all values set to 0

---

**CALCulate<cnum>:LIMit:DISPlay[:STATe] <ON | OFF>**

(Read-Write) Turns the display of limit segments ON or OFF (if the data trace is turned ON).

**See Critical Note**

**Parameters**

*Cnum>  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<ON | OFF>  ON (or 1) - turns the display of limit segments ON.
OFF (or 0) - turns the display of limit segments OFF.

**Examples**  CALC:LIM:DISP:STAT ON
calculate2:limit:display:state off

**Query Syntax**  CALCulate<cnum>:LIMit:DISPlay[:STATe]?

**Return Type**  Boolean (1 = ON, 0 = OFF)

**Default**  ON

---

**CALCulate<cnum>:LIMit:SEGMENT<snum>AMPLitude:STARt <num>**
(Read-Write) Sets the start (beginning) of the Y-axis amplitude (response) value.

**See Critical Note**

**Parameters**
- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<snum>` Segment number; if unspecified, value is set to 1.
- `<num>` Choose any number between: **-500** and **500**

Display value is limited to the Maximum and Minimum displayed Y-axis values.

**Examples**
```
CALC:LIM:SEGM1:AMPL:STAR 10
calculate2:limit:segment2:amplitude:start 10
```

**Query Syntax**
`CALCulate<cnum>:LIMit:SEGMent<snum>AMPLitude:STARt?`

**Return Type** Numeric

**Default** 0

---

**CALCulate<cnum>:LIMit:SEGMent<snum>AMPLitude:STOP <num>**

(Read-Write) Sets the stop (end) of the Y-axis amplitude (response) value.

**See Critical Note**

**Parameters**
- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<snum>` Segment number; if unspecified, value is set to 1.
- `<num>` Choose any number between: **-500** and **500**

Display value is limited to the Maximum and Minimum displayed Y-axis values.

**Examples**
```
CALC:LIM:SEGM1:AMPL:STOP 10
calculate2:limit:segment2:amplitude:stop 10
```

**Query Syntax**
`CALCulate<cnum>:LIMit:SEGMent<snum>AMPLitude:STOP?`

**Return Type** Numeric

**Default** 0
CALCulate<cnum>:LIMit:SEGMENT<snum>STIMulus:STARt <num>

(Read-Write) Sets the start (beginning) of the X-axis stimulus value.

See Critical Note

Parameters

- <cnum>  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <snum>  Segment number; if unspecified, value is set to 1.
- <num>   Choose any number within the X-axis span of the analyzer.

Examples

- CALC:LIM:SEG1:STIM:STAR 10
- calculate2:limit:segment2:stimulus:start 10

Query Syntax

CALCulate<cnum>:LIMit:SEGMENT<snum>STIMulus:STARt?

Return Type

Numeric

Default

0

CALCulate<cnum>:LIMit:SEGMENT<snum>STIMulus:STOP <num>

(Read-Write) Sets the stop (end) of the X-axis stimulus value.

See Critical Note

Parameters

- <cnum>  Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <snum>  Segment number; if unspecified, value is set to 1.
- <num>   Choose any number within the X-axis span of the analyzer.

Examples

- CALC:LIM:SEG1:AMPL:STOP 10
- calculate2:limit:segment2:stimulus:stop 10

Query Syntax

CALCulate<cnum>:LIMit:SEGMENT<snum>STIMulus:STOP?

Return Type

Numeric

Default

0

CALCulate<cnum>:LIMit:SEGMENT<snum>:TYPE <char>
(Read-Write) Sets the type of limit segment.

See Critical Note

Parameters
- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<snum>`: Segment number. Choose any number between: 1 and 100
  - If unspecified, value is set to 1.
- `<char>`: Choose from:
  - **LMAX** - a MAX limit segment. Any response data exceeding the MAX value will fail.
  - **LMIN** - a MIN limit segment. Any response data below the MIN value will fail.
  - **OFF** - the limit segment (display and testing) is turned OFF.

Examples
- `CALC:LIM:SEGM:TYPE LMIN`
- `calculate2:limit:segment3:type lmax`

Query Syntax
- `CALCulate<cnum>:LIMit:SEGMent<snum>:TYPE?`

Return Type
- Character
  - Default: OFF

---

CALCulate<cnum>:LIMit:SOUNd[:STATe] <ON | OFF>

(Read-Write) Turns limit testing fail sound ON or OFF.

See Critical Note

Parameters
- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<ON | OFF>`: Choose:
  - **ON** (or 1) - turns sound ON.
  - **OFF** (or 0) - turns sound OFF.

Examples
- `CALC:LIM:SOUN ON`
- `calculate2:limit:sound:state off`

Query Syntax
- `CALCulate<cnum>:LIMit:SOUNd[:STATe]?`

Return Type
- Boolean (1 = ON, 0 = OFF)
  - Default: OFF
**CALCulate<cnup>:LIMit:STATe <ON | OFF>**

(Read-Write) Turns limit segment testing ON or OFF.

- Use **CALC:LIM:DISP** to turn ON and OFF the display of limit segments.
- If using **Global Pass/Fail** status, trigger the PNA AFTER turning Limit testing ON.

**See Critical Note**

**Parameters**

- **<cnup>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnup> is set to 1.
- **<ON | OFF>** ON (or 1) - turns limit testing ON.
  OFF (or 0) - turns limit testing OFF.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CALC:LIM:STAT ON</strong></td>
</tr>
<tr>
<td><strong>calculate2:limit:state off</strong></td>
</tr>
</tbody>
</table>

**Query Syntax**

CALCulate<cnup>:LIMit:STATe?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF
Calculate:Marker Commands

Controls the marker settings used to remotely output specific data to the computer.

Click on a blue keyword to view the command details.

See Also

- Example Programs
- See Marker Readout number and size commands.
- Learn about Markers
- Synchronizing the PNA and Controller
- SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

Note: The Reference Marker is Marker Number 10

CALCulate<cnum>:MARKer:AOFF
(Write-only) Turns all markers off for selected measurement.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

Examples

- `CALC:MARK:AOFF`
- `calculate2:marker:aoff`

Query Syntax

Not applicable

Default

Not applicable

CALCulate<cnum>:MARKer:BWIDth <num>

(Read-Write) Turns on and sets markers 1 through 4 to calculate filter bandwidth. The `<num>` parameter sets the value below the maximum bandwidth peak that establishes the bandwidth of a filter. For example, if you want to determine the filter bandwidth 3 db below the bandpass peak value, set `<num>` to -3.

This feature activates markers 1 through 4. To turn off these markers, either turn them off individually or turn them All Off.

The analyzer screen will show either Bandwidth statistics OR Trace statistics; not both.

To search a User Range with the bandwidth search, first activate marker 1 and set the desired User Range. Then send the CALC:MARK:BWID command. The user range used with bandwidth search only applies to marker 1 searching for the max value. The other markers may fall outside the user range.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Target value below filter peak. Choose any number between -500 and 500

Examples

- `CALC:MARK:BWID -3`
- `calculate2:marker:bwidth -2.513`

Query Syntax

CALCulate<cnum>:MARKer:BWIDth?

Returns the results of bandwith search:
**Return Type**
Numeric - Four Character values separated by commas: bandwidth, center Frequency, Q, loss.

**Default**
-3

---

**CALCulate<cnum>:MARKer<mkr>:COUPling[:STATe]<ON|OFF>**

*(Read-Write)* Sets and Reads the state of Coupled Markers (ON and OFF)

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any existing marker number from 1 to 10; if unspecified, value is set to 1.
- `<ON|OFF>` **False (0)** - Turns Coupled Markers OFF  
  **True (1)** - Turns Coupled Markers ON

**Examples**

- `CALC:MARK:COUP ON`
- `calculate2:marker8:coupling off`

**Query Syntax**

`CALCulate<cnum>:MARKer<mkr>:COUPling[:STATe]`?

**Return Type**
Boolean (1 = ON, 0 = OFF)

**Default**
OFF

---

**CALCulate<cnum>:MARKer<mkr>:DELTa <ON|OFF>**
(Read-Write) Specifies whether marker is relative to the Reference marker or absolute.

**Note:** The reference marker must already be turned ON with \texttt{CALC:MARK:REF:STATE}.

\textbf{See Critical Note}

**Parameters**

\begin{itemize}
  \item \texttt{<cnum>} Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \texttt{<cnum>} is set to 1.
  \item \texttt{<mkr>} Any existing marker number from 1 to 10; if unspecified, value is set to 1.
  \item \texttt{<ON|OFF>} \texttt{ON} (or 1) - Specified marker is a Delta marker
  \texttt{OFF} (or 0) - Specified marker is an ABSOLUTE marker
\end{itemize}

**Examples**

\begin{verbatim}
CALC:MARK:DELT ON
calculate2:marker8:delta off
\end{verbatim}

**Query Syntax**

\texttt{CALCulate<cnum>:MARKer<mkr>:DELTa?}

**Return Type**

Boolean (1 = ON, 0 = OFF)

\textbf{Default} OFF

\textbf{CALCulate<cnum>:MARKer<mkr>:DISCrete <ON|OFF>}

(Read-Write) Makes the specified marker display either a calculated value between data points (interpolated data) or the actual data points (discrete data).

\textbf{See Critical Note}

**Parameters**

\begin{itemize}
  \item \texttt{<cnum>} Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \texttt{<cnum>} is set to 1.
  \item \texttt{<mkr>} Any existing marker number from 1 to 10; if unspecified, value is set to 1.
  \item \texttt{<ON|OFF>} \texttt{ON} (or 1) - Specified marker displays the actual data points
  \texttt{OFF} (or 0) - Specified marker displays calculated data between the actual data points.
\end{itemize}

**Examples**

\begin{verbatim}
CALC:MARK:DISC ON
calculate2:marker8:discrete off
\end{verbatim}

**Query Syntax**

\texttt{CALCulate<cnum>:MARKer<mkr>:DISCrete?}

**Return Type**

Boolean (1 = ON, 0 = OFF)

\textbf{Default} OFF
CALCulate<cnum>:MARKer<mkr>:DISTance <num>

(Read-Write) Set or query marker distance on a time domain trace.

The Write command moves the marker to the specified distance value. Once moved, you can read the Y axis value or read the X-axis time value. (Distance is calculated from the X-axis time value.)

The Read command reads the distance of the marker.

If the marker is set as delta, the WRITE and READ data is relative to the reference marker.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- `<mkr>` Any existing marker number from 1 to 10; if unspecified, value is set to 1.
- `<num>` Marker distance in the unit of measure specified with CALC:TRAN:TIME:MARK:UNIT

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MARK:DIST .1</td>
<td></td>
</tr>
<tr>
<td>calculate2:marker8:distance 5</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax

CALCulate<cnum>:MARKer<mkr>:DISTance?

Return Type

- Numeric
- Default: Not Applicable

CALCulate<cnum>:MARKer<mkr>:FORMat <char>

(Read-Write) Sets the format of the data that will be returned in a marker data query CALC:MARK:Y? and the displayed value of the marker readout. The selection does not have to be the same as the measurement's display format.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- `<mkr>` Any marker number from 1 to 10; if unspecified, value is set to 1
- `<char>` Choose from: DEFault - The format of the selected measurement
MLINear - Linear magnitude
MLOGarithmic - Logarithmic magnitude
IMPedance - (R+jX)
ADMittance - (G+jB)
PHASE - Phase
IMAGinary - Imaginary part (Im)
REAL - Real part (Re)
Polar - (Re, Im)
GDElay - Group Delay
LINPhase - Linear Magnitude and Phase
LOGPhase - Log Magnitude and Phase
KELVin - temperature
FAHRENheit - temperature

Examples
CALC:MARK:FORMat MLIN
calculate2:marker8:format Character

Query Syntax
CALCulate<cnum>:MARKer<mkr>:FORMat?

Return Type
Character
Default DEFault

CALCulate<cnum>:MARKer<mkr>:FUNCTION:APEak:EXCursion <num>
(Read-Write) Sets amplitude peak excursion for the specified marker. The Excursion value determines what is considered a "peak". This command applies to marker peak searches (Next peak, Peak Right, Peak Left).

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any existing marker number from 1 to 10; if unspecified, value is set to 1.
- `<num>` Excursion value. Choose any number between -500 and 500.

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

- `CALC:MARK:FUNC:APE:EXC 10`
- `calculate2:marker8:function:apeak:excursion maximum`

**Query Syntax**

`CALCulate<cnum>:MARKer<mkr>:FUNCtion:APEak:EXCursion?`

**Return Type** Numeric

**Default** 3

---

**CALCulate<cnum>:MARKer<mkr>:FUNCtion:APEak:THReshold <num>**

(Read-Write) Sets peak threshold for the specified marker. If a peak (using the criteria set with :EXCursion) is below this reference value, it will not be considered when searching for peaks. This command applies to marker peak searches (Next peak, Peak Right, Peak Left).

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 10; if unspecified, value is set to 1
- `<num>` Threshold value. Choose any number between -500 and 500.

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

- `calculate2:marker8:function:apeak:threshold -55`
CALCulate<cnum>:MARKer<mkr>:FUNCtion:DOMain:USER <range>

(Read-Write) Assigns the specified marker to a range number. The x-axis travel of the marker is constrained to the range's span. The span is specified with the CALC:MARK:FUNC:DOM:USER:START and STOP commands, unless range 0 is specified which is the full span of the analyzer.

Each channel has 16 user ranges. (Trace statistics use the same ranges.) More than one marker can use a domain range.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 10; if unspecified, value is set to 1
- `<span>` User span. Choose any Integer from 0 to 16

0 is Full Span of the analyzer
1 to 16 are available for user-defined x-axis span

Examples

- CALC:MARK:FUNC:DOM:USER 1
- calculate2:marker8:function:domain:user 1

Query Syntax


Returns the user span number that the specified marker is assigned to.

Return Type

Numeric

Default

0 - Full Span
(Read-Write) Sets the start of the span that the specified marker's x-axis span will be constrained to.

Use `CALC:MARK:FUNC:DOM:USER<range>` to set range number


**Note:** If the marker is assigned to range 0 (full span), the USER:STARt and STOP commands generate an error. You cannot set the STARt and STOP values for "Full Span".

**Note:** This command does the same as `CALC:FUNC:DOM:USER:STAR`

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 10; if unspecified, value is set to 1
- `<start>` The analyzer's **Minimum** x-axis value

**Examples**

```
calculate2:marker8:function:domain:user:start 1e12
```

**Query Syntax**

```
CALCulate<cnum>:MARKer<mkr>:FUNCtion:DOMain:USER:STARt?
```

**Return Type** Numeric

**Default** The analyzer's **Minimum** x-axis value

---

**CALCulate<cnum>:MARKer<mkr>:FUNCtion:DOMain:USER:STOP <stop>**

(Read-Write) Sets the stop of the span that the marker's x-axis travel will be constrained to.

Use `CALC:MARK:FUNC:DOM:USER<range>` to set range number

Use `CALC:MARK:FUNC:DOM:USER:STARt` to set the stop value.

**Note:** If the marker is assigned to range 0 (full span), the USER:STARt and STOP commands generate an error. You cannot set the STARt and STOP values for "Full Span".

**Note:** This command does the same as `CALC:FUNC:DOM:USER:STOP`

**See Critical Note**

**Parameters**
<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<mkr> Any marker number from 1 to 10; if unspecified, value is set to 1.

<stop> Stop value of x-axis span; Choose any number between the analyzer's MINimum and MAXimum x-axis value.

Examples

```
calculate2:marker8:function:domain1:USER:stop 1e12
```

Query Syntax

CALCulate<cnum>:MARKer<mkr>:FUNCtion:DOMain:USER:STOP?

Return Type Numeric

Default The analyzer's MAXimum x-axis value.

CALCulate<cnum>:MARKer<mkr>:FUNCTION:EXECute [<func>]

(Write-only) Immediately executes (performs) the specified search function. If no function is specified, executes the selected function. Select the function with CALC:MARK:FUNCtion:SEL.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <mkr> Any marker number from 1 to 10; if unspecified, value is set to 1.
- <func> Optional argument. The function that is to be performed. Choose from:

  - MAXimum - finds the highest value
  - MINimum - finds the lowest value
  - RPEak - finds the next valid peak to the right
  - LPEak - finds the next valid peak to the left
  - NPEak - finds the next highest value among the valid peaks
  - TARGet - finds the target value to the right, wraps around to the left
  - LTARget - finds the next target value to the left of the marker
  - RTARget - finds the next target value to the right of the marker
**CALCulate<cnum>:MARKer<mkr>:FUNCtion[:SELect] <char>**

*(Read-Write)* Sets the search function that the specified marker will perform when executed. To execute (or perform) the function, use:

CALC:MARK:FUNC:EXEC or
CALC:MARK:FUNC:TRAC ON to automatically execute the search every sweep.

See Critical Note

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<mkr>** Any marker number from 1 to 10; if unspecified, value is set to 1.
- **<char>** Marker function. Choose from:
  - **MAXimum** - finds the highest value
  - **MINimum** - finds the lowest value
  - **RPEak** - finds the next valid peak to the right
  - **LPEak** - finds the next valid peak to the left
  - **NPEak** - finds the next highest value among the valid peaks
  - **TARGET** - finds the target value to the right, wraps around to the left
  - **LTARGET** - finds the next target value to the left of the marker
  - **RTARGET** - finds the next target value to the right of the marker

**Examples**

```
CALC:MARK:FUNC MAX
calculate2:marker8:function:execute maximum
```

**Query Syntax**

CALCulate<cnum>:MARKer<mkr>:FUNCtion[:SELect]?

**Return Type** Character

**Default** MAX
**CALCulate<cnum>:MARKer<mkr>:TARGet <num>**

(Read-Write) Sets the target value for the specified marker when doing Target Searches with `CALC:MARK:FUNC:SEL <TARget | RTARget | LTARget>`

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 10; if unspecified, value is set to 1.
- `<num>` Target value to search for; Units are NOT allowed.

**Examples**

```
CALC:MARK:TARG 2.5
calculate2:marker8:target -10.3
```

**Query Syntax**

`CALCulate<cnum>:MARKer<mkr>:TARGet?`

**Return Type**

Numeric

**Default**

0

---

**CALCulate<cnum>:MARKer<mkr>:FUNCtion:TRACking <ON | OFF>**

(Read-Write) Sets the tracking capability for the specified marker. The tracking function finds the selected search function every sweep. In effect, turning Tracking ON is the same as doing a `CALC:MARK:FUNC:EXECute` command every sweep.

**See Critical Note**

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 10; if unspecified, value is set to 1.
- `<ON | OFF>`
  - **ON** (or 1) - The specified marker will "Track" (find) the selected function every sweep.
  - **OFF** (or 0) - The specified marker will find the selected function only when the `CALC:MARK:FUNC:EXECute` command is sent.

**Examples**

```
CALC:MARK:FUNC:TRAC ON
calculate2:marker8:function:tracking OFF
```

**Query Syntax**

`CALCulate<cnum>:MARKer<mkr>:FUNCtion:TRACking?`
CALCulate<cnum>:MARKer:REFerence[:STATe] <ON | OFF>

(Read-Write) Turns the reference marker (marker 10) ON or OFF. When turned OFF, existing Delta markers revert to absolute markers.

See Critical Note

Parameters

Parameters

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<ON | OFF>** ON (or 1) - turns reference marker ON
  
  OFF (or 0) - turns reference marker ON

Examples

- **CALC:MARK:REF ON**
- **calculate2:marker:reference:state OFF**

Query Syntax

CALCulate<cnum>:MARKer:REFerence[:STATe]?

Return Type  Boolean (1 = ON, 0 = OFF)

Default  OFF

CALCulate<cnum>:MARKer:REFerence:X <num>

(Read-Write) Sets and returns the absolute x-axis value of the reference marker (marker 10).

See Critical Note

Parameters

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<num>** X-axis value. Choose any number within the operating domain of the reference marker.

Examples

- **CALC:MARK:REF:X 1e9**
- **calculate2:marker:reference:x 1e6**

Query Syntax

CALCulate<cnum>:MARKer:REFerence:X?

Return Type  Numeric
**CALCulate\<cnum>:MARKer:REFerence:Y**?

*(Read-only)* Returns the absolute Y-axis value of the reference marker.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.

**Examples**

```
CALC:MARK:REF:Y?
calculate2:marker:reference:y?
```

**Return Type** Character

**Default** Not applicable

---

**CALCulate\<cnum>:MARKer\<mkr>:TYPE \<char>**

*(Read-Write)* Sets the type of the specified marker.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>`: Any marker number from 1 to 10; if unspecified, value is set to 1
- `<char>`: Choose from:
  - **NORMAL** - a marker that stays on the assigned X-axis position unless moved or searching.
  - **FIXed** - a marker that will not leave the assigned X or current Y-axis position.

**Examples**

```
CALC:MARK:TYPE NORMAL
calculate2:marker2:type fixed
```

**Query Syntax** `CALCulate\<cnum>:MARKer\<mkr>:TYPE?`

**Return Type** Character
CALCulate<cnum>:MARKe<mkr>:SET <char>

(Write-only) Sets the selected instrument setting to assume the value of the specified marker.

Marker Functions CENT, SPAN, STARt, and STOP do not work with channels that are in CW or Segment Sweep mode.

See Critical Note

Parameters

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 10; if unspecified, value is set to 1
- `<char>` Choose from:
  - CENTer - changes center frequency to the value of the marker
  - SPAN - changes the sweep span to the span that is defined by the delta marker and the marker that it references. Unavailable if there is no delta marker.
  - START - changes the start frequency to the value of the marker
  - STOP - changes the stop frequency to the value of the marker
  - RLEVel - changes the reference level to the value of the marker
  - DELay - changes the line length at the receiver input to the phase slope at the active marker stimulus position.

Examples

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALC:MARK:SET CENT</td>
<td>Change center frequency to the value of the marker</td>
</tr>
<tr>
<td>calculate2:marker8:SET span</td>
<td>Set the span to the value of the marker</td>
</tr>
</tbody>
</table>

Query Syntax Not Applicable

Default Not Applicable

CALCulate<cnum>:MARKe<mkr>[STATE] <ON|OFF>
(Read-Write) Turns the specified marker ON or OFF. **Marker 10 is the Reference Marker.** To turn all markers off, use CALC:MARK:AOFF.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>`: Any marker number from 1 to 10; if unspecified, value is set to 1.
- `<ON|OFF>`: 
  - **ON** (or 1) - turns marker ON.
  - **OFF** (or 0) - turns marker OFF.

**Examples**

```
CALC:MARK ON
calculate2:marker8 on
```

**Query Syntax**

`CALCulate<cnum>:MARKer<mkr>:STATe?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

Off

---

**CALCulate<cnum>:MARKer<mkr>:X <num>**

(Read-Write) Sets the marker's X-axis value (frequency, power, or time). If the marker is set as delta, the SET and QUERY data is relative to the reference marker.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>`: Any marker number from 1 to 10; if unspecified, value is set to 1.
- `<num>`: Any X-axis position within the measurement span of the marker.

**Note:** This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

**Examples**

```
CALC:MARK:X 100Mhz
calculate2:marker8:x maximum
```

**Query Syntax**

`CALCulate<cnum>:MARKer<mkr>:X?`

**Return Type**

Numeric

**Default**

First Marker turns ON in the middle of the X-axis span. Subsequent markers turn ON at the position of the active marker.
CALCulate<cnum>:MARKer<mkr>:Y?

(Read-only) Reads the marker's Y-axis value. The format of the value depends on the current CALC:MARKER:FORMAT setting. If the marker is set as delta, the data is relative to the reference marker. The query always returns two numbers:

- Smith and Polar formats - (Real, Imaginary)
- LINPhase and LOGPhase - (Real, Imaginary)
- All other formats - (Value,0)

**Note:** To accurately read the marker Y-axis value with trace smoothing applied, the requested format must match the displayed format. Otherwise, the returned value is un-smoothed data. For example, to read the smoothed marker value when measuring group delay, both the display format and the marker format must be set to (Group) Delay.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<mkr>` Any marker number from 1 to 10; if unspecified, value is set to 1.

**Examples**

CALC:MARK:Y?
calculate2:marker3:y?

**Query Syntax**

CALCulate<cnum>:MARKer<mkr>:Y?

**Return Type**

Numeric

**Default**

Not applicable

Last modified:

1-Oct-2007 Added temperature formats
March 27, 2007 Corrected Set?
Dec. 4, 2006 Added smoothing note to Y?
Calculate: Math Commands

Controls math operations on the currently selected measurement and memory.

CALCulate : MATH

MEMorize  FUNCTION

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Math Operations
- Synchronizing the PNA and Controller
- SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

CALCulate<nnum>:MATH:FUNCtion <char>

(Read-Write) Sets math operations on the currently selected measurement and the trace stored in memory. (There MUST be a trace stored in Memory. See CALC:MATH MEM)

See Critical Note

Parameters

- <nnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <nnum> is set to 1.
- <char> The math operation to be applied. Choose from the following:
  - NORMAL Trace data only
  - ADD Data + Memory
  - SUBTract Data - Memory
  - MULTiply Data * Memory
  - DIVide Data / Memory
### CALC:MATH:FUNC NORM

**Query Syntax**
CALC:MEM

**Return Type**
Character

**Default**
NORMal

### CALC:MEM

(Write-only) Puts the currently selected measurement trace into memory. (Data-> Memory).

**See Critical Note**

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

**Examples**
CALC:MEM
calculate2:memorize

**Query Syntax**
Not applicable

**Default**
Not applicable
Calculate:Mixer Command

CALCulate<ch>:MIXer:XAXis <char>

(Read-Write) Sets or returns the swept parameter to display on the X-axis for the selected FCA measurement. Learn more about X-axis display.

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.
<char> Parameter to display on the X-axis. Choose from:

- **INPUT** - Input frequency span
- **OUTPUT** - Output frequency span
- **LO_1** - First LO frequency span
- **LO_2** - Second LO frequency span

Examples

CALC:MIX:XAX INPUT
calc2:mixer:xaxis output

See an example that creates, selects, and calibrates an SMC and VMC measurement using SCPI.

Query Syntax

CALCulate<ch>:MIXer:XAXis?

Return Type

Character

Default

OUTPUT
**Calculate:Normalize Commands**

Specifies the normalization features used for a receiver power calibration.

These commands are **Superseded** (Sept 2004).

See the replacement commands in a new [Receiver Power Cal example](#).

![CALCulate:NORMalize](image)

Click on a blue keyword to view the command details.

**See Also**

- [Example Programs](#)
- [Learn about Receiver Cal](#)
- [SCPI Command Tree](#)

Save and recall your receiver power calibration (which use .CST file commands):

- `SENS:CORR:CSET:SAVE`
- `SENS:CORR:CSET[:SEL]`

Or use these two commands and specify either .STA or .CST file extensions:

- `MMEM:LOAD`
- `MMEM:STOR`

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using [Calc:Par:Select](#)
Note: This command is replaced with `SENS:CORR:COLL:METH RPOWER` and `SENS:CORR:COLL[:ACQ] POWer`.

See an example of a Receiver Power Calibration.

(Write only) Stores the selected measurement’s data to that measurement’s “divisor” buffer for use by the Normalization data processing algorithm. This command is not compatible with ratioed measurements such as S-parameters. It is intended for receiver power calibration when the selected measurement is of an unratioed power type.

See Critical Note

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
</tr>
<tr>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:NORM</code></td>
</tr>
<tr>
<td><code>calculate1:normalize:immediate</code></td>
</tr>
</tbody>
</table>

Query Syntax

Not Applicable

Default

Not Applicable

---

**CALCulate<cnum>:NORMalize:STATe <ON | OFF>** Superseded

Note: This command is replaced with `SENS:CORR[:STATe] ON|OFF`.

(Read-Write) Specifies whether or not normalization is applied to the measurement. Normalization is enabled only for measurements of unratioed power where it serves as a receiver power calibration.

See Critical Note

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cnum&gt;</code></td>
</tr>
<tr>
<td>Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <code>&lt;cnum&gt;</code> is set to 1.</td>
</tr>
</tbody>
</table>

| `<ON | OFF>` |
|----------|
| **ON (or 1)** - normalization is applied to the measurement. |
| **OFF (or 0)** – normalization is NOT applied to the measurement. |

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CALC:NORM:STAT ON</code></td>
</tr>
<tr>
<td><code>calculate2:normalize:state off</code></td>
</tr>
</tbody>
</table>

Query Syntax

`CALCulate<cnum>:NORMalize:STATe?`  

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF
**CALCulate\(cnum\):NORMalize:INTerpolate[:STATe] <ON | OFF>**  
*Superseded*

**Note:** This command is replaced with **SENS:CORR:INT[:STATe] ON|OFF**

*(Read-Write)* Turns normalization interpolation ON or OFF. Normalization is enabled only for measurements of unratioed power, where it serves as a receiver power calibration.

See Critical Note

**Parameters**

- \(<cnum>\) Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<cnum>\) is set to 1.
- \(<ON | OFF>\) **ON (or 1)** – turns interpolation ON.
  
  **OFF (or 0)** – turns interpolation OFF.

**Examples**

- CALC:NORM:INT ON
- calculate2:normalize:interpolate:state off

**Query Syntax**

CALCulate\(<cnum>:NORMalize:INTerpolate[:STATe]?**

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON
Calculate:Offset Commands

Allows the data trace magnitude and phase to be offset.

CALCulate:OFFSet

MAGNitude PHASE

SLOPe

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Magnitude Offset
- Learn about Phase Offset
- Synchronizing the PNA and Controller
- SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

CALCulate<cnum>:OFFSet:MAGNitude <num>
(Read-Write) Offsets the data trace magnitude by the specified value.
To offset the data trace magnitude to a slope value that changes with frequency, use
**CALC:OFFS:MAGN:SLOP**

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>`: Offset value in dB.

**Examples**

```
CALC:OFFS:MAGN:4
calculate1:offset:magnitude -2
```

**Query Syntax**

`CALCulate<cnum>:OFFSet:MAGNitude?`

**Return Type**

Numeric

**Default**

0

---

**CALCulate<cnum>:OFFSet:MAGNitude:SLOPe <num>**

(Read-Write) Offsets the data trace magnitude to a value that changes linearly with frequency. The offset slope begins at 0 Hz.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>`: Offset slope value in dB/ 1GHz.

**Examples**

```
CALC:OFFS:MAGN:SLOP 1 'Offset slope set to 1dB/GHz
calculate1:offset:magnitude:slope -2 'Offset slope set to -2dB/GHz
```

**Query Syntax**

`CALCulate<cnum>:OFFSet:MAGNitude:SLOPe?`

**Return Type**

Numeric

**Default**

0

---

**CALCulate<cnum>:OFFSet:PHASe <num>[<char>]**
(Read-Write) Sets the phase offset for the selected measurement.

See Critical Note

Parameters

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<num>** Offset phase value. Choose any number between: **-360** and **360**
- **<char>** Units for phase. OPTIONAL. Choose either: **DEG** - Degrees (default) **RAD** - Radians

Examples

```
CALC:OFFS:PHAS 10
calculate:offset:phase 20rad
```

Query Syntax

CALCulate:OFFSet:PHASe?

Return Type

Numeric, returned value always in degrees

**Default** 0 degrees
Calculate:Parameter Commands

Lists, creates, selects and deletes measurements.

```
CALC:PARameter

CATalog  DELete  MNUMber  SELect

DEFine  ALL  MODify

EXTended  EXTended
```

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Measurement Parameters
- Synchronizing the PNA and Controller
- SCPI Command Tree

Critical Note: CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

### CALCulate<cnum>:PARameter:CATalog?

(Read-only) Returns the names and parameters of existing measurements for the specified channel.

Note: For Balanced Measurements: CALC:PAR:CAT? may have an unexpected behavior. Learn more.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurements to be listed. If unspecified, `<cnum>` is set to 1.

**Examples**

```
CALC:PAR:CAT?
calculate2:parameter:catalog?
```

**Return Type** String - "<measurement name>,<parameter>,[<measurement name>,<parameter>...]"
CALCulate<cnum>:PARameter:DEFine <Mname>,<param>[,port]  Superseded

Note: This command is replaced with CALC:PAR:DEFine:EXTended. This command will continue to work for up to 4 port parameters.

(Write-only) Creates a measurement but does NOT display it.

There is no limit to the number of measurements that can be created. However, there is a limit to the number of measurements that can be displayed. See Traces, Channels, and Windows on the PNA.

- Use DISP:WIND:STATe to create a window if it doesn't already exist.
- Use DISP:WIND<wnum>:TRAC<tnum>:FEED <Mname> to display the measurement.

For FCA Measurements see CALC:CUST:DEF

You must select the measurement (CALC<cnum>:PAR:SEL <mname>) before making additional settings.

See Critical Note

Parameters

<cnum> Channel number of the new measurement. If unspecified, value is set to 1.
<Mname> Name of the measurement. Any non-empty, unique string, enclosed in quotes.
<param> For S-parameters:

Any S-parameter available in the PNA

For ratioed measurements:

Any two receivers that are available in the PNA. (See the block diagram showing the receivers in YOUR PNA.)

For example: AR1 (this means A/R1)

For non-ratioed measurements:

Any receiver that is available in the PNA. (See the block diagram showing the receivers in YOUR PNA.)

For example: A
For Balanced Measurements:

First create an S-parameter measurement, then change the measurement using **CALC:FSIM:BAL** commands. See an example.

For FCA Measurements see **CALC:CUST:DEF**

[port] Optional argument;

For multi-port reflection S-parameter measurements: specifies the PNA port which will provide the load for the calibration. This argument is ignored if a transmission S-parameter is specified.

For all non S-parameter measurements: specifies the source port for the measurement.

Examples

```
CALC4:PAR:DEF 'ch4_S33',S33,2 'Defines an S33 measurement with a load on port2 of the analyzer.
```

```
calculate2:parameter:define 'ch1_a', a, 1 'unratioed meas
```

```
calculate2:parameter:define 'ch1_a', ar1,1 'ratioed meas
```

Query Syntax Not Applicable; see **Calc:Par:Cat**

**Default** Not Applicable

---

**CALCulate<cnum>:PARameter:DEFine:EXTended <Mname>,<param>**

**Note:** This command replaces **CALC:PAR:DEF** as it allows the creating of measurements using external multiport testsets.

(Write-only) Creates a measurement but does NOT display it.

There is no limit to the number of measurements that can be created. However, there is a limit to the number of measurements that can be displayed. See Traces, Channels, and Windows on the PNA.

- Use **DISP:WIND:STATe** to create a window if it doesn't already exist.
- Use **DISP:WIND<wnum>:TRAC<tnum>:FEED <Mname>** to display the measurement.

You must select the measurement (CALC<cnum>:PAR:SEL <mname>) before making additional settings.

**See Critical Note**

**Parameters**
<cnum> Channel number of the new measurement. If unspecified, value is set to 1.

<Mname> (String) Name of the measurement. Any non-empty, unique string, enclosed in quotes.

<param> (String) Measurement Parameter to create. Case sensitive.

For S-parameters:

Any S-parameter available in the PNA

Single-digit port numbers CAN be separated by "_" (underscore). For example: "S21" or "S2_1"

Double-digit port numbers MUST be separated by underscore. For example: "S10_1"

For ratioed measurements:

Any two PNA physical receivers separated by forward slash '/' followed by comma and source port.

For example: "A/R1, 3"

Learn more about ratioed measurements

See a block diagram showing the receivers in YOUR PNA.

For non-ratioed measurements:

Any PNA physical receiver followed by comma and source port.

For example: "A, 4"

Learn more about unratioed measurements.

See the block diagram showing the receivers in YOUR PNA.

With PNA Rev 6.2, Ratioed and Unratioed measurements can also use logical receiver notation to refer to receivers. This notation makes it easy to refer to receivers with an external test set connected to the PNA. You do not need to know which physical receiver is used for each test port. Learn more.

For ADC measurements:

Any ADC receiver in the PNA followed by a comma, then the source port.

For example: "AI1,2" indicates the Analog Input1 with source port of 2.
Learn more about ADC receiver measurements.

For Balanced Measurements:

First create an S-parameter measurement, then change the measurement using CALC:FSIM:BAL "define" commands. See an example.

Examples

<table>
<thead>
<tr>
<th>Description</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defines an S33 measurement</td>
<td>CALC4:PAR:DEF:EXT 'ch4_S33', 'S33' 'Defines an S33 measurement</td>
</tr>
<tr>
<td>logical receiver notation for unratioed meas of test port 9 receiver with</td>
<td>calculate2:parameter:define:extended 'ch1_a', 'b9, 1' 'logical</td>
</tr>
<tr>
<td>source port 1.</td>
<td>receiver notation for unratioed meas of test port 9 receiver with</td>
</tr>
<tr>
<td>logical receiver notation for ratioed meas of test port 9 receiver divided</td>
<td>source port 1.</td>
</tr>
<tr>
<td>by the reference receiver for port 10 using source port 1</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax

Not Applicable; see Calc:Par:Cat?

Default

Not Applicable

CALCulate<cnm>:PARameter:DELete[:NAME] <Mname>

(Write-only) Deletes the specified measurement.

See Critical Note

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnm&gt;</td>
<td>Channel number of the measurement. There must be a selected measurement</td>
</tr>
<tr>
<td></td>
<td>on that channel. If unspecified, &lt;cnm&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;Mname&gt;</td>
<td>String - Name of the measurement</td>
</tr>
</tbody>
</table>

Examples

<table>
<thead>
<tr>
<th>Description</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CALC:PAR:DEL 'TEST'</td>
</tr>
<tr>
<td></td>
<td>calculate2:parameter:delete 'test'</td>
</tr>
</tbody>
</table>

Query Syntax

Not Applicable

Default

Not Applicable

CALCulate:PARameter:DELete:ALL
(Write-only) Deletes all measurements on the PNA.

See Critical Note

Parameters

Examples | CALC:PAR:DEL:ALL

Query Syntax Not Applicable

Default Not Applicable

CALCulate<cnum>:PARameter:MNUMber?

(Read-only) Returns the measurement number of the selected measurement. This is useful when needing to identify a measurement by number, such as with Status:Ques:Lim or Status:Oper:Aver commands.

See Critical Note

Parameters

<cnun> Channel number of the measurement. If unspecified, <cnum> is set to 1.

Examples | CALC:PAR:MNUM?
calculate2:parameter:mnumber?

Return Type Numeric

Default Not Applicable

CALCulate<cnum>:PARameter:MODify <param> Superseded

Note: This command is replaced with CALC:PAR:MOD:EXT. This command will continue to work for up to 4 port parameters.

(Write-only) Modifies a standard measurement using the same arguments as CALC:PAR:DEF. To modify an FCA measurement, use CALC:CUST:MOD.

See Critical Note

Parameters

<cnun> Channel number of the measurement. The selected measurement on that channel will be changed. If unspecified, <cnum> is set to 1.

<param> Measurement parameter to change to. Use the same <param> arguments as CALC:PAR:DEF.

Examples | SYST:PRESET
CALC:PAR:DEF "MyMeas", S11
CALC:PAR:SEL "MyMeas"
CALC:PAR:MOD AR1 'changes the selected S11 measurement to an A/R1 measurement

Query Syntax  Not Applicable
Default        Not Applicable

CALCulate<cnum>:PARameter:MODify:EXTended <param>

**Note:** This command replaces CALC:PAR:MOD as it allows modification of measurements using external multiport testsets.

(Write-only) Modifies a standard measurement using the same arguments as CALC:PAR:DEF:EXT. To modify an FCA measurement, use CALC:CUST:MOD.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. The selected measurement on that channel will be changed. If unspecified, `<cnum>` is set to 1.
- `<param>` *(String)* New measurement parameter. Use the same `<param>` arguments as CALC:PAR:DEF:EXT.

**Examples**

SYST:PRESET

CALC:PAR:DEF:EXT "MyMeas", "S10_1"
CALC:PAR:SEL "MyMeas"
CALC:PAR:MOD:EXT "a4b4,1" 'changes the selected S10_1 measurement to an a4/b4 measurement with source port 1

Query Syntax  Not Applicable
Default        Not Applicable

CALCulate<cnum>:PARameter:SESelect <Mname>
(Read-Write) Sets the selected measurement. Most CALC: commands require that this command be sent before a setting change is made. One measurement on each channel can be selected at the same time. To obtain a list of currently named measurements, use CALC:PAR:CAT?

**Parameters**

- `<cnum>` Channel number of the measurement to be selected. If unspecified, `<cnum>` is set to 1.
- `<Mname>` String - Name of the measurement. (Do NOT include the parameter name.)

**Examples**

```
CALC:PAR:SEL 'TEST'
calculate2:parameter:select 'test'
```

**Query Syntax**
CALCulate:PARameter:SELect?

**Return Type**
String

**Default**
The preset measurement name is "CH1_S11_1"

Last modified:

April 19, 2007  MX Added ADC meas
9/12/06  MQ New Extended commands.
**Calculate:RData? Command**

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

CALCulate<cnum>:RDATA? <char>

(Read-only) Returns receiver data for the selected measurement. To query measurement data, see CALC:DATA?

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<char>** Choose from any physical receiver in the PNA.

For example: "A"

Also, **REF** - returns data for either R1 or R2 data depending on the source port of the selected measurement.

See the block diagram showing the receivers in YOUR PNA.

**Note:** Logical receiver notation is NOT allowed with this command. Learn more.

**Example**

```
GPIB.Write "INITiate:CONTinuous OFF"
GPIB.Write "INITiate:IMMediate:*wai"
GPIB.Write "CALCulate:RDATA? A"

GPIB.Write "CALCulate:RDATA? REF"
```

**Return Type** Depends on FORM:DATA - Two numbers per data point

**Default** Not Applicable

**Notes:**

Generally when you query the analyzer for data, you expect that the number of data values returned will be consistent with the number of points in the sweep.

However, if you query receiver data while the instrument is sweeping, the returned values may contain zeros. For example, if your request for receiver data is handled on the 45th point of a 201 point sweep, the first 45 values will be valid data, and the remainder will contain complex zero.

This can be avoided by synchronizing this request with the end of a sweep or putting the channel in hold mode.
Learn about Unratioed Measurements
**Calculate:Smoothing Commands**

Controls point-to-point smoothing. Smoothing is a noise reduction technique that averages adjacent data points in a measurement trace. Choose the amount of smoothing by specifying either the number of points or the aperture. Smoothing is not the same as CALC:AVERage which averages each data point over a number of sweeps.

![Diagram of CALC:SMoothing](image)

Click on a blue keyword to view the command details.

**See Also**

- Example Programs
- Learn about Smoothing
- Synchronizing the PNA and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using Calc:Par:Select

**CALCulate<cnum>:SMOothing:APERture <num>**

(Read-Write) Sets the amount of smoothing as a percentage of the number of data points in the channel.

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` Percentage value. Choose any number between: 1 and 25

**Examples**

```
CALC:SMO:APER 2
calculate2:smoothing:aperture 20.7
```

**Query Syntax**

CALCulate<cnum>:SMOothing:APERture?

**Return Type** Numeric
CALCulate<cnum>:SMOothing:POINts <num>
(Read-Write) Sets the number of adjacent data points to average.

See Critical Note

Parameters
  <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
  <num> Number of points from 1 point to maximum of 25% of data points in the channel. For example: if number of points in a data trace = 401, the maximum value for points = 100. The points value is always rounded to the closest odd number.

Examples
  CALC:SMO:POIN 50
  calculate2:smoothing:points 21

Query Syntax
  CALCulate<cnum>:SMOothing:POINts?

Return Type
  Numeric

Default
  3

CALCulate<cnum>:SMOothing[:STATe] <ON | OFF>
(Read-Write) Turns data smoothing ON or OFF.

See Critical Note

Parameters
  <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
  <ON | OFF> ON (or 1) - turns smoothing ON.
              OFF (or 0) - turns smoothing OFF.

Examples
  CALC:SMO ON
  calculate2:smoothing:state off

Query Syntax
  CALCulate<cnum>:SMOothing[:STATe]?

Return Type
  Boolean (1 = ON, 0 = OFF)

Default
  OFF
**Calculate:Transform Commands**

Specifies the settings for time domain transform.

![Diagram of Calculate:Transform Commands]

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Time Domain
- Synchronizing the PNA and Controller
- SCPI Command Tree

**Critical Note:** CALCulate commands act on the selected measurement. You can select one measurement for each channel using **Calc:Par:Select**

**CALCulate<cnum>:TRANSform:COUPle:PARameters <num>**

*(Read-Write)* Specifies the time domain transform parameters to be coupled. The settings for those parameters will be copied from the selected measurement to all other measurements on the channel.

- To turn coupling ON and OFF, use **SENS:COUP:PAR**
- To specify Gating parameters to couple, use **CALC:FILT:COUP:PAR**

Learn more about **Time Domain Trace Coupling**

See Critical Note

**Parameters**

- `<cnum>` Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>` (Numeric) Parameters to couple. To specify more than one parameter, add the
numbers.

1 - Transform Stimulus (Start, Stop, Center, and Span TIME settings.)

2 - Transform State (ON / OFF)

4 - Transform Window (Kaiser Beta / Impulse Width)

8 - Transform Mode (Low Pass Impulse, Low Pass Step, Band Pass)

16 - Transform Distance Marker Units

Examples

'To couple all parameters:

CALC:TRAN:COUP:PAR 31

'To couple Stimulus and Mode:

calculate2:transform:couple:parameters 9

Query Syntax

CALCulate<cnum>:TRANsform:COUPle:PARameters?

Return Type

Numeric

Default

29 (All parameters except 2 - Transform State)

---

**CALCulate<cnum>:TRANsform:TIME:CENTer <num>**

(Read-Write) Sets the center time for time domain measurements.

**See Critical Note**

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- **<num>** Center time in seconds; any number between:
  
  ± \((\text{number of points}-1) / \text{frequency span}\)

  **Note:** This command will accept MIN or MAX instead of a numeric parameter. See **SCPI Syntax** for more information.

**Examples**

CALC:TRAN:TIME:CENT 1e-8

calculate2:transform:time:center 15 ps

**Query Syntax**

CALCulate<cnum>:TRANsform:TIME:CENTer?

**Return Type**

Numeric

**Default**

0

1964
**CALCulate\(<cnum>\):TRANsform:TIME:IMPulse:WIDTh <num>**

*(Read-Write)* Sets the impulse width for the transform window.

**See Critical Note**

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- **<num>** Impulse width in seconds; Choose any number between: 
  \[ \frac{.6}{\text{frequency span}} \text{ and } \frac{1.39}{\text{frequency span}} \]

**Examples**

- `CALC:TRANS:TIME:IMP:WIDTh 10`
- `calculate2:transform:time:impulse:width 13`

**Query Syntax**

- `CALCulate\(<cnum>\):TRANsform:TIME:IMPulse:WIDTh?`

**Return Type**

- Numeric

**Default**

- \(.98 / \text{Default Span}\)

---

**CALCulate\(<cnum>\):TRANsform:TIME:KBESsel <num>**

*(Read-Write)* Sets the parametric window for the Kaiser Bessel window.

**See Critical Note**

**Parameters**

- **<cnum>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- **<num>** Window width for Kaiser Bessel in seconds; Choose any number between: 
  \[ 0.0 \text{ and } 13.0 \]

**Examples**

- `CALC:TRANS:TIME:KBES 10`
- `calculate2:transform:time:kbessel 13`

**Query Syntax**

- `CALCulate\(<cnum>\):TRANsform:TIME:KBESsel?`

**Return Type**

- Numeric

**Default**

- \(6\)

---

**CALCulate\(<cnum>\):TRANsform:TIME:LPFREQuency**
(Write-only) Sets the start frequencies in LowPass Mode.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

Examples

CALC:TRAN:TIME:LPFR
calculate2:transform:time:lpfrequency

Query Syntax

Not applicable

Default

Not applicable

CALCulate<cnum>:TRANsform:TIME:MARKer:MODE <char>

(Read-Write) Specifies the measurement type in order to determine the correct marker distance.

- Select Auto for S-Parameter measurements.
- Select Reflection or Transmission for arbitrary ratio or unratioed measurements.

This setting affects the display of ALL markers for only the ACTIVE measurement.

Learn more about Distance Markers.

See Critical Note

Parameters

<cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.

<char> Choose from:

AUTO If the active measurement is an S-Parameter, automatically chooses reflection or transmission. If non S-Parameter measurements, reflection is chosen.

REFLECTION Displays the distance from the source to the receiver divided by two (to compensate for the return trip.)

TRANSMISSION Displays the distance from the source to the receiver.

Examples

CALC:TRAN:TIME:MARK:MODE REFL
calculate2:transform:time:marker:mode auto

Query Syntax

CALCulate<cnum>:TRANsform:TIME:MARKer:MODE?
**CALCulate<cnm>:TRANSform:TIME:MARKer:UNIT <char>**

*(Read-Write)* Specifies the unit of measure for the display of marker distance values. This setting affects the display of all markers for only the ACTIVE measurement (unless Distance Marker Units are coupled using **CALC:TRAN:COUP:PAR**).

Learn more about [Distance Markers](#).

**See Critical Note**

**Parameters**

- **<cnm>** Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnm> is set to 1.
- **<char>** Choose from:

  - METRs
  - FEET
  - INCHes

**Examples**

```
CALC:TRAN:TIME:MARK:UNIT INCH
```

```
calculate2:transform:time:marker:unit feet
```

**Query Syntax**

CALCulate<cnm>:TRANSform:TIME:MARKer:UNIT?

**Return Type** Character

**Default** Auto

---

**CALCulate<cnm>:TRANSform:TIME:SPAN <num>**
(Read-Write) Sets the span time for time domain measurements.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>`: Span time in seconds; any number between: 0 and 2* [(number of points-1) / frequency span]

**Note**: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

**Examples**

```
CALC:TRAN:TIME:SPAN 1e-8
calculate2:transform:time:span maximum
```

**Query Syntax**

CALCulate<cnum>:TRANsform:TIME:SPAN?

**Return Type**

Numeric

**Default**

20 ns

---

**CALCulate<cnum>:TRANsform:TIME:STARt <num>**

(Read-Write) Sets the start time for time domain measurements.

**See Critical Note**

**Parameters**

- `<cnum>`: Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, `<cnum>` is set to 1.
- `<num>`: Start time in seconds; any number between: ± (number of points-1) / frequency span

**Note**: This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.

**Examples**

```
CALC:TRAN:TIME:STAR 1e-8
calculate2:transform:time:start minimum
```

**Query Syntax**

CALCulate<cnum>:TRANsform:TIME:STARt?

**Return Type**

Numeric

**Default**

-10 ns
CALCulate<cnum>:TRANsform:TIME:STATe <ON | OFF>

(Read-Write) Turns the time domain transform capability ON or OFF.

See Critical Note

Note: Sweep type must be set to Linear Frequency in order to use Time Domain Transform.

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <ON|OFF> ON (or 1) - turns time domain ON.
  OFF (or 0) - turns time domain OFF.

Examples

```
CALC:TRAN:TIME:STAT ON
calculate2:transform:time:state off
```

Query Syntax

CALCulate<cnum>:TRANsform:TIME:STATe?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

CALCulate<cnum>:TRANsform:TIME:STOP <num>

(Read-Write) Sets the stop time for time domain measurements.

See Critical Note

Parameters

- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <num> Stop time in seconds; any number between:
  \[ \pm \frac{(\text{number of points}-1)}{\text{frequency span}} \]

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

```
CALC:TRAN:TIME:STOP 1e-8
calculate2:transform:time:stop maximum
```

Query Syntax

CALCulate<cnum>:TRANsform:TIME:STOP?

Return Type

Numeric

Default

10 ns
**CALCulate\(<cnum>\):TRANSform:TIME:STEP:RTIMe <num>**

*(Read-Write)* Sets the step rise time for the transform window.

### See Critical Note

**Parameters**

- \(<cnum>\) Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<cnum>\) is set to 1.
- \(<num>\) Rise time in seconds; Choose any number between: \(.45 / \text{frequency span}\) and \(1.48 / \text{frequency span}\)

### Examples

<table>
<thead>
<tr>
<th>CALC:TRAN:TIME:STEP:RTIM 1e-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>calculate2:transform:time:step:rtime 15 ps</td>
</tr>
</tbody>
</table>

### Query Syntax

CALCulate\(<cnum>\):TRANSform:TIME:STEP:RTIMe?

### Return Type

Numeric

**Default** .99 / Default Span

---

**CALCulate\(<cnum>\):TRANSform:TIME:STIMulus <char>**

*(Read-Write)* Sets the type of simulated stimulus that will be incident on the DUT.

### See Critical Note

**Parameters**

- \(<cnum>\) Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, \(<cnum>\) is set to 1.
- \(<char>\) Choose from:
  - **STEP** - simulates a step DUT stimulus
  - **IMPulse** - simulates a pulse DUT stimulus

**STEP** can ONLY be used when **CALC:TRAN:TIME:TYPE** is set to LPASs (Lowpass). (STEP cannot be used with TYPE = BPASs.)

:STIM STEP will set :TYPE to **LPASs**

:TYPE BPASs will set :STIM to **IMPulse**

### Examples

<table>
<thead>
<tr>
<th>CALC:TRAN:TIME:STIM STEP</th>
</tr>
</thead>
<tbody>
<tr>
<td>calculate2:transform:time:stimulus impulse</td>
</tr>
</tbody>
</table>

### Query Syntax

CALCulate\(<cnum>\):TRANSform:TIME:STIMulus?

### Return Type

Character
CALCulate<cnum>:TRANsform:TIME[:TYPE] <char>

(Read-Write) Sets the type of time domain measurement.

See Critical Note

Parameters
- <cnum> Channel number of the measurement. There must be a selected measurement on that channel. If unspecified, <cnum> is set to 1.
- <char> Type of measurement. Choose from:
  - LPASs - Lowpass; Must also send CALC:TRAN:TIME:LPFRequency before calibrating.
  - BPASs - Bandpass;

BPASs can only be used when CALC:TRAN:TIME:STIM is set to IMPulse.
(BPASs cannot be used with :STIM = STEP)

:STIM STEP will set :TYPE to LPASs

:TYPE BPASs will set :STIM to IMPulse

Examples
- CALC:TRAN:TIME LPAS
- calculate2:transform:time:type bpas

Query Syntax
- CALCulate<cnum>:TRANsform:TIME[:TYPE]?

Return Type
- Character

Default
- BPAS
Control Commands

Specifies the settings to remotely control the rear panel connectors and ECAL Module state.

| CONTrol
| AUXiliary - More Commands
| CHANnel:INTerface:CONTrol:
  | CONFig:RECall
  | [STATe]
| ECAL:MODule:
  | PATH:
    | COUNt?
    | STATe
    | STATe
| EXTernal:TESTset - More Commands
| HANDler - More Commands
| NOISe:SOURce[:STATe]
| SIGNal:
  | TRIGger
    | ATBA
    | OUTP

Click on a blue keyword to view the command details.

Red command is superseded.

See Also

- Example Programs
- Synchronizing the PNA and Controller
See a pinout and detailed description of the rear panel connectors:

- Auxilliary IO connector
- External Test Set IO connector
- Material Handler IO connector

**CONTrol:CHANnel:INTerface:CONTrol:CONFig:RECall <string>**
(Write-only) Recalls an Interface Control configuration file. [Learn more about Interface Control.](#)

**Parameters**

- `<string>` File name and extension (.xml) of the configuration file to recall. Files are typically stored in the default folder "C:/Program Files/Agilent/Network Analyzer/Documents". To recall from a different folder, specify the full path name.

**Examples**

```
CONT:CHAN:INT:CONT:CONF:REC 'MyConfigFile.xml'
```

**Query Syntax** Not Applicable

**Default** Not Applicable

**CONTrol:CHANnel:INTerface:CONTrol[:STATe] <bool>**
(Read-Write) Enables and disables ALL Interface Control settings. To send data, the individual interfaces must also be enabled. [Learn more about Interface Control.](#)

**Parameters**

- `<bool>` Boolean

**False** (0) - Interface Control is disabled; NO control data is sent.

**True** (1) - Interface Control is enabled.

**Examples**

```
CONT:CHAN:INT:CONT 1
control:channel:interface:control:state 0
```

**Query Syntax** CONTrol:CHANnel:INTerface:CONTrol[:STATe]?

**Return Type** Boolean
**CONTrol:ECAL:MODule[num]:PATH:COUNt? <name>**

(Read-only) Returns the number of unique states that exist for the specified path name on the selected ECal module.

This command performs exactly the same function as **SENS:CORR:CKIT:ECAL:PATH:COUNt?**

Use the **CONT:ECAL:MOD:PATH:STAT** command to set the module into one of those states.

Use **SENS:CORR:CKIT:ECAL:PATH:DATA?** to read the data for a state.

**Parameters**

[num] Optional argument. USB number of the ECal module. If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use **SENS:CORR:CKIT:ECAL:LIST?** to determine how many, and **SENS:CORR:CKIT:ECAL:INF?** to verify their identities.

<name> Name of the path for which to read number of states. Choose from:

Reflection paths

- A
- B
- C (4-port modules)
- D (4-port modules)

Transmission paths

- AB
- AC (4-port modules)
- AD (4-port modules)
- BC (4-port modules)
- BD (4-port modules)
- CD (4-port modules)

Note: For each transmission path, the first of the available states is the through state, the second is the confidence (attenuator) state.
**Examples**

- `CONT:ECAL:MOD:PATH:COUNT? A`
- `control:ecal:module2:path:count? cd`

**Return Type**

- Integer

**Default**

- Not Applicable

---

**CONTrol:ECAL:MODule[num]:PATH:STATe <path>, <stateNum>**

(Write-only) Sets the internal state of the selected ECAL module. This command supersedes **CONT:ECAL:MOD:STAT**.

- Use `CONT:ECAL:MOD:PATH:COUN?` to read the number of unique states that exist for the specified path name on the module.

- Use `SENS:CORR:CKIT:ECAL:PATH:DATA?` to read the data for a state (from the module memory) corresponding to the stimulus values of a channel.

**Parameters**

- `[num]` Optional argument. USB number of the ECal module. If unspecified (only one ECal module is connected to the USB), `<num>` is set to 1. If two or more modules are connected, use `SENS:CORR:CKIT:ECAL:LIST?` to determine how many, and `SENS:CORR:CKIT:ECAL:INF?` to verify their identities.

- `<path>` Path name for which to set a state.

**Note:** The impedance paths are not independent. For example, changing the impedance presented on path A will cause a change to the impedance on path B.

Choose from:

**Reflection paths**

- A
- B
- C (4-port modules)
- D (4-port modules)

**Transmission paths**

- AB
- AC (4-port modules)
- AD (4-port modules)
- BC (4-port modules)
- BD (4-port modules)
- CD (4-port modules)

<stateNum>  Number of the state to set. Refer to the following table to associate the <stateNum> with a state in your ECal module.

In addition, **CONT:ECAL:MOD:PATH:COUNt?** returns the number of states in the specified ECal module.

<table>
<thead>
<tr>
<th>&lt;stateNum&gt;</th>
<th>N4432A and N4433A States</th>
<th>N4431A States</th>
<th>N469x States**</th>
<th>8509x States</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>One-Port Reflection States</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Open</td>
<td>Open</td>
<td>Impedance 1</td>
<td>Open</td>
</tr>
<tr>
<td>2</td>
<td>Short</td>
<td>Short</td>
<td>Impedance 2</td>
<td>Short</td>
</tr>
<tr>
<td>3</td>
<td>Impedance 1</td>
<td>Impedance 1</td>
<td>Impedance 3</td>
<td>Impedance 1</td>
</tr>
<tr>
<td>4</td>
<td>Impedance 2</td>
<td>Impedance 2</td>
<td>Impedance 4</td>
<td>Impedance 2</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>Impedance 5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td>Impedance 6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td>Impedance 7</td>
<td></td>
</tr>
<tr>
<td><strong>Two-Port Transmission States</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Thru</td>
<td>Thru</td>
<td>Thru</td>
<td>Thru</td>
</tr>
<tr>
<td>2</td>
<td>Confidence</td>
<td>Confidence</td>
<td>Confidence</td>
<td>Confidence</td>
</tr>
</tbody>
</table>

** The following modules have only FOUR Impedance states (1, 2, 3, 4): N4690B, N4691B, N4692A, N4696B.

** Examples **
- **CONT:ECAL:MOD:PATH:STATE A, 5**
- **control:ecal:module2:state BC, 1**

** Query Syntax **  Not Applicable
**Default** Not Applicable

**CONTrol:ECAL:MODule[num]:STATe <value> Superseded**

This command is replaced with **CONT:ECAL:MOD:PATH:STATe**.

*Write-only* Sets the internal state of the selected ECAL module.

**Parameters**

- **[num]** Optional argument. USB number of the ECal module. If unspecified (only one ECAl module is connected to the USB), <num> is set to 1. If two or more modules are connected, use **SENS:CORR:COLL:CKIT:INF?** to verify their identity.

- **<value>** Integer code for switching the module. The following are codes for Agilent ECAl modules.

### Agilent 8509x Modules

<table>
<thead>
<tr>
<th>State</th>
<th>Port A</th>
<th>Port B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Short</td>
<td>43</td>
<td>43</td>
</tr>
<tr>
<td>Load</td>
<td>33</td>
<td>33</td>
</tr>
<tr>
<td>Mismatch</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>Thru</td>
<td></td>
<td>84</td>
</tr>
<tr>
<td>Confidence</td>
<td></td>
<td>20</td>
</tr>
</tbody>
</table>

### Agilent N469x Modules

<table>
<thead>
<tr>
<th>State</th>
<th>Port A</th>
<th>Port B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>36</td>
<td>33</td>
</tr>
<tr>
<td>Short</td>
<td>39</td>
<td>45</td>
</tr>
<tr>
<td>Load</td>
<td>37</td>
<td>37</td>
</tr>
<tr>
<td>Mismatch (Offset short)</td>
<td>53</td>
<td>53</td>
</tr>
<tr>
<td>Impedance 5 (Offset open)</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Impedance 6 (Offset short)</td>
<td>21</td>
<td>21</td>
</tr>
</tbody>
</table>
### Agilent N4431A Modules

<table>
<thead>
<tr>
<th>State</th>
<th>Port A</th>
<th>Port B</th>
<th>Port C</th>
<th>Port D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>-1398</td>
<td>-1384</td>
<td>-2774</td>
<td>-2654</td>
</tr>
<tr>
<td>Short</td>
<td>-1350</td>
<td>-1381</td>
<td>-2582</td>
<td>-2642</td>
</tr>
<tr>
<td>Load</td>
<td>26985</td>
<td>-26986</td>
<td>-26986</td>
<td>26985</td>
</tr>
<tr>
<td>Mismatch</td>
<td>-26986</td>
<td>26985</td>
<td>26985</td>
<td>-26986</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Path</th>
<th>Thru</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB Path</td>
<td>-2590</td>
<td>-598</td>
</tr>
<tr>
<td>AC Path</td>
<td>-4011</td>
<td>85</td>
</tr>
<tr>
<td>AD Path</td>
<td>-2517</td>
<td>16042</td>
</tr>
<tr>
<td>BC Path</td>
<td>-1650</td>
<td>-598</td>
</tr>
<tr>
<td>BD Path</td>
<td>-4011</td>
<td>85</td>
</tr>
<tr>
<td>CD Path</td>
<td>-1352</td>
<td>16042</td>
</tr>
</tbody>
</table>

### Agilent N4432A and N4433A Modules

<table>
<thead>
<tr>
<th>State</th>
<th>Port A</th>
<th>Port B</th>
<th>Port C</th>
<th>Port D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>-6971</td>
<td>-11835</td>
<td>-14895</td>
<td>-14876</td>
</tr>
<tr>
<td>Short</td>
<td>-14395</td>
<td>-12859</td>
<td>-14899</td>
<td>-14905</td>
</tr>
<tr>
<td>Load</td>
<td>-14907</td>
<td>-14907</td>
<td>-14907</td>
<td>-14907</td>
</tr>
<tr>
<td>Offset Short</td>
<td>-9787</td>
<td>-6459</td>
<td>-14874</td>
<td>-14887</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Path</th>
<th>Thru</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB Path</td>
<td>13765</td>
<td>30069</td>
</tr>
<tr>
<td>AC Path</td>
<td>-10519</td>
<td>-2327</td>
</tr>
<tr>
<td>AD Path</td>
<td>-10538</td>
<td>-2346</td>
</tr>
<tr>
<td>BC Path</td>
<td>-5655</td>
<td>-1559</td>
</tr>
<tr>
<td>BD Path</td>
<td>-5674</td>
<td>-1578</td>
</tr>
<tr>
<td>CD Path</td>
<td>-15051</td>
<td>30069</td>
</tr>
</tbody>
</table>

**Examples**

- `CONT:ECAL:MOD:STAT 36`
- `control:ecal:module2:state 38`
**CONTrol:NOISE:SOURce[:STATE] <bool>**

*(Read-Write) Set and read the noise source ON | OFF state. For Noise Figure measurements.*

**Parameters**

- `<bool>` Boolean
  - **False (0)** - Noise Source OFF
  - **True (1)** - Noise Source ON

**Examples**

```
CONT:NOIS:SOUR 1
control:noise:source:state 0
```

**Query Syntax** CONTrol:NOISE:SOURce[:STATE]?

**Return Type** Boolean

**Default** False (0)

---

**CONTrol:SIGNal <conn>,<char>**

*(Read-Write) Configures external triggering in the PNA.*

- To control BNC1 and BNC2 with this command, then you **MUST** have TRIG:PREF:AIGLobal = True. [Learn more about this setting.](#)
- Trigger:Sequence:Source is automatically set to External when **CONTrol:SIGNal** is sent.
- Edge triggering is only available on some Microwave PNA models.
- For more information, see External Triggering in the PNA.

**Parameters**

- `<conn>` Rear Panel connector to send or receive trigger signals. Choose from:
  - **BNC1** Trigger IN from rear-panel [Trigger IN BNC connector](#)
  - **AUXT** Trigger IN from [AUX IO connector Pin 19](#)

**Note:** Only one of the input connectors is active at a time. When a command is sent to one, the PNA automatically makes the other INACTIVE.
BNC2 Trigger OUT to rear-panel Trigger OUT BNC connector.

MATHtrigger - Trigger IN from rear-panel Material Handler connector Pin 18

<char> INACTIVE - Disables the specified connector <conn>.

Choose from ONLY the following when <conn> is set to BNC1 or AUXT or MATHtrigger:

TIENEGATIVE - (Trigger In Edge Negative) - Triggers the PNA when receiving a negative going signal

TIEPOSITIVE - (Trigger In Edge Positive) - Triggers the PNA when receiving a positive going signal

TILLOW - (Trigger In Level Low) - Triggers the PNA when receiving a low level signal

TILHIGH - (Trigger In Level High) - Triggers the PNA when receiving a High-level signal

Choose from ONLY the following when <conn> is set to BNC2:

Use CONTrol:SIGNal:TRIGger:OUTP to enable the BNC2 output.

The following selections send a positive or negative pulse before or after each trigger acquisition. This normally occurs each sweep unless a channel is in point trigger mode.

TOPPAFTER - (Trigger Out Pulse Positive After) - Sends a POSITIVE going TTL pulse at the END of each trigger acquisition.

TOPPBEFORE - (Trigger Out Pulse Positive Before) - Sends a POSITIVE going TTL pulse at the START of each trigger acquisition.

TOPNAFTER - (Trigger Out Pulse Negative After) - Sends a NEGATIVE going TTL pulse at the END of each trigger acquisition.

TOPNBEFORE - (Trigger Out Pulse Negative Before) - Sends a NEGATIVE going TTL pulse at the START of each trigger acquisition.
CONT:SIGN BNC1, TIENEGATIVE
control:signal bnc2, toppbefore

Examples

Query Syntax
CONTrol:SIGNal? <conn>

In addition to the arguments listed above, the following is also a possible returned value:

NAVAILABLE - This feature is not available on this PNA

Return Type Character

Default At Preset:

BNC1 = INACTIVE
BNC2 = INACTIVE
AUXT = TILHIGH

When Output is enabled:

BNC1 = INACTIVE
BNC2 = TOPPAFTER
AUXT = TILHIGH

CONTrol:SIGNal:TRIGger:ATBA <bool>

(Read-Write) Accept Trigger Before Armed Determines what happens to an EDGE trigger signal if it occurs before the PNA is ready to be triggered. (LEVEL trigger signals are always ignored.) For more information, see External triggering.

Parameters

<bool> Boolean

False (0) - A trigger signal is ignored if it occurs before the PNA is ready to be triggered.

True (1) - A trigger signal is remembered and then used when the PNA becomes armed (ready to be triggered). The PNA remembers only one trigger signal.

Examples
CONT:SIGN:TRIG:ATBA 0
control:signal:trigger:atba true

Query Syntax
CONTrol:SIGNal:TRIGger:ATBA?

Return Type Boolean

Default False
**Contol:Signal:Trigger:Outp <bool>**

(Read-Write) **Output Enabled**  The PNA can be enabled to send trigger signals out the rear-panel **Trigger OUT BNC** connector. Use **Contol:Signal** to configure for output triggers.

For more information, see [External triggering](#).

**Parameters**

- `<bool>`  Boolean

  - False (0) - PNA does NOT output trigger signals.
  - True (1) - PNA DOES output trigger signals.

**Examples**

- `Contol:Signal:Trigger:Outp 1`
- `control:signal:trigger:outp false`

**Query Syntax**  `Contol:Signal:Trigger:Outp?`

**Return Type**  Boolean

**Default**  False

---

Last Modified:

- **25-Feb-2008**  Clarified **Cont:SIGN**
- **30-Jan-2008**  Added **ECal states note**
- **22-Aug-2007**  Added **noise command**
- **18-Jan-2007**  Fixed **count? example**
Display Commands

Controls the settings of the front panel screen.
DISplay:ANNotation:FREQuency[:STATe] <ON | OFF>

(Read-Write) Turns frequency information on the display title bar ON or OFF for all windows.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ON</td>
<td>OFF&gt;</td>
</tr>
</tbody>
</table>

Examples

```
DISP:ANN:FREQ ON
display:annotation:frequency:state off
```

Query Syntax

DISPlay:ANNotation:FREQuency[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON (1)

DISPlay:ANNotation:MESSage:STATe <ON | OFF>

(Read-Write) Enables and disables error pop-up messages on the display.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ON</td>
<td>OFF&gt;</td>
</tr>
</tbody>
</table>

Examples

```
DISP:ANN:MESS:STAT ON
display:annotation:message:state off
```

Query Syntax

DISPlay:ANNotation:MESSage:STATe?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON (1)
(Read-Write) Turns the status bar at the bottom of the screen ON or OFF. The status bar displays information for the active window.

**Parameters**

<ON | OFF>  
- **ON** (or 1) - turns status bar ON.  
- **OFF** (or 0) - turns status bar OFF.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:ANN:STAT ON</td>
<td>display:annotation:status on</td>
</tr>
<tr>
<td>display:annotation:status off</td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

DISPlay:ANNotation:STATus?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

Last state that was set

---

DISPlay:ARRange <char>

(Write-only) Places EXISTING measurements into pre-configured window arrangements. Overlay, Stack(2), Split(3), and Quad(4) creates new windows. To learn more, see Arrange Existing Measurements.

**Parameters**

<char>  
Window arrangement. Choose from:  

- TILE - tiles existing windows  
- CASCade - overlaps existing windows  
- OVERlay - all traces placed in 1 window  
- STACk - 2 windows  
- SPLit - 3 windows  
- QUAD - 4 windows

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:ARR CASC</td>
<td>display:arrange cascade</td>
</tr>
</tbody>
</table>

**Query Syntax**

Not Applicable

**Default**

TILE

---

DISPlay:CATalog?
(Read-only) Returns the existing Window numbers.

**Return Type** String of Character values, separated by commas

**Example**
Two windows with numbers 1 and 2 returns: "1,2"

**Default** Not applicable

---

DISPlay:ENABLE <ON | OFF>

(Read-Write) Specifies whether to disable or enable all analyzer display information in all windows in the analyzer application. Marker data is not updated. More CPU time is spent making measurements instead of updating the display.

**Parameters**

<ON | OFF>  
- **ON** (or 1) - turns the display ON.  
- **OFF** (or 0) - turns the display OFF.

**Examples**

DISP:ENAB ON  
**display:enable on**

**Query Syntax**  
DISPlay:ENABle?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** ON

---

DISPlay:FSIGn <ON | OFF>

(Read-Write) Shows or hides the window which displays global pass/fail results.

**Parameters**

<ON | OFF>  
- **ON** (or 1) - displays the pass/fail dialog  
- **OFF** (or 0) - hides the pass/fail dialog

**Examples**

DISP:FSIG ON  
**display:fsign on**

**Query Syntax**  
DISPlay:FSIGn?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF

---

DISPlay[:TILE] - Superseded
This command is replaced by DISP:ARRange

(Write-only) Tiles the windows on the screen.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP display:tile</td>
<td></td>
</tr>
</tbody>
</table>

Default Not Applicable

DISPlay:WINDow<wnum>:ANNotation:MARKer:SINGle[:STATe] <bool>

(Read-Write) Either shows marker readout of only the active trace or other traces simultaneously.

See other SCPI Marker commands. Learn more about Marker readout.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;wnum&gt;</td>
<td>Any existing window number. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;bool&gt;</td>
<td>ON (or 1) - Shows the readout of only the active marker for each trace. OFF (or 0) - Shows up to 5 marker readouts per trace, up to 20 total readouts.</td>
</tr>
</tbody>
</table>

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:WIND:ANN:MARK:SALE ON</td>
<td></td>
</tr>
<tr>
<td>display:window:annotation:marker:single off</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax

DISPlay:WINDow:ANNotation:MARKer:SINGle?

Return Type

Boolean (1 = ON, 0 = OFF)

Default OFF

DISPlay:WINDow<wnum>:ANNotation:MARKer:SIZE <char>

(Read-Write) Specifies the size of the marker readout text. See other SCPI Marker commands. Learn more about Marker readout.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;wnum&gt;</td>
<td>Any existing window number. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;char&gt;</td>
<td>Readout text size. Choose from: NORMal</td>
</tr>
</tbody>
</table>

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:WIND:ANN:MARK:SIZE LARG</td>
<td></td>
</tr>
<tr>
<td>display:window:annotation:marker:size normal</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax

DISPlay:WINDow:ANNotation:MARKer:SIZE?

Return Type

Character

Default NORMal
DISPlay:WiNDow<wnum>:ANNotation:MARKer:STATe <ON | OFF>

(Read-Write) Specifies whether to show or hide the Marker readout (when markers are ON) on the selected window. See other SCPI Marker commands. Learn more about Marker readout.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<ON | OFF>` ON (or 1) - turns marker readout ON.
  OFF (or 0) - turns marker readout OFF.

**Examples**

```
DISP:WIND:ANN:MARK:STAT ON
display:window:annotation:marker:state off
```

**Query Syntax**

DISPlay:WINDow:ANNotation:MARKer:STATe?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default** ON

DISPlay:WiNDow<wnum>:ANNotation:TRACe:STATe <ON | OFF>

(Read-Write) Specifies whether to show or hide the Trace Status buttons on the left of the display.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<ON | OFF>` ON (or 1) - turns the buttons ON.
  OFF (or 0) - turns the buttons OFF.

**Examples**

```
DISP:WIND:ANN:TRAC:STAT ON
display:window:annotation:trace:state off
```

**Query Syntax**

DISPlay:WINDow:ANNotation:TRACe:STATe?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default** ON

DISPlay:WiNDow<wnum>:CATalog?
(Read-only) Returns the trace numbers for the specified window.

**Parameters**

<wnum> Any existing window number. If unspecified, value is set to 1.

**Example**

Window 1 with four traces:
`DISPlay:WINDow1:CATalog?`
Returns:
"1,2,3,4"

**Return Type** String of Character values separated by commas

**Default** Not applicable

---

**DISPlay:WINDow<wnum>:ENABle <ON | OFF>**

(Read-Write) Specifies whether to disable or enable all analyzer display information in the specified window. Marker data is not updated. More CPU time is spent making measurements instead of updating the display.

**Parameters**

<wnum> Any existing window number. If unspecified, value is set to 1.

<ON | OFF> ON (or 1) - turns the display ON.
OFF (or 0) - turns the display OFF.

**Examples**

`DISP:WIND:ENABle ON`
`display:window1:enable off`

**Query Syntax** `DISPlay:WINDow<wnum>:ENABle?`

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** ON

---

**DISPlay:WINDow<wnum>:SIZE <char>**
(Read-Write) Sets or returns the window setting of Maximized, Minimized, or Normal. To arrange all of the windows, use DISP:ARR.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1
- `<char>` Window size. Choose from:
  - MIN
  - MAX
  - NORM

**Examples**

- DISP:WIND:SIZE MAX
- display:window:size norm

**Query Syntax**

- DISPLAY:WINDow:SIZE?

**Default** Not Applicable

---

**DISPLAY:WINDow<wnum>[::STATe] <ON | OFF>**

(Read-Write) Write to create or delete a window on the screen or Read whether a window is present.

**Parameters**

- `<wnum>` Window number to create; choose any integer between 1 and the maximum number of windows allowed in the PNA.
- `<ON | OFF>`
  - ON (or 1) - The window `<wnum>` is created.
  - OFF (or 0) - The window `<wnum>` is deleted.

**Examples**

- DISP:WIND ON
- display:window2:state off

**Query Syntax**

- DISPLAY:WINDow<wnum>[::STATe]?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** Window number "1" ON

---

**DISPLAY:WINDow<wnum>:TABLe <char>**
(Read-Write) Write to show the specified table at the bottom of the analyzer screen or Read to determine what table is visible.

**Parameters**

- `<wnum>`: Any existing window number. If unspecified, value is set to 1.
- `<char>`: Table to show. Choose from:
  - ```OFF | MARKer | LIMit | SEGment```

**Examples**

```
DISP:WIND:TABLE SEGM
display:window:table off
```

**Query Syntax**

```
DISPlay:WINDow? <wnum>:TABLe?
```

**Default**

OFF

---

**DISPlay:WINDow<wnum>:TITLe:DATA <string>**

(Read-Write) Sets data in the window title area. The title is turned ON and OFF with **DISP:WIND:TITL:STAT OFF**.

**Parameters**

- `<wnum>`: Any existing window number. If unspecified, value is set to 1.
- `<string>`: Title to be displayed. Any characters, enclosed with quotes. If the title string exceeds 50 characters, an error will be generated and the title not accepted. Newer entries replace (not append) older entries.

**Examples**

```
DISP:WIND:TITL:DATA 'hello'
display:window2:title:data 'hello'
```

**Query Syntax**

```
DISPlay:WINDow<wnum>:TITLe:DATA?
```

**Return Type**

String

**Default**

NA

---

**DISPlay:WINDow<wnum>:TITLe[:STATe] <ON | OFF>**
(Read-Write) Turns display of the title string ON or OFF. When OFF, the string remains, ready to be redisplayed when turned back ON.

**Parameters**

- `<wnum>`: Any existing window number. If unspecified, value is set to 1.
- `<ON | OFF>`: ON (or 1) - turns the title string ON. 
  OFF (or 0) - turns the title string OFF.

**Examples**

```
DISP:WIND:TITL ON
Display:window1:title:state off
```

**Query Syntax**

`DISP:WIND<wnum>:TITLe[:STATe]?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

**DISPlay:WINDow<wnum>:TRACe<tnum>:DELeTe**

(Write-only) Deletes the specified trace from the specified window. The measurement parameter associated with the trace is not deleted.

**Parameters**

- `<wnum>`: Any existing window number. If unspecified, value is set to 1.
- `<tnum>`: The number of the trace to be deleted; if unspecified, value is set to 1.

**Examples**

```
DISP:WIND:DEL
display:window2:trace2:delete
```

**Query Syntax**

Not applicable

**Default**

NA

**DISPlay:WINDow<wnum>:TRACe<tnum>:FEED <name>**
(Write-only) Creates a new trace <tnum> and associates (feeds) a measurement <name> to the specified window<wnum>. This command should be executed immediately after creating a new measurement with `CALC:PAR:DEF<name>,<parameter>`.

To feed the same measurement to multiple traces, create another measurement with the same <name>,<parameter> using the CALC:PAR:DEF command. The analyzer will collect the data only once.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Trace number to be created. Choose any Integer between 1 and 8
- `<name>` Name of the measurement that was defined with `CALC:PAR:DEF<name>,<parameter>`

**Examples**

```
DISP:WIND:TRAC:FEED 'test'
display:window2:trace2:feed 'test'
```

**Query Syntax**

Not applicable

**Default** "CH1_S11"

---

**DISPlay:WINDow<wnum>:TRACe<tnum>:MEMory[:STATe] <ON | OFF>**

(Read-Write) Turns the memory trace ON or OFF.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Any existing trace number; if unspecified, value is set to 1
- `<ON | OFF>` **ON** (or 1) - turns the memory trace ON.  
  **OFF** (or 0) - turns the memory trace OFF.

**Examples**

```
DISP:WIND:TRAC:MEM ON
display:window2:trace2:memory:state off
```

**Query Syntax**

`DISPlay:WIND<wnum>:TRACe<tnum>:MEMory[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default** OFF

---

**DISPlay:WINDow<wnum>:TRACe<tnum>:SELect**
(Write-only) Activates the specified trace in the specified window for front panel use.

**Parameters**

- `<wnum>`: Any existing window number. If unspecified, value is set to 1.
- `<tnum>`: Any existing trace number; if unspecified, value is set to 1

**Examples**

```
DISP:WIND:TRAC:SEL
display:window2:trace2:select
```

**Query Syntax**

Not applicable

**Default**

NA

---

**DISPlay:WINDow<wnum>:TRACe<tnum>[:STATe] <ON | OFF>**

(Read-Write) Turns the display of the specified trace in the specified window ON or OFF. When OFF, the measurement behind the trace is still active.

**Parameters**

- `<wnum>`: Any existing window number. If unspecified, value is set to 1.
- `<tnum>`: Any existing trace number; if unspecified, value is set to 1
- `<ON | OFF>`: 
  - ON (or 1) - turns the trace ON.
  - OFF (or 0) - turns the trace OFF.

**Examples**

```
DISP:WIND:TRAC ON
display:window2:trace2:state off
```

**Query Syntax**

DISPlay:WIND<wnum>:TRACe<tnum>[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

---

**DISPlay:WINDow<wnum>:TRACe<tnum>TITLe:DATA <string>**
(Read-Write) Writes and read data to the trace title area. The trace title is embedded in the trace status field. Learn more.

Newer entries replace (not append) older entries. The title is turned ON and OFF with DISP:WIND:TRAC:TITL:STAT.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Trace number of the specified window. If unspecified, value is set to 1. The trace number displayed in trace status is the trace number of the channel; not the window. Use Display:Cat? to read the window numbers. Use Disp:Window:Cat? to read the trace numbers of the specified window.
- `<string>` Title to be displayed. Any characters (not spaces) enclosed with quotes.

**Examples**

```
DISP:WIND:TRAC:TITL:DATA 'MyNewMeas'
display:window2:trace3:title:data 'hello'
```

**Query Syntax**

```
DISPlay:WINDow<wnum>:TRACe<tnum>TITLe:DATA?
```

**Return Type** String

**Default** Not Applicable

---

**DISPlay:WINDow<wnum>:TRACe<tnum>TITLe:STATe <bool>**

(Read-Write) Turns display of the Trace Title ON or OFF. When turned OFF, the previous trace title returns. Set a new trace title using DISP:WIND:TRAC:TITL:DATA

Learn more about Trace Titles

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1
- `<tnum>` Trace number of the specified window. If unspecified, value is set to 1. The trace number displayed in trace status is the trace number of the channel; not the window. Use Display:Cat? to read the window numbers. Use Disp:Window:Cat? to read the trace numbers of the specified window.
- `<bool>` ON (or 1) - turns the title ON.
  
  OFF (or 0) - turns the title OFF.

**Examples**

```
DISP:WIND:TRAC:TITL ON
Display:window2:trace3:title:state off
```

**Query Syntax**

```
DISPlay:WINDow<wnum>:TRACe<tnum>:TITLe[:STATe]?
```
**Return Type**  Boolean (1 = ON, 0 = OFF)

**Default**  ON

---

**DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:AUTO**

(Write-only) Performs an **Autoscale** on the specified trace in the specified window, providing the best fit display. Autoscale is performed only when the command is sent; it does NOT keep the trace autoscaled indefinitely.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Any existing trace number; if unspecified, value is set to 1

**Examples**

```
DISP:WIN:TRAC:Y:AUTO
display:window2:trace2:y:scale:auto
```

**Query Syntax**  Not applicable

**Default**  Not applicable

---

**DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:PDIVision <num>**

(Read-Write) Sets the Y axis **Per Division** value of the specified trace in the specified window.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Any existing trace number; if unspecified, value is set to 1
- `<num>` Units / division value. The range of acceptable values is dependent on format and domain.

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See **SCPI Syntax** for more information.

**Examples**

```
DISP:WIN:TRAC:Y:PDIV 1
display:window2:trace2:y:scale:pdivision maximum
```

**Query Syntax**

```
DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:PDIVision?
```

**Return Type**  Numeric

**Default**  10

---

**DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RLEVel <num>**
(Read-Write) Sets the Y axis Reference Level of the specified trace in the specified window.

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Any existing trace number; if unspecified, value is set to 1
- `<num>` Reference level value. The range of acceptable values is dependent on format and domain.

**Examples**

```plaintext
DISP:WIND:TRAC:Y:RLEV 0
display:window2:trace2:y:scale:rlevel minimum
```

**Query Syntax**

DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RLEVel?

**Return Type**

Numeric

**Default**

NA

---

**DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RPOSition <num>**

(Read-Write) Sets the Reference Position of the specified trace in the specified window

**Parameters**

- `<wnum>` Any existing window number. If unspecified, value is set to 1.
- `<tnum>` Any existing trace number; if unspecified, value is set to 1
- `<num>` Reference position on the screen measured in horizontal graticules from the bottom. The range of acceptable values is dependent on format and domain.

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```plaintext
DISP:WIND:TRAC:Y:RPOS 0
display:window2:trace2:y:rp:position maximum
```

**Query Syntax**

DISPlay:WINDow<wnum>:TRACe<tnum>:Y[:SCALe]:RPOSition?

**Return Type**

Numeric

**Default**

5

---

**DISPlay:VIStible <ON | OFF>**

1997
(Read-Write) Makes the PNA application visible or not visible. In the Not Visible state, the analyzer cycle time for making measurements, and especially data transfer, can be significantly faster because the display does not process data.

**Parameters**

<ON | OFF>

- **ON** (or 1) - PNA app is visible
- **OFF** (or 0) - PNA app is NOT visible

**Examples**

<table>
<thead>
<tr>
<th>Display:VISIBLE</th>
<th>ON</th>
<th>OFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP:VIS ON</td>
<td></td>
<td></td>
</tr>
<tr>
<td>display:visible</td>
<td></td>
<td></td>
</tr>
<tr>
<td>visible</td>
<td></td>
<td></td>
</tr>
<tr>
<td>off</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

DISPlay:VISible?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

---

Last modified:

9/12/06  Modified for number of windows.
Format Commands

Specifies the way that data will be transferred when moving large amounts of data. These commands will affect data that is transferred with the CALC:DATA and CALC:RDATA commands.

FORMat

BORDer [DATA]

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Synchronizing the PNA and Controller
- SCPI Command Tree

FORMat:BORDer <char>

(Read-Write) Set the byte order used for GPIB data transfer. Some computers read data from the analyzer in the reverse order. This command is only implemented if FORMAT:DATA is set to :REAL. If FORMAT:DATA is set to :ASCII, the swapped command is ignored.

Parameters

<char> Choose from:

- **NORMal** - Use when your controller is anything other than an IBM compatible computers
- **SWAPped** - for IBM compatible computers

Note: Use **NORMal** if you are using drivers like VISA or development environments like VEE, LabView, or T&M Tool kit.

Examples

- FORM:BORD SWAP
- format:border normal

Query Syntax

FORMATION:BORDer?

Return Type

Character
FORMat[:DATA] <char>

(Read-Write) Sets the data format for data transfers.

To transfer measurement data, use CALC:DATA.

To transfer Cal Set data, use SOUR:POWer:CORRection:COLLect:TABLE:DATA

To transfer Source Power correction data, use SOUR:POWer:CORRection:COLLect:TABLE:FREQuency

SOUR:POWer:CORRection:DATA

**Parameters**

Choose from:
- **REAL,32** - (default value for REAL) Best for transferring large amounts of measurement data.
- **REAL,64** - Slower but has more significant digits than REAL,32. Use REAL,64 if you have a computer that doesn't support REAL,32.
- **ASCii,0** - The easiest to implement, but very slow. Use if small amounts of data to transfer.

Note: The **REAL,32** and **REAL,64** arguments transfer data in block format as explained in Transferring Measurement Data.

**Examples**

```
FORM REAL,64
format:DATA ascii
```

**Query Syntax**

FORMat:DATA?

**Return Type**

Character,Character

**Default**

ASCii,0
Hardcopy Command

Learn about Printing

SCPI Command Tree

**HCOPY:FILE <filename>**

(Write-only) Saves the screen image to a file.

**Parameters**

- **<filename>** Name of the file to save the screen to. The file is saved to the current working directory unless a valid full path name is specified.
  
  Use one of the following suffixes:
  
  - .bmp - not recommended due to large file size
  
  - .jpg - not recommended due to poor quality
  
  - .png - recommended

**Examples**

- HCOPY:FILE "myFile.png"
- \`hc\copy:file "c:\data\myfile.png"

**Query Syntax**

Not Applicable

**Default**

Not Applicable

**HCOPY[:IMMediate]**

(Write-only) Prints the screen to the default printer.

**Examples**

- HCOP
  
  - hcopy:immediate

**Query Syntax**

Not applicable

**Default**

Not Applicable
**Initiate Commands**

Controls triggering signals

![Diagram of Initiate Commands]

Click on a blue keyword to view the command details.

See Also

- **Example** [Triggering the PNA](#)
- **Learn about Triggering**
- **Synchronizing the PNA and Controller**
- **SCPI Command Tree**

---

### INITiate:CONTinuous <boolean>

**(Read-Write)** Specifies whether the PNA trigger source is set to Internal (continuous) or Manual.

This command is a subset of [TRIG:SEQ:SOURce](#), which can also set the trigger source to External.

To set how a channel responds to trigger signals, use [SENS:SWE:MODE](#).

See a [map of user interface to SCPI triggering commands](#).

For more information on triggering, see the [PNA Trigger Model](#).

**See the Example program:** [Triggering the PNA using SCPI](#).

**Parameters**

- `<boolean>`
  - **ON** (or 1) - Internal (continuous) trigger.
  - **OFF** (or 0) - Manual sweep. Use [INIT:IMMediate](#) to send a trigger signal

**Examples**

```
INIT:CONT ON
initiate:continuous off
```

**Query Syntax**

INITiate:CONTinuous?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

---
**INITiate<ch>:IMMediate**

(Write-only) Stops the current sweeps and immediately sends a trigger. (Same as Trigger! on the PNA front panel).

See the Example program:  Triggering the PNA using SCPI

**Note:** An SMC Fixed Output measurement cannot be triggered using this command. For more information, see the example program.

This command requires **Trigger:Source** to be set to Manual. This causes ONE trigger signal to be SENT each time INIT:IMM is issued.

**Sens<ch>:Sweep:Mode** sets the number of trigger signals each channel will ACCEPT (Continuous - unlimited, Groups - a specific number, or HOLD - none.)

**To trigger ALL channels in turn:**

Set ALL channels to Sens<ch>:Sweep:Mode Continuous. The <ch> argument in INIT<ch>:IMM is ignored.

Then…

- TRIG:SCOP ALL triggers ALL channels (in sequence) each time Init:Imm is sent.
- TRIG:SCOP CURRent triggers ONLY the NEXT channel each time Init:Imm is sent.

**To trigger ONLY a specified channel:**

1. Set ALL channels to Sens<ch>:Sweep:Mode HOLD
2. Send TRIG:SCOP CURRent
3. Send Init<ch>:Imm where <ch> is the channel to be triggered.

**Advanced** Situations that require some channels to be in CONT and others in HOLD are rare. The following describes the behavior of the Init:Imm command in these situations:

When **Trigger:Scope** = Global:

- If the SPECIFIED <ch> channel is in hold mode, it is put in single trigger (accepts 1 trigger signal) and goes to the end of the queue of channels to be triggered. The other 'non-hold' channels are triggered. The next Init:Imm triggers the specified channel first.

  For example: ch1 is in Hold, ch2 and ch3 are in CONT and we send INIT1:IMM
On the first INIT:IMM, ch2 and ch3 is triggered.

next INIT:IMM, ch1, ch2, ch3 is triggered.

next INIT:IMM, ch2 and ch3 is triggered.

next INIT:IMM, ch1, ch2, ch3 is triggered, and so forth.

When Trigger:Scope = Channel

- Only ONE channel is triggered for each issued INIT<ch>:IMM command.
- If the specified channel is in hold, it is put in single trigger (accepts 1 trigger signal) and goes the end of the queue of channels to be triggered as in the 'Global' example.

This is one of the PNA overlapped commands. Learn more.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

Examples

- `INIT initiate2:immediate`

Query Syntax

- Not applicable

Default

- Not applicable

Last modified:

April 23, 2007  Updated Init:Imm
Memory Commands

The memory commands control saving and loading instrument states and measurement trace data to the hard drive. To read and write trace data in GPIB format, see CALC:DATA.

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Save / Recall and File Types
- Synchronizing the PNA and Controller
- SCPI Command Tree

Specifying Path Names

The MMEM commands use the following rules to specify path names:

- The default folder is "C:\Program Files\Agilent\Network Analyzer\Documents"
- You can change the active directory using MMEMory:CDIrectory.
- Specify only the file name if using the active directory.
- You can also use an absolute path name to specify the folder and file.

MMEMory:CATalog[:<char>]? [<folder>]

2005
(Read-only) Returns a comma-separated string of file names that are in the specified folder. If there are no files of the specified type, "NO CATALOG" is returned. Learn about File Types

**Parameters**

<char> The type of files to list. Choose from:

- **STATE** - Instrument states (.sta)
- **CORRection** - Calibration Data (.cal)
- **CSARchive** - Instrument state and calibration data (.csa)
- **CSTate** - Instrument state and link to Calibration data (.cst)

If unspecified then ALL file types (even unknown types) are listed.

<folder> String - Any existing folder name. See Specifying Path Names

**Examples**

```
MMEM:CAT? 'lists all files from the current folder
memory:catalog:correction? 'C:\Program Files\Agilent\Network Analyzer\Documents' 'lists .cal files from the specified folder
```

**Default** Not applicable

**MMEMory:CDIRectory <folder>**

(Read-Write) Changes the folder name.

**Parameters**

<folder> Any drive and folder name that already exists.
If the same level as "C:\Program Files\Agilent\Network Analyzer\Documents", then no punctuation is required

**MMEM:CDIR Service**

If the new folder is at a different level than "C:\Program Files\Agilent\Network Analyzer\Documents", use a slash (/) before the folder name and enclose in quotes.

```
memory:cdirectory '\automation' 'changes default directory up one level.
```

You can use an absolute path to specify the new folder.

```
memory:cdirectory 'c:\automation\service'
```

**Query Syntax** MMEMory:CDIRectory? 'Returns the current folder name
Return Type: String
Default: 'C:\Program Files\Agilent\Network Analyzer\Documents'

**MMEMory:COPY <file1>,<file2>**

(Write-only) Copies file1 to file2. Extensions must be specified.

**Parameters**
- `<file1>` String - Name of the file to be copied. See Specifying Path Names
- `<file2>` String - Name of the file to be created from file1.

**Examples**
```
MMEM:COPY 'MyFile.cst','YourFile.cst'
```

**Query Syntax** Not applicable

**Default** Not applicable

**MMEMory:DELete <file>**

(Write-only) Deletes file. Extensions must be specified.

**Parameters**
- `<file>` String - Name of the file to be deleted. See Specifying Path Names

**Examples**
```
MMEM:DEL 'MyFile.cst'
```

**Query Syntax** Not applicable

**Default** Not applicable

**MMEMory:LOAD[:<char>] <file>**

2007
(Write-only) Loads the specified file. Learn about File Types

Parameters

<char> The type of file to load. Choose from:

- **STATE** - Instrument states (.sta)
- **CORRection** - Calibration Data (.cal)
- **CSARchive** - Instrument state and calibration data (.csa)
- **CSTate** - Instrument state and link to Calibration data (.cst)
- **ENR** - Excess Noise Source data (Noise Figure App only)

If <char> is unspecified, the extension must be included in the filename.

If an extension is specified in <file> that does not agree with <char> then no action is taken.

<file> String - Name of the file to be loaded. See Specifying Path Names

Examples

**MMEM:LOAD** 'MyFile.cst'
**mmemory:load:state** 'MyInstState'

Query Syntax

Not applicable

Default

Not applicable

**MMEMory:MDIRectory** <folder>

(Write-only) Makes a folder.

Parameters

<folder> String - Name of the folder to make. See Specifying Path Names

Examples

**MMEM:MDIR** 'MyFolder'
**mmemory:mdirectory** 'c:\NewFolder'

Query Syntax

Not applicable

Default

Not applicable

**MMEMory:MOVE** <file1>,<file2>
(Write-only) Renames <file1> to <file2>. File extensions must be specified.

Parameters

<file1>  String - Name of the file to be renamed. See Specifying Path Names
<file2>  String - Name of the new file.

Examples  MMEM:MOVE 'MyFile.cst','YourFile.cst'

Query Syntax  Not applicable

Default  Not applicable

MMEMory:RDIRecoty <folder>

(Write-only) Removes the specified folder.

Parameters

<folder>  String - Name of the folder to remove. See Specifying Path Names

Examples  MMEM:RDIR 'MyFolder'

Query Syntax  Not applicable

Default  Not applicable

MMEMory:STOR[e]::<char> <file>

(Write-only) Stores the specified file (.sta, .cal, .csa, .cst, prn).

- To save snp files, use Calc:Data:SNP:PORTs:SAVE.
- To save *.cti files, use MMEM:STOR:CIT:DATA or MMEM:STOR:CIT:FORM.

Parameters

<char>  The type of file to store. Choose from:

- **STATE** - Instrument states (.sta)
- **CORREction** - Calibration Data (.cal)
- **CSARchive** - Instrument state and calibration data (.csa)
- **CSTate** - Instrument state and link to Calibration data (.cst)
- **ENR** - Excess Noise Source data (Noise Figure App only)
No <char> is specified for prn.

If <char> is unspecified, then the extension must be included in the filename.

If an extension is specified in <file> that does not agree with <char> then no action is taken.

Learn about File Types

<file> String - Name of any valid file that is not already in existence. See Specifying Path Names

Examples

MMEM:STOR:STAT 'myState'
memory:store 'c:\bin\myState.sta'

MMEM:STOR 'MyData.prn'

Query Syntax Not applicable

Default Not applicable

MMEMory:STORe:CITifile:DATA <filename>

(Write only) Saves UNFORMATTED trace data to .cti file. Learn more.

Parameters

<filename> Any drive and folder name that already exists.
If the same level as "C:\Program Files\Agilent\Network Analyzer\Documents", then no punctuation is required

MMEM:STOR:CIT:DATA MYCTIFile

If the new folder is at a different level than "C:\Program Files\Agilent\Network Analyzer\Documents", use a slash (/) before the folder name and enclose in quotes.

memory:cdirectory '\ automation' 'changes default directory up one level.'

You can use an absolute path to specify the new folder.

memory:cdirectory 'c:\automation\service'

Query Syntax Not Applicable

Default C:\Program Files\Agilent\Network Analyzer\Documents'

MMEMory:STORe:CITifile:FORMat <filename>
(Write only) Saves FORMATTED trace data to .cti file. Learn more.

**Parameters**

- `<filename>`
  - Any drive and folder name that already exists.
  - If the same level as "C:\Program Files\Agilent\Network Analyzer\Documents", then no punctuation is required

```plaintext
MMEM:STOR:CIT:DATA MYFile
```

- If the new folder is at a different level than "C:\Program Files\Agilent\Network Analyzer\Documents", use a slash (\) before the folder name and enclose in quotes.

```plaintext
mmemory:cdirectory '\automation' 'changes default directory up one level.
```

- You can use an absolute path to specify the new folder.

```plaintext
mmemory:cdirectory 'c:\automation\service'
```

**Query Syntax**

- **Not Applicable**

**Default**

- C:\Program Files\Agilent\Network Analyzer\Documents’

---

**MMEMory:STORE:TRACE:CONTents:CITifile <char>**

*(Read-Write)* Specifies the contents of subsequent citifile save statements. (See Data Define Saves)

**Parameters**

- `<char>`
  - Choose from:
    - **SING** - Single trace
    - **DISP** - All displayed traces
    - **AUTO** - All displayed traces

**Examples**

```plaintext
MMEM:STOR:TRAC:CONT:CIT SING
```

**Query Syntax**

- **MMEMory:STORe:TRACe:CONTenTs?**

**Return Type**

- Character

**Default**

- Auto

---

**MMEMory:STORe:TRACe:FORMat:CITifile <char>**
(Read-Write) Specifies the format of subsequent citifile save statements. (See Data Define Saves)

Parameters

<char> Format in which the citifile will be saved with subsequent MMEMory:STORe:CIT:FORMat statements. Choose from:

MA - Linear Magnitude / degrees

DB - Log Magnitude / degrees

RI - Real / Imaginary

AUTO - Format in which the trace is already displayed. If other than Log Mag, Linear Magnitude, or Real/Imag, then the format will be in Real/Imag.

Examples

MMEM:STOR:TRAC:FORM:CIT MA

Query Syntax

MMEMory:STORe:TRACe:FORMat:CITifile?

Return Type

Character

Default

Auto'

---

MMEMory:STORe:TRACe:FORMat:SNP <char>

(Read-Write) Specifies the format of subsequent .s1p, .s2p, .s3p; s4p save statements. (See Data Define Saves).

Parameters

<char> Choose from:

MA - Linear Magnitude / degrees

DB - Log Magnitude / degrees

RI - Real / Imaginary

AUTO - data is output in currently selected trace format. If other than LogMag, LinMag, or Real/Imag, then output is in Real/Imag.

Examples

MMEM:STOR:TRAC:FORM:SNP MA

Query Syntax

MMEMory:STORe:TRACe:FORMat:SNP?

Return Type

Character

Default

Auto’
**MMEMory:TRANsfer <fileName>,<dataBlock>**

*(Read-Write)* Transfers data between the PNA and an external controller. Other MMEM commands transfer data between the PNA application and the PNA hard drive. If `<fileName>` already exists, it will be overwritten. The file must be no larger than 20MB.

To read **trace data** from the PNA in block format, use **CALC:DATA**.

**Parameters**

- `<fileName>`: String - File name. See [Specifying Path Names](#).
- `<dataBlock>`: Block Data - The contents of the file.

The data block is a block of binary data. Use the following syntax:

```plaintext
#<num_digits><byte count><data bytes><NL><END>
```

where:

- `<num_digits>` specifies how many digits are contained in `<byte_count>`
- `<byte_count>` specifies how many data bytes will follow in `<data bytes>`

**Example**

See example program

**Query Syntax**

**MMEMory:TRANsfer? <fileName>**

Reads block data from the specified file location.

**Default** Not applicable

---

**Last modified:**

- 30-Jul-2007 Added noise keywords to load and store.
- 9/12/06 MQ Store command has reference to PORTS Snp.
Output Commands

Controls two output functions: RF power and Noise Source.

<table>
<thead>
<tr>
<th>OUTPut:</th>
</tr>
</thead>
<tbody>
<tr>
<td>MANual:NOISe[:STATE]</td>
</tr>
<tr>
<td>[STATE]</td>
</tr>
</tbody>
</table>

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Synchronizing the PNA and Controller
- SCPI Command Tree

**OUTPut:MANual:NOISe[:STATE] <bool>**

*(Read-Write)* Sets and reads the noise source state ON or OFF. For Noise Figure measurements.

**Parameters**

- **<bool>**
  - True (1) - Noise source ON
  - False (0) - Noise source OFF

**Examples**

OUTP:MAN:NOIS 0
output:manual:noise:state 1

**Query Syntax**

OUTPut:MANual:NOISe[:STATE]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

False (0)

**OUTPut[:STATE] <ON | OFF>**
(Read-Write) Turns RF power from the source ON or OFF.

See note about source power state with instrument state save and recall.

Parameters

<ON | OFF>  
- **ON** (or 1) - turns RF power ON
- **OFF** (or 0) - turns RF power OFF

Examples

- OUTP ON
- output:state off

Query Syntax

OUTP[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

ON

Last Modified:

22-Aug-2007  Added noise command
**Route Command**

Learn about Frequency Offset

SCPI Command Tree

**ROUTe<cnum>:PATH:LOOP[<R1>] <char>**

*(Read-Write)* Throws internal switch to reference receiver when the specified channel is measured. This feature is only available on PNA models with [Option 081](#) - external reference switch. See block diagram of the reference switch.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Position of the switch. Choose from:

  **INTernal** - bypass R1 Loop. Connects the port 1 source directly to the R1 receiver.

  **EXTernal** - flow through R1 Loop. Allows direct access to the R1 receiver through the Reference 1 front-panel connectors.

**Examples**

```
ROUTe:PATH:LOOP INT
route2:path:loop:r1 external
```

**Query Syntax** ROUTe<cnum>:PATH:LOOP:R1?

**Return Type** Character

**Default** INTernal
**Sense:Average Commands**

Sets sweep-to-sweep averaging parameters. Averaging is a noise reduction technique that averages each data point over a user-specified number of sweeps. Averaging affects all of the measurements in the channel.

```
SENSe:AVERage
     CLEAR  COUNT  [STATE]
```

Click on a blue keyword to view the command details.

**See Also**

- [Example](#) using some of these commands.
- [Learn about Averaging](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

---

**SENSe<cnum>:AVERage:CLEAR**

(Write-only) Clears and restarts averaging of the measurement data. Must also set `SENSe:AVERage[:STATe] ON`.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.

**Examples**

```
SENSe:AVER:CLE
sense2:average:clear
```

**Query Syntax**

Not applicable

**Default**

Not applicable

---

**SENSe<cnum>:AVERage:COUNt <num>**
(Read-Write) Sets the number of measurement sweeps to combine for an average. Must also set SENS:AVER[:STATe] ON

Parameters
<cnun> Any existing channel number; if unspecified, value is set to 1.
<num> Number of measurement sweeps to average. Choose any number between 1 and 65536 (2^16).

Examples
SENS:AVER:COUN 999
sense2:average:count 73

Query Syntax
SENSe<cnun>:AVERage:COUNt?

Return Type
Numeric

Default
1

SENSe<cnun>:AVERage[:STATe] <ON | OFF>
(Read-Write) Turns trace averaging ON or OFF.

Parameters
<cnun> Any existing channel number; if unspecified, value is set to 1.
<ON | OFF> ON (or 1) - turns averaging ON.
OFF (or 0) - turns averaging OFF.

Examples
SENS:AVER ON
sense2:average:state off

Query Syntax
SENSe<cnun>:AVERage[:STATe]?

Return Type
Boolean (1 = ON, 0 = OFF)

Default
Off
SENSe:BANDwidth:
  RESolution <num>
  TRACk <bool>

See Also

- Example Programs
- Learn about IF Bandwidth
- Synchronizing the PNA and Controller
- SCPI Command Tree

SENSe<cnum>:BANDwidth | BWIDth[:RESolution] <num>

(Read-Write) Sets the bandwidth of the digital IF filter to be used in the measurement. (Use either Sense:Bandwidth or Sense:Bwidth)

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <num> IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. (Click to see the lists.) If an invalid number is specified, the analyzer will round up to the closest valid number.

  This parameter supports MIN and MAX as arguments. Learn more.

Examples

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:BWID 1KHZ</td>
<td></td>
</tr>
<tr>
<td>sense2:bandwidth:resolution 1000</td>
<td></td>
</tr>
</tbody>
</table>

Query Syntax

SENSe<cnum>:BANDwidth | BWIDth[:RESolution]?

Return Type

Numeric

Default

See Preset IFBW for your PNA model.

SENSe<cnum>:BANDwidth | BWIDth:TRACk <bool>
(Read-Write) Sets and returns the state of the Reduce IF BW at Low Frequencies feature. (Use either Sense:Bandwidth:Track or Sense:Bwidth:Track).

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<bool>`: Choose from:
  - **ON** or **1** - Reduce IF BW at Low Frequencies is set ON
  - **OFF** or **0** - Reduce IF BW at Low Frequencies is set OFF

Examples

<table>
<thead>
<tr>
<th>SENSE:BWID:TRAC</th>
<th>OFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>sense2:bandwidth:track</td>
<td>1</td>
</tr>
</tbody>
</table>

Query Syntax

SENSe<cnum>:BANDwidth | BWIDth:TRACk?

Return Type

Boolean

Default

ON

Last Modified:

15-Jan-2008 MIN and MAX added
Sense:Correction Commands

Performs and applies calibration and other error correction features.

To perform Guided Calibration, see Sens:Corr Coll:GUIDed.

```
SENSe:CORRection
  CCEheck
    | [ACQuire]
    | DONE
    | PAArameter
CKIT - More Commands
COLLect
  | [ACQuire]
  | APPLy
  | CKIT - More Commands
DISPlay:WINDow
  | AOFF
  | [STATe]
GUIDed - More Commands
ISOLation
  | AVERage
    | INCRement
  | ECAL
    | [STATe]
  | METHod
SAVE
SESSion - More Commands
SWEep:CHANnel
  | AOFF
  | [STATe]
  | WINDow[:STATe]
CSET - More Commands
ENR:CALibration:TABLE
  | DATA
  | ID:DATA
  | SERial:DATA
EXTension - More Commands
GCSetup:POWer
IMPedance:INPut
  | MAGNitude
```
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTerpolate</td>
<td>[STATe]</td>
</tr>
<tr>
<td>ISOLation</td>
<td>[STATe]</td>
</tr>
<tr>
<td>PREFerence</td>
<td>CSET</td>
</tr>
<tr>
<td></td>
<td>SAVE</td>
</tr>
<tr>
<td></td>
<td>SAVUser</td>
</tr>
<tr>
<td></td>
<td>ECAL</td>
</tr>
<tr>
<td></td>
<td>ORIentation</td>
</tr>
<tr>
<td></td>
<td>PMAP</td>
</tr>
<tr>
<td>SIMCal</td>
<td></td>
</tr>
<tr>
<td>TRIG</td>
<td></td>
</tr>
<tr>
<td>FREE</td>
<td></td>
</tr>
<tr>
<td>RPOWer:OFFSet</td>
<td>[AMPLitude]</td>
</tr>
<tr>
<td>RVELocity</td>
<td>COAX</td>
</tr>
<tr>
<td>SFWOard</td>
<td>[STATe]</td>
</tr>
<tr>
<td>[STATe]</td>
<td>TCOLd:USER:VALue</td>
</tr>
<tr>
<td>TSTandards</td>
<td>[STATe]</td>
</tr>
<tr>
<td>TYPE</td>
<td>CATalog?</td>
</tr>
</tbody>
</table>

Click on a blue keyword to view the command details.

Red commands are superseded.

See Also

- Example Programs
- New See Calibrating the PNA Using SCPI
- Learn about Measurement Calibration
- Synchronizing the PNA and Controller
- SCPI Command Tree

**SENSe<cnum>:CORRection:CCHeck[:ACQuire] <mod>[,char]**
(Write-only) Reads the 'confidence data' associated with the specified ECaL module and puts it into memory. The measurement is selected using SENS:CORR:CC:PAR. This command is compatible with *OPC.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<mod>` ECaL Module that contains the confidence data. Choose from:

  **ECAL1**
  
  ..through..

  **ECAL8**

- `[char]` Optional argument. Specifies which characterization within the ECaL module that the confidence data will be read from.

  **CHAR0** Factory characterization (data that was stored in the ECaL module by Agilent). Default if not specified.

  **CHAR1** User characterization #1

  **CHAR2** User characterization #2

  **CHAR3** User characterization #3

  **CHAR4** User characterization #4

  **CHAR5** User characterization #5

**Examples**

```
SENS:CORR:CCheck ECAL2
sense2:correction:ccheck:acquire ec1l, char1
```

**Query Syntax**

- **Default** Not applicable

---

**SE**NSE<cnum>:CORRection:CCheck:DONE
(Write-only) Concludes the Confidence Check and sets the ECal module back into the idle state.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

- `SENS:CORR:CCH:DONE`
- `sense2:correction:ccheck:done`

**Query Syntax**

Not applicable

**Default**

Not applicable

---

**SENSe<cnum>:CORRection:CCHeck:PARameter <Mname>**

(Read-Write) Specifies an existing measurement to be used for the Confidence Check.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<Mname>` Name of the measurement you are selecting for the confidence check. The measurement must already exist.

**Examples**

- `SENS:CORR:CCH:PAR 'TEST'`
  - 'selects the measurement "test" on channel 1 for the confidence check'
- `sense2:correction:ccheck:parameter 'test'`
  - 'selects the measurement "test" on channel 2 for the confidence check'

**Query Syntax**

`SENSe<cnum>:CORRection:CCHeck:PARameter?`

Returns the name of the selected measurement on channel `<cnum>`. 

**Return Type**

String

**Default**

Not applicable

---

**SENSe<cnum>:CORRection:COLLect[:ACQuire] <class>[,subclass]**
(Write-only) For UNGUIDED calibration, measures the specified standards from the selected calibration kit. The calibration kit is selected using the Sense:Correction:Collect:CKIT command.

For using two sets of standards, see SENS:CORR:TST.

**Note:** Before using this command you must select two items:
1. Select a calibration method using SENS:CORR:COLL:METH
2. Select a measurement using CALC:PAR:SEL. You can select one measurement for each channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<class>` **Measures the standards associated with these class labels.** Choose from:

<table>
<thead>
<tr>
<th>Label</th>
<th>Forward</th>
<th>Reverse</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAN1</td>
<td>S11A</td>
<td>S22A</td>
</tr>
<tr>
<td>STAN2</td>
<td>S11B</td>
<td>S22B</td>
</tr>
<tr>
<td>STAN3</td>
<td>S11C</td>
<td>S22C</td>
</tr>
<tr>
<td>STAN4</td>
<td>S21T</td>
<td>S12T</td>
</tr>
<tr>
<td>STAN5</td>
<td>Generic Isolation; not associated with calibration kit definition.</td>
<td></td>
</tr>
<tr>
<td>ECAL1</td>
<td>ECAL modules through</td>
<td></td>
</tr>
<tr>
<td>ECAL8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POWER</td>
<td>Take a receiver power cal sweep and turn correction ON</td>
<td></td>
</tr>
<tr>
<td>SLSET</td>
<td>Sets 'sliding load type', and increments the &quot;number of slides&quot; count. The total number of slides is critical to the correct calculation of the sliding load algorithm. See a <a href="#">sliding load cal example</a>.</td>
<td></td>
</tr>
<tr>
<td>SLDONE</td>
<td>Computes the sliding load using a circle fit algorithm.</td>
<td></td>
</tr>
<tr>
<td>[subclass]</td>
<td>Optional argument. For mechanical calibration kits, choose from the following to specify the standard to be acquired from the SENS:CORR:COLL:CKIT:ORDer list. If not specified, subclass is set to SST1.</td>
<td></td>
</tr>
</tbody>
</table>
SST1  First standard in the order list
SST2  Second standard in the order list
SST3  Third standard in the order list
SST4  Fourth standard in the order list
SST5  Fifth standard in the order list
SST6  Sixth standard in the order list
SST7  Seventh standard in the order list

If an ECAL module (1 through 8) is specified for <class>, choose one of the following for specifying which characterization within the ECal module will be used for the acquire. If not specified, the default is **CHAR0**.

**CHAR0**  Factory characterization (data that was stored in the ECal module by Agilent)
CHAR1  User characterization #1
CHAR2  User characterization #2
CHAR3  User characterization #3
CHAR4  User characterization #4
CHAR5  User characterization #5

**Examples**

```
SENS:CORR:COLL STAN1

'If SENS:CORR:COLL:CKIT:ORDer2 5,3,7 was specified, the following command measures standard 3 (the second in the order list)
sense1:correction:collect:acquire stan3,sst2

SENS:CORR:COLL ECAL4

sense2:correction:collect:acquire ecal2,char1
```

**Query Syntax**

**Default**

Not applicable

**SENSe<cnum>:CORRection:COLLect:APPLy**
(Write-only) Applies error terms to the measurement that is selected using `Calc:Par:Select`.

**Note:** Before using this command you must select a measurement using `CALC:PAR:SEL`. You can select one measurement for each channel.

**Note:** This command is only necessary if you need to modify error terms. If you do not need to modify error terms, `SENSe<nun>:CORRection:COLLect:SAVE` calculates and then automatically applies error terms after you use `SENS:CORR:COLL:ACQuire` to measure cal standards.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Example**

1. `CALCulate2:PARameter:SELect S21_2 'select the measurement to apply terms to`
2. `SENSe2:CORRection:COLLect:METHod SPARSOLT 'set type of cal method.`
3. `CALCulate2:DATA? SCORR1 'download the error term of interest`
4. `'Modify the error term here`
5. `CALCulate2:DATA SCORR1 'upload the error term of interest`
6. `SENSe2:CORRection:COLLect:APPLy 'applies the error terms to the measurement`

**Query Syntax**

- **Default** Not applicable

---

**SENSe:CORRection:COLLect:DISPlay:WINDow:AOFF**

(Write-only) Clears the flags for windows to be shown during calibrations. To flag a window to be shown see `SENS:CORR:COLL:DISP:WIND`.

**Examples**

- `SENS:CORR:COLL:DISP:WIND:AOFF`

  `sense:correction:collect:display:window:aoff`

  See an example using this command.

**Query Syntax**

- **Default** Not Applicable

---

**SENSe:CORRection:COLLect:DISPlay:WINDow<wNum>[:STATe] <bool>**
(Read-Write) Set and read the 'show' state of the window to be displayed during a calibration. Learn more.

When this command is sent, the specified window is 'flagged' to be shown during calibration. The flag is cleared when the window is closed. A Preset or Instrument State Recall also closes the window. If the same window number is reopened, this command must be sent again to show the window during a calibration. The flag is NOT saved with an instrument state.

Send this command for each additional window to show during a calibration.

**Parameters**

- `<wNum>` Window number to show during a calibration. The calibration window will also be shown with this window.

  The window must already be created.

  Use **DISPlay:CATalog?** to read all existing window numbers.

- `<bool>` Window state. Choose from:

  - **ON** (or 1) - Show the specified window during calibration.
  - **OFF** (or 0) - Do NOT show the specified window during calibration.

**Examples**

```
SENS:CORR:COLL:DISP:WIND1 1
sense:correction:collect:display:window2:state off
```

See an example using this command.

**Query Syntax**

```
SENSe:CORRection:COLLect:DISPlay:WINDow<wnum>[:STATe]?
```

**Return Type**

Boolean

**Default**

OFF

---

```
SENSe:CORRection:COLLect:ISOLation:AVERage:INCRement <num>
```
(Read-Write) Specifies amount to increment (increase) the channel averaging factor during isolation measurement of the ECal module during an unguided ECal calibration.

**Note:** if the channel currently has averaging turned OFF and <num> is greater than 1, averaging will be turned ON only during the isolation measurements and with the averaging factor equal to <num>.

**Parameters**

- **<num>**
  - Incremental Averaging factor. The maximum averaging factor is 65536 (2^16).

**Examples**

```
SENS:CORR:COLL:ISOL:AVER:INCR 16
sense:correction:collect:isolation:average:increment 0
```

**Query Syntax**

```
SENSe:CORRection:COLLect:ISOLation:AVERage:INCRement?
```

**Return Type**

Numeric

**Default**

8 - If this command is NOT sent, but ECal isolation is measured, then averaging will be turned ON with factor set to 8 during the isolation measurement.

---

**SENSe<cnum>:CORRection:COLLect:ISOLation:ECAL[:STATE] <bool>**

(Read-Write) Specifies whether or not the isolation state of the ECal module will be measured as part of an unguided ECal calibration.

An unguided calibration is performed using the SENS:CORR:COLL:METH and SENS:CORR:COLL:ACQ commands.

**Note:** The inherent isolation of the PNA is better than that attained with this command. ONLY use this command when using an external test set, and ONLY using a 8509x ECal module.

**Parameters**

- **<cnum>**
  - Any existing channel number. If unspecified, value is set to 1
- **<bool>**
  - **ON** (or 1) - isolation is measured during the unguided ECal calibration.
  - **OFF** (or 0) isolation is NOT measured during the unguided ECal calibration.

**Examples**

```
SENS1:CORR:COLL:ISOL:ECAL:STATE ON
sense2:correction:collect:isolation:ecal:state 0
```

**Query Syntax**

```
SENSe:CORRection:COLLect:ISOLation:ECAL:STATe?
```

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

2029
**SENSe<cnum>:CORRection:COLLect:METHod <char>**

*(Read-Write)* For UNGUIDED calibration, sets the calibration method (also known as 'Calibration Type' on calibration dialog box.) To select a Cal Type from a Cal Set, use `CALC:CORR:TYPE`.

**Note:** Before using this command you must select a measurement using `CALC:PAR:SEL`. You can select one measurement for each channel.

### Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<char>`: Choose from:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>No Cal method</td>
</tr>
<tr>
<td>REFL1OPEN</td>
<td>Response Open</td>
</tr>
<tr>
<td>REFL1SHORT</td>
<td>or Response Short</td>
</tr>
<tr>
<td>REFL1</td>
<td></td>
</tr>
<tr>
<td>REFL3</td>
<td>Full 1 port</td>
</tr>
<tr>
<td>RPOWER</td>
<td>Receiver Power Cal</td>
</tr>
<tr>
<td>TRAN1</td>
<td>Response Thru</td>
</tr>
<tr>
<td>TRAN2</td>
<td>Response Thru and Isolation</td>
</tr>
<tr>
<td>SPARSOLT</td>
<td>Full SOLT 2 port</td>
</tr>
<tr>
<td>SPARTRL</td>
<td>TRL Cal <em>(not available on all PNAs.)</em></td>
</tr>
</tbody>
</table>

### Examples

- `SENSe:CORRection:COLLect:METH REFL1`  
- `sense2:correction:collect:meth sparsolt`

### Query Syntax

`SENSe<cnum>:CORRection:COLLect:METHod?`

**Return Type**

- **Character**

**Default**

- Not Applicable

---

**SENSe<cnum>:CORRection:COLLect:SAVE**

---

2030
(Write-only) For UNGUIDED calibrations ONLY. This command does the following:

- calculates the error terms using the selected :METHod
- applies the error terms to the selected measurement (turns error correction ON.)
- saves the calibration error-terms to the channels Cal Register or a User Cal Set.

The Cal Register or User Cal Set is determined by the setting of the SENS:CORR:PREFerence:CSET:SAVE command.

Do NOT use this command during an ECAL. When performing an ECAL calibration using SENS:CORR:COLL:ACQuire, this SAVE operation is performed automatically before the completion of a successful ACQuire.

Before using this command you must select a measurement using CALC:PAR:SEL. You can select one measurement for each channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```
SENS:CORR:COLL:SAVE
sense2:correction:collect:save
```

**Query Syntax**

Not applicable

**Default**

Not applicable

---

**SENSe:CORRection:COLLect:SWEep:CHANnel:AOFF**

(Write-only) Clears ALL flags for channels to sweep during calibration. To flag a channel, see SENS:CORR:COLL:SWE:CHAN.

**Examples**

```
SENS:CORR:COLL:SWE:CHAN:AOFF
sense:correction:collect:sweep:channel:aoff
```

See an example using this command.

**Default**

Not applicable

---

**SENSe<cnum>:CORRection:COLLect:SWEep:CHANnel<cnum2>[:STATe] <bool>**
Specifies the channel to sweep during a Calibration.

When this command is sent, the <cnum2> channel is 'flagged' to be swept during calibration. The flag is cleared when the channel is deleted or if the Measurement Class is changed. If the same channel number is recreated, this command must be sent again to sweep the channel during a calibration. The flag is NOT saved with an instrument state.

A Preset or Instrument State Recall deletes the channel.

**Parameters**

- `<cnum>`: The channel to be calibrated. If unspecified, value is set to 1.
- `<cnum2>`: The channel to sweep when waiting to measure a standard.
  
  This channel must already exist. If this channel is in continuous sweep mode, it must have the same attenuator settings and path configuration (PNA-X only).
- `<bool>`: Channel sweep state. Choose from:
  
  **ON** (or 1) - Sweep the channel during calibration.
  
  **OFF** (or 0) - Do NOT sweep the channel during calibration.

**Examples**

```plaintext
SENS:CORR:COLL:SWE:CHAN2 1
sense2:correction:collect:sweep:channel3:state off
```

See an example using this command.

**Query Syntax**

`SENSe<cnum>:CORRection:COLLect:SWEep:CHANel<cnum2>:[STATe]?

**Return Type**

Boolean

**Default**

OFF

---

`SENSe:CORRection:ENR:CALibration:TABLE:DATA <freq, value, freq, value...>`
(Read-Write) Set and read the ENR calibration data. All of the frequency and ENR data must be sent at the same time. Use **MMEM:LOAD** and **MMEM:SAVE** to load and save ENR table data from disk. Learn more about **Noise Source ENR files**.

**Parameters**

- `<freq, value>`  (Numeric) ENR data. Frequency value in Hz followed by a ENR noise value in dB. Enter as many pairs as necessary.

**Examples**

```plaintext
SENS:CORR:ENR:CAL:TABL:DATA 1.0E9,14.37,2.5E9,15.28
sense:correction:enr:calibration:table:-data 1.0E9,14.37,2.5E9,15.28
```

**Query Syntax**

```
SENS:CORRection:ENR:CALibration:TABLe:DATA?
```

**Return Type**

Comma separated numeric values

**Default**

Not Applicable

---

**SENSe:CORRection:ENR:CALibration:TABLE:ID:DATA <id>**

(Read-Write) Sets and returns ID of ENR table. While this is for informational purposes only, it can be used to record the model of the noise source. Learn more about **ENR files**.

**Parameters**

- `<id>`  (String) Identifier for the ENR table.

**Examples**

```plaintext
SENS:CORR:ENR:CAL:TABL:ID:DATA "346C"
sense:correction:enr:calibration:table:id:data "ENR Table"
```

**Query Syntax**

```
SENS:CORRection:ENR:CALibration:TABLE:ID:DATA?
```

**Return Type**

String

**Default**

Not Applicable

---

**SENSe:CORRection:ENR:CALibration:TABLe:SERial:DATA <sn>**
**SENSe:CORRection:ENR:CALibration:TABLe:SERial:DATA**

(Read-Write) Sets and returns the serial number of noise source. This is for informational purposes only to identify the specific noise source for which the data pertains. [Learn more about ENR files.]

**Parameters**

- `<sn>` Serial number of the noise source for which the data applies, enclosed in quotes.

**Examples**

```plaintext
SENS:CORR:ENR:CAL:TABLE:SER:DATA "ABCD1234"
sense:correction:enr:calibration:table:serial:data "ABCD1234"
```

**Query Syntax**

SENSe:CORRection:ENR:CALibration:TABLe:SERial:DATA?

**Return Type** String

**Default** Not Applicable

**SENSe<ch>:CORRection:GCSetup:POWer <num>**

(Read-Write) Set and read the power level at which to perform the Source Power Cal portion of a Gain Compression (Opt 086) Calibration. [Learn more about this setting.]

**Parameters**

- `<num>` Power level in dB. Choose a value from +30 to (-30).

**Examples**

```plaintext
SENS:CORR:GCS:POW 0
sense:correction:gcsetup:power 5
```

**Query Syntax**

SENSe:CORRection:GCSetup:POWer?

**Return Type** Numeric

**Default** 0

**SENSe:CORRection:IMPedance:INPut:MAGNitude <num>**

(Read-Write) Sets and returns the system impedance value for the analyzer.

**Parameters**

- `<num>` System Impedance value in ohms. Choose any number between 0 and 1000 ohms.

**Examples**

```plaintext
SENS:CORR:IMP:INP:MAGN 75
sense:correction:impedance:input:magnitude 50.5
```

**Query Syntax**

SENSe:CORRection:IMPedance:INPut:MAGNitude?

**Return Type** Numeric

**Default** 50
SENSe<cnum>:CORRection:INTerpolate[:STATE] <ON | OFF>
(Read-Write) Turns correction interpolation ON or OFF.

**Note:** Before using this command you must select a measurement using CALC:PAR:SEL. You can select one measurement for each channel.

**Parameters**
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <ON | OFF> ON (or 1) - turns interpolation ON.
  OFF (or 0) - turns interpolation OFF.

**Examples**
SENSe:CORR:INT ON
SENSe:correction:interpolate:state off

**Query Syntax**
SENSe<cnum>:CORRection:INTerpolate[:STATE]?

**Return Type**
Boolean (1 = ON, 0 = OFF)

**Default**
ON

SENSe<cnum>:CORRection:ISOLation[:STATE] <ON | OFF> OBSOLETE

This command no longer works beginning in the PNA 5.2 release. To perform isolation as part of an unguided calibration, you must explicitly measure the isolation standard using SENS:CORR:COLL:ACQ Stan5. See Example program.

To measure isolation as part of an ECal, use SENS:CORR:COLL:ISOL:ECAL.

(Read-Write) Turns isolation cal ON or OFF during Full 2-port calibration. If this command is not sent, the default state is to disable Isolation.

**Parameters**
- <cnum> Any existing channel number. If unspecified, value is set to 1
- <ON | OFF> ON (or 1) - turns isolation ON.
  OFF (or 0) - turns isolation OFF.

**Examples**
SENSe:CORR:ISOL ON
SENSe:correction:isolation:state off

**Query Syntax**
SENSe<cnum>:CORRection:ISOLation[:STATE]?

**Return Type**
Boolean (1 = ON, 0 = OFF)

**Default**
OFF - (Isolation disabled)
SENSe:CORRection:PREFerence:CSET:SAVE <char>

Important Notes:

- This command replaces SENS:CORR:PREF:CSET:SAVU

- With 6.0 we implemented a change that defaults to saving completed calibrations to Cal Registers instead of User Cal Sets. To revert to the old behavior, send this command with the USER argument.

(Read-Write) Specifies the default manner in which calibrations that are performed using SCPI or COM are to be stored. Cal data is ALWAYS stored to the channel Cal Register regardless of this setting.

This setting survives instrument preset and reboot. It remains until changed by another execution of this command.

Note: Cal Set arguments used with commands such as SENS:CORR:COLL:GUID:INIT, SENS:CORR:COLL:GUID:SAVE and SENS:CORR:COLL:GUID:SAVE:CSET will override of any of these default preference settings.

Learn about Cal Registers and User Cal Sets.

Parameters

<char> CALRegister - Cal is saved ONLY to the channel Cal Register.

USER - Cal is saved to a new User Cal Set file when performing a SCPI calibration. The Cal Set name is automatically generated. To change the name, send SENS:CORR:CSET:NAME after the cal is complete. This reverts to pre-6.0 behavior.

REUSE - The cal is saved to the Cal Set that is currently selected on the specified channel, which could be the channel Cal Register. If the channel does not yet have a selected Cal Set, the cal will be saved to a new User Cal Set with an automatically-generated name.

Examples

SENS:CORR:PREF:CSET:SAVE USER
sense:correction:preference:cset:save reuse

Query Syntax

SENSe:CORRection:PREFerence:CSET:SAVE?

Return Type

Character

Default

CALRegister

SENSe:CORRection:PREFerence:CSET:SAVUser <bool> Superseded
This command is replaced with `SENS:CORR:PREF:CSET:SAVE`.

**NOTE:** With 6.0 we implemented a change that defaults to saving completed calibrations to Cal Registers instead of User Cal Sets. To revert to the old behavior, send this command as ON (1). For UI and COM use, this can be done from the GPIB console.

*(Read-Write)* Specifies whether cal data is automatically saved to a User Cal Set file after performing a SCPI calibration. Cal data is always saved to a Cal Register regardless of this setting.

This setting survives instrument preset and reboot. It remains until changed by another execution of this command.

Learn about [Cal Registers and User Cal Sets](#).

**Parameters**

<`bool`> **ON** or **1** - Cal is automatically saved to a User Cal Set file when performing a SCPI calibration. The Cal Set name is automatically generated. To change the name, send `SENS:CORR:CSET:NAME` after the cal is complete. Reverts to pre-6.0 behavior.

**OFF** or **0** - Cal is NOT automatically saved to a User Cal Set. To save a calibration to a User Cal Set, use `SENS:CORR:COLL:GUID:INIT`.

**Examples**

```
SENS:CORR:PREF:CSET:SAV 1
```

```
sense:correction:preference:cset:savuser 0
```

**Query Syntax**

`SENSe:CORRection:PREFerence:CSET:SAVUser?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF (0)
Specifies whether or not the PNA should perform orientation of the ECal module during calibration. Orientation is a technique by which the PNA automatically determines which ports of the module are connected to which ports of the PNA. Orientation begins to fail at very low power levels or if there is much attenuation in the path between the PNA and the ECal module. If orientation is turned OFF, the `SENS:CORR:PREF:ECAL:PMAP` command must be used to specify the port connections before performing a cal.

**Note:** For 3-port or 4-port measurements, when orientation is OFF, you are not allowed to specify how the ECAL module is connected. Instead, the PNA determines the orientation. Use `SENS:CORR:COLL:GUID:DESC?` to query the orientation. The PNA does not verify that you made the connection properly.

This setting remains until the PNA is restarted or this command is sent again.

### Parameters

<bool> ECAL orientation state. Choose from:

- **ON** or **1** - PNA performs orientation of the ECal module.
- **OFF** or **0** - PNA does NOT perform orientation of the ECal module.

### Examples

```
SENS:CORR:PREF:ECAL:ORI OFF
```

```
sense:correction:preference:ecal:orientation:state on
```

### Query Syntax

```
SENSe:CORRection:PREFerence:ECAL:ORIentation[:STATe]?
```

### Return Type

Boolean (1 = ON, 0 = OFF)

### Default

ON (1)

---

**SENSe:CORRection:PREFerence:ECAL:PMAP <module>,<string>**

(Read-Write) When ECal module orientation is turned OFF (SENS:CORR:PREF:ECAL:ORI), this command specifies the port mapping (which ports of the module are connected to which ports of the PNA) prior to performing ECal calibrations.

This setting remains until the PNA is restarted or this command is sent again.

### Parameters

- **<module>** Specifies which ECal module this port map is being applied to. Choose from:

  - **ECAL1**
  - through.
  - **ECAL8**
Format this parameter in the following manner:

Aw,Bx,Cy,Dz

where

- A, B, C, and D are literal ports on the ECAL module
- w,x,y, and z are substituted for PNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECal module with

- port A connected to PNA port 2
- port B connected to PNA port 3
- port C not connected
- port D connected to PNA port 1

the string would be: A2,B3,D1

If either the receive port or source port (or load port for 2-port cal) of the CALC:PAR:SELected measurement is not in this string and orientation is OFF, an attempt to perform an ECal calibration will fail.

**Examples**

```
sense:correction:preference:ecal:pmap ecal2, 'a2,b1,c3'
```

**Query Syntax**

```
SENSe:CORRection:PREFerence:ECAL:PMAP? <module>
```

**Return Type**

String

**Default**

Null string ()
(Read-Write) Sets and returns a preference for the Unguided Cal behavior described below. This setting persists until it is changed.

This preference can also be set True by executing the script on the PNA at C:/Program Files/Agilent/Network Analyzer/System/wincal32.reg.

**Parameters**

<table>
<thead>
<tr>
<th>&lt;bool&gt;</th>
<th>Boolean - Choose from:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - <strong>False</strong></td>
<td>Reverts to new (preferred) behavior. An error is returned if standard data is not acquired before sending SENS:CORR:COLL:SAVE.</td>
</tr>
<tr>
<td>1 - <strong>True</strong></td>
<td>(WinCal compatible) Prevents SENS:CORR:COLL:SAVE from failing when standard data has not, and will not, be acquired.</td>
</tr>
</tbody>
</table>

Learn more about old and new behaviors.

**Examples**

```
SENS:CORR:PREF:SIMC 0
sense:correction:preference:simcal 1
```

**Query Syntax**

SENSe:CORRection:PREFerence:SIMCal?

**Return Type**

Boolean

**Default**

0

---

**SENSe:CORRection:PREFerence:TRIG:FREE <char>, <bool>**

(Read-Write) Sets and returns the preference for the trigger behavior during a calibration. This setting persists until it is changed.

**Note:** If TRIGger:SOURce = Manual, during a calibration the PNA ALWAYS switches to Internal for one trigger, then back to Manual, regardless of this preference command.

**Parameters**

<table>
<thead>
<tr>
<th>&lt;char&gt;</th>
<th>Character - Calibration type. Choose from:</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUIDed</td>
<td>preference setting pertains to a Guided calibration.</td>
</tr>
<tr>
<td>UNGuided</td>
<td>preference setting pertains to an Unguided calibration.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>&lt;bool&gt;</th>
<th>Boolean - Choose from:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - <strong>False</strong></td>
<td>The trigger behavior during the specified calibration type DOES respect the setting of the TRIGger:SOURce command. For example, when Trigger source = External, the single trigger method will wait for the External</td>
</tr>
</tbody>
</table>
trigger signal and then allow only one sweep.

1 - True - (Pre-6.0 behavior) The trigger behavior during the specified calibration type does NOT respect the setting of the TRIGger:SOURce command. For example, when Trigger source = External, during calibration the PNA switches to Internal sweep, responds to one trigger signal to measure the standard, then switches back to External.

**Examples**

```
SENS:CORR:PREF:TRIG:FREE GUID,1
sense:correction:preference:trig:free unguided,0
```

**Query Syntax**

```
SENSe:CORRection:PREFerence:TRIG:FREE? <char>
```

**Return Type**

Boolean

**Default**

False for both calibration types.

---

**SENSe<cnum>:CORRection:RPOWer:OFFSet[:AMPLitude] <num>**

*(Read-Write)* Adjusts a receiver power cal to account for components or adapters that are added between the source port and receiver while performing this cal. For more information, see Receiver Cal.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Offset Value in dB. Specify loss as a negative number; and gain as a positive number. Choose a number between -200 and 200.

**Examples**

```
SENS:CORR:RPow:OFFS .5
sense2:correction:rpower:offset:amplitude .-5
```

**Query Syntax**

```
SENSe<cnum>:CORRection:RPOWer:OFFSet[:AMPLitude]?
```

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>:CORRection:RVELoCity:COAX <num>**
(Read-Write) Sets the velocity factor to be used with Electrical Delay and Port Extensions.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Velocity factor. Choose a number between 0 and 10 (.66 polyethylene dielectric; .7 teflon dielectric)

**Examples**

```
SENS:CORR:RVEL:COAX .66
sense2:correction:rvelocity:coax .70
```

**Query Syntax**

```cnum>:CORRection:RVELocity:COAX?```

**Return Type**
Numeric

**Default**
1

---

**SENS<cn>:CORRection:SFORward[:STATe] <boolean>**

(Read-Write) Sets the direction a calibration will be performed when only one set of standards is used.

Use **SENS:CORR:TSTandards[:STATe] OFF** to specify that only one set of standards will be used.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<boolean>`: ON (1) - FORWARD direction of a 2-port calibration will be performed
  OFF (0) - REVERSE direction of a 2-port calibration will be performed

**Examples**

```
SENS:CORR:SFOR 1
sense2:correction:sforward:state 0
```

See an example using this command

**Query Syntax**

```cnum>:CORRection:SFORward[:STATe]?```

**Return Type**
Boolean

**Default**
ON

---

**SENS<cn>:CORRection[:STATe] <ON | OFF>**
(Read-Write) Turns error correction ON and OFF for the specified channel.

**Note:** Before using this command you must select a measurement using `CALC:PAR:SEL`. You can select one measurement for each channel.

**Parameters**
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<ON | OFF>`: ON (or 1) - correction is applied to the channel. OFF (or 0) - correction is NOT applied to the channel.

**Examples**
- `SENS:CORR ON`
- `sense2:correction:state off`

**Query Syntax**
- `SENSe<cnum>:CORRection[:STATe]?

To query the error correction state for a measurement, use `CALC:CORR:STATe?`

**Return Type**
- Boolean (1 = ON, 0 = OFF)

**Default**
- OFF

---

**SENSe<cnum>:CORRection:TCOLd:USER:VALue <num>**

(Read-Write) Sets and returns the temperature of the noise source connector. Learn more about [Noise Figure Calibration](#).

**Parameters**
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<num>`: Noise source temperature in Kelvin.

**Examples**
- `SENS:CORR:TCOL 295`
- `sense2:correction:tcold 298`

See an example using this command

**Query Syntax**
- `SENSe<cnum>:CORRection:TCOLd:USER:VALue?`

**Return Type**
- Numeric

**Default**
- Not Applicable

---

**SENSe<cnum>:CORRection:TSTandards[:STATe] <boolean>**
(Read-Write) Specifies the acquisition of calibration data using ONE or TWO sets of standards.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<boolean>`
  - **ON (1)** - TWO identical sets of standards will be used to simultaneously calibrate two ports (for both Forward and Reverse parameters).
  - **OFF (0)** - ONE set of standards will be used to perform a full 2-port calibration, one port at a time.

When specifying ON (use two sets), the [SENS:CORR:COLL:ACQuire](#) command uses the same standard index for each calibration class. To specify the calibration standard gender for each port, you must first ensure that the order of calibration class accurately reflects the configuration of your DUT. For example, for a DUT with a male connector on port 1 and a female connector on port 2, order the devices within the S11 classes (A, B, and C) such that the MALE standards are first in the list. Then order the S22 classes specifying the FEMALE standards as the first in the list.

**Examples**

```
SENS:CORR:TST 1
sense2:correction:tstandard:state 0
```

See an [example](#) using this command

**Query Syntax**

`SENSe<cnum>:CORRection:TSTandards[:STATe]?
`

**Return Type**

Boolean

**Default**

ON

`SENSe:CORRection:TYPE:CATalog? <char>`
(Read-Write) Lists the Cal Types in the PNA by either GUID or registered name. Learn more about applying Cal Type using SCPI.

**Note:** Before using this command you must select a measurement using `CALC:PAR:SEL`. You can select one measurement for each channel.

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;char&gt;</td>
<td>Specifies the type of list. Choose from:</td>
</tr>
<tr>
<td>GUID</td>
<td>the registered GUID of the Cal Type</td>
</tr>
<tr>
<td>NAME</td>
<td>the registered name of the Cal Type</td>
</tr>
</tbody>
</table>

### Examples

```
SENS:CORR:TYPE:CAT? GUID
```

### Query Syntax

```
SENSe<cnum>:CORRection:TYPE:CATalog? <char>
```

### Return Type

Comma-separated string

### Default

Not Applicable

---

Last modified:

- **6-Mar-2008**: Added Noise TCOLd
- **19-Sep-2007**: Added missing `<cnum>` arguments
- **July 30, 2007**: Added ENR commands
- **April 14, 2007**: Addd ECal isolation commands
- **Oct 30, 2006**: Modified SavUser command
**Sense:Correction:Collect:Guided Commands**

Performs and applies a SmartCal (Guided) calibration and other error correction features.

```
SENSe:CORRection:COLLect:GUIDed:
   ACQuire
   ADAPter
      | COUNt?
      | CREate?
      | DELay
      | DESCription
      | PATHs
   CKIT:PORT
      | CATalog?
      | [SELect]
   CONNector
      | CATalog?
      | PORT
      | [SELect]
   DESCription
   DMATch
      | APPLy
      | [IMMediate]
      | PORTs?
      | [INITiate]
   ETERms:LOAD
      | [CSET]
   INITiate
   ISOLation
      | AVERage
      | INCRement
      | PATHs
   METHOD
   PACQuire
   PATH
      | CMETHod
      | TMETHod
   SAVE
      | CSET
```
SENSe:CORRection:COLLect:GUIDed:ACQuire <std>

(Write-only) Initiates the measurement of the specified calibration standard. Executing this command with an unnecessary standard has no affect.

The measured data is stored and used for subsequent calculations of error correction coefficients. All standards must be measured before a calibration can be completed. Any measurement can be repeated until the SENS:CORR:COLL:GUID:SAVE command is executed.

Query the user prompt description using SENS:CORR:COLL:GUID:DESC?

Query the required calibration steps using SENS:CORR:COLL:GUID:STEP?

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Any existing channel number. If unspecified, value is set to 1</td>
</tr>
<tr>
<td>&lt;std&gt;</td>
<td>Choose from: STAN1, STAN2, STAN3, through STAN40</td>
</tr>
</tbody>
</table>

Examples

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:COLL:GUID:ACQ STAN1</td>
<td>Sends: correction:collect:guided:acquire stan1</td>
</tr>
</tbody>
</table>

Query Syntax

Not Applicable

Default

Not Applicable
Specifies the use of a THRU adapter to be used during the Guided Cal Unknown THRU and Adapter Removal Cal. Returns an adapter index <n> which is used to refer to the adapter in several related commands. See Cal Thru Methods. While the choice of which end of the adapter is <conn1> and <conn2> is arbitrary, it is necessary to remember which will be used on each test port.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<conn1>` Adapter port 1 connector type. Use SENS:CORR:COLL:GUID:CONN:CAT? to return a list of valid connector types.
- `<conn2>` Adapter port 2 connector type.

**Examples** See example using this command.

**Return Type** Numeric

**Default** Not Applicable

---

**SENS<cnm>:CORRection:COLLect:GUIDed:ADAPter:COUNt?**

(Read-Only) Returns the number of THRU adapters that have been created for this calibration using SENS:CORR:COLL:GUID:ADAP:CREate?

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples** See example using this command.

**Return Type** Numeric

**Default** Not Applicable

---

**SENS<cnm>:CORRection:COLLect:GUIDed:ADAPter<n>:DELay <coax>, [w phase, wdelay]**
(Write-only) Specifies the adapter delay and optionally waveguide delay and optional phase offset (degrees) of adapter <n>.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Adapter index number that was returned from `SENS:CORR:COLL:GUID:ADAP:CREate?`
- `<coax>` Delay value of coax adapter <n> in seconds. If the adapter has no coax connector, enter 0.
- `<wphase>` Waveguide phase offset in degrees. If the adapter has no waveguide connector, do not enter a value.
- `<wdelay>` Waveguide delay in seconds. If the adapter has no waveguide connector, do not enter a value.

**Examples**

See example using this command.

**Default** Not Applicable

---

**SENSe<cnum>:CORRection:COLLect:GUIDed:ADAPter<n>:DESCription <string>**

(Write-only) Specifies the adapter description for use as the guided cal connection prompts.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Adapter index number that was returned from `SENS:CORR:COLL:GUID:ADAP:CREate?`
- `<string>` Adapter description.

**Examples**

See example using this command.

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**SENSe<cnum>:CORRection:COLLect:GUIDed:ADAPter<n>PATHs <port pairs>**
(Write-only) Specifies the port pairs for which the adapter will be used for a THRU connection.

For example, for a 3-port cal on channel 1 using ports 1,2,and 3), to use adapter 1 between the ports (1 to 2) and (1 to 3) the following command is used: SENS1:CORR:COLL:GUID:ADAP1:PATH 1,2,1,3.

The adapter must have the same DUT connectors as the ports that are already specified for these ports.

**Parameters**

- `<cnum>`   Any existing channel number. If unspecified, value is set to 1
- `<n>`     Adapter index number that was returned from SENS:CORR:COLL:GUID:ADAP:CREate?
- `<port pair>` Ports for which the adapter will be used. The orientation is not critical, as the PNA will align the connector types as necessary. The minimum number of Thru connections required is the number of ports to calibrated -1.

**Examples** See example using this command.

**Query Syntax** Not Applicable

**Default** Not Applicable

---


(Read-only) Returns a comma-separated list of valid kits for each port. In addition to mechanical calibration kits, this will include applicable characterizations found within ECal modules currently connected to the PNA.

If two or more identical ECal modules are connected to the PNA, the returned list will include the serial numbers to distinguish the ECal modules.

Use items in the list to select the kit to be used with the SENS:CORR:COLL:GUID:CKIT:PORT command.

**Parameters**

- `<pnum>` Any existing port number. If unspecified, value is set to 1

**Examples**


'When "Type N (50) male" is specified for connector type, returns:

"85054D, 85032F"

'When two identical ECal modules are connected for the connector type, 'the return string includes serial numbers

(Read-Write) Specifies the calibration kit for each port to be used during a guided calibration. An unused port does NOT need to have a specified Cal Kit.

If two or more identical ECal modules are connected to the PNA, the serial number must be included in the <kit> string to distinguish the ECal modules. See SENS:CORR:COLL:GUID:CKIT:PORT:CAT?

Note: Sliding loads are not fully supported from Sens:Corr:Coll:Guided... The Measure button must be manually pressed.

1. Specify the connector type for the port with SENS:CORR:COLL:GUID:CONN:PORT.

2. Query the valid available kits for the connector on each port with SENS:CORR:COLL:GUID:CKIT:PORT:CAT?

3. Specify the kit using this command.

4. Perform a query of this command. If the <kit> parameter was incorrectly entered, an error will be returned.

Parameters

- `<pnum>` Any existing port number. If unspecified, value is set to 1
- `<kit>` Calibration kit to be used for the specified port.

Examples

```
SENS:CORR:COLL:GUID:CKIT:PORT1 '85055A'
'The following includes a serial number when two or more ECal mods are connected
sense:correction:collect:guided:ckit:port2:select '85092-60010 ECal 10685'
```

Query Syntax
SENSe:CORRection:COLLect:GUIDed:CKIT:PORT<pnum>[:SELect]?

Return Type
String - If the <kit> parameter was incorrectly entered while writing, an error will be returned.

Default Not Applicable
The PNA remembers previous Guided Cal settings. Therefore, for completeness, unused ports should be defined as "Not used". See Guided Cal examples.

- A single port with a valid <conn> name indicates a 1-Port calibration will be performed.
- Two ports with valid <conn> names indicate either a 2-Port SOLT or TRL calibration will be performed depending on the standards definition found within the cal kit and the capability of the PNA.
- Three ports with valid <conn> names indicate a 3-Port calibration will be performed, and so forth.

The following steps could be followed to ensure port connectors are specified correctly:

2. Select a connector type using this command.
3. Perform a query of this command. If the <conn> parameter was incorrectly entered, an error will be returned.
4. Specify the cal kit to use for each port with SENS:CORR:COLL:GUID:CKIT:PORT

Parameters

- <pnum> Any existing port number. If unspecified, value is set to 1
- <conn> String - DUT connector type to connect with PNA port <pnum>
Examples

'Specifying a 2-port cal (1 & 2) on a 4-port PNA

SENS:CORR:COLL:GUID:CONN:PORT1 'Type N (50) female'
SENS:CORR:COLL:GUID:CONN:PORT2 'Type N (50) male'
SENS:CORR:COLL:GUID:CONN:PORT3 'Not Used'
SENS:CORR:COLL:GUID:CONN:PORT4 'Not Used'

Query Syntax

SENSe:CORRection:COLLect:GUIDed:CONNector:PORT<pnum>[:SELect]?

Return Type

String

Default

Not Applicable


(Read-only) Returns the connection description for the specified calibration step.

Parameters

<step>  A number from 1 to the number of steps required to complete the calibration
(Use  SENS:CORR:COLL:GUID:STEP? to query the number of steps )

Examples

SENS:CORR:COLL:GUID:DESC? 10

'Returns:
Connect APC 7 Open to port3

Return Type

String

Default

Not Applicable

SENSe:CORRection:COLLect:GUIDed:DMATch:APPLy[:IMMediate] [<CalSetGUID>]

(Write-only) Specifies a Cal Set as a source of delta match correction.

If CalSetGUID is not specified, then the Global Delta Match Cal Set is assumed. An error is returned if the specified Cal Set does not meet the following Delta Match criteria. The Global Delta Match Cal can ALWAYS be applied.

- Must have been performed using ECal or as a guided mechanical cal (not Unguided).
- Must have the same start freq, stop freq, and number of points as the channel being calibrated.
- Must calibrate the ports that are required by the TRL or Unknown Thru cal as indicated by SENS:CORR:COLL:GUID:DMATch:APPLy:PORTs?

Learn more about Delta match calibration.

See example of a complete Delta Match calibration.
### Parameters

<CalSetGUID> Optional. GUID of the Cal Set to use. If unspecified, the Global Delta Match Cal Set is used.

### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:COLL:GUID:DMAT:APPL</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

### Query Syntax

**SENSe:CORRection:COLLect:GUIDed:DMATch:APPLy:PORTs?**

(Read-only) Returns the port numbers for which delta match correction is required. 0 (zero) is returned if the Cal does NOT require Delta Match correction for one of the following reasons:

- The Cal does NOT involve Unknown THRU or TRL. You specify this using SENS:CORR:COLL:GUID:METH <UNKN | TRL>.

- The Cal DOES involve Unknown THRU or TRL, but the delta match data can be calculated by the Unknown Thru or TRL Cal. Learn how this is possible. However, you can force the Cal to use the Delta Match data from a Cal Set. Learn more about Delta match calibration.

See example of a complete Delta Match calibration.

### Parameters

### Examples

<table>
<thead>
<tr>
<th>Query Syntax</th>
<th>Default</th>
</tr>
</thead>
</table>

### Return Type

Numeric

### Default

Not Applicable

### SENSse:CORRection:COLLect:GUIDed:DMATch[:INIItiate] <conn>,<cKit>
(Write-only) Initiates a global delta match calibration.

Learn more about Global Delta Match calibration.

See example of a complete Delta Match calibration.

**Parameters**

- `<conn>` **String**. Connector type for port 1. All other ports are set automatically.
- `<cKit>` **String** Cal Kit for all ports. If incorrectly entered while writing, an error is returned.

**Examples**

- SENS:CORR:COLL:GUID:DMAT 3.5 female,"85052B"

**Query Syntax**

Not Applicable

**Default**

Not Applicable

(SENS)e:CORRection:COLLect:GUIDed:ETERms:LOAD[:CSET] `<cset>`, `<calPort>` [,`csPort`]

(Write-only) Loads 1-port error terms from a Cal Set into the current Guided Cal sequence. When the Cal steps are recomputed, connection steps are removed due to the loading of the error terms.

See example of how to use this command.

**Parameters**

- `<cset>` **String** Name of User Cal Set in which the error terms reside.
- `<pnum>` **Integer** Port number of the current cal to receive error terms.
- `[csPort]` **Integer** Optional argument. Port number associated with the error terms in the Cal Set. If unspecified, the same port number as `<calPort>` is used.

**Examples**

See example

**Query Syntax**

Not Applicable

**Default**

Not Applicable

(SENS)e<cnum>ese:CORRection:COLLect:GUIDed:INITiate [cal set name or GUID [,bool]]
(Write-only) Initiates a guided calibration.


- After this command is executed, subsequent commands can be used to query the number of measurement steps, issue the acquisition commands, query the connection description strings, and subsequently complete a guided calibration.

### Parameters

- `<cnum>` Channel to be calibrated. Must be the active channel. Use `Calc:Par:Sel` to activate any measurement on the channel.

- `<cal set name or GUID> (String)` Optional argument.
  - If specified, choose an existing Cal Set, either by name or by GUID.
  - If NOT specified, behavior depends on the `SENS:CORR:PREFerence:CSET:SAVE` setting.
    - By Cal Set name: include quotes.
    - By Calset GUID in the form: `{GUID}`; including quotes and curly brackets.
    - Query all Cal Set GUIDs with `SENS:CORR:CSET:CAT?` An error is reported if the Cal Set is not found.

  The Cal Set is either supplemented or overwritten depending on the method, connectors, and ports selected. Learn more about Cal Sets.

- `<bool>` Optional argument.
  - **False (0)** If Cal Set stimulus settings differ from the existing channel, do not change channel stimulus settings. The Cal Set is save to the current setting of the `SENS:CORR:PREF:CSET:SAVE` command. This is the default setting if not specified.

  **True (1)** If cal set stimulus settings differ from the existing channel, change the channel stimulus settings to match the Cal Set settings.

### Examples

```
sense2:correction:collect:guided:initiate 'myCalSet'
```

### Query Syntax

- **Default**: Not Applicable

(Read-Write) Specifies amount to increment (increase) the channel’s averaging factor during measurement of isolation standards in a guided calibration.

Note: If the channel has averaging turned OFF and the value of <num> is greater than 1, averaging will be turned ON only during the isolation measurements and with the averaging factor equal to <num>.

Parameters

<num> Amount to increment the averaging factor for the isolation measurement. The maximum averaging factor for the channel is 65536 (2^16).

Examples

'Measure isolation on all paths for the cal
SENSe:CORRection:COLLect:GUIDed:ISOLation ALL

'Remove the port pairs 1-to-2 and 1-to-3 from the list of paths on which to measure isolation
SENSe:CORRection:COLLect:GUIDed:ISOLation:PATHs REMove,1,2,1,3

Query Syntax
SENSe:CORRection:COLLect:GUIDed:ISOLation:PATHs?

Note: if isolation is not measured, the query returns 0.

Return Type Numeric

Default 8 - If this command is NOT sent, but isolation is measured, then averaging will be turned ON with factor set to 8 during the isolation measurements.

SENSe:CORRection:COLLect:GUIDed:ISOLation[:PATHs] <char>,<p1a, p1b, p2a, p2b,...>
(Read-Write) Specifies the paths (port pairs) to make isolation measurements on during a guided calibration.

**Parameters**

<char>

**ALL** – Measure isolation on all pairings of the ports that are to be calibrated.

**NONE** – Do not measure isolation on any pairing of the ports to be calibrated. (Default behavior).

**ADD** – Add one or more specific pairings of ports to the list of port pairings for which isolation will be measured.

**REMove** – Remove one or more specific pairings of ports from the list of port pairings for which isolation will be measured. If many paths are to be measured, it may be easier to first send **ALL**, then **REMove** and specify the paths to remove.

<p1a, p2a...> For use when <char> is **ADD** or **REMove**.

Specify Port numbers in pairs:

- For 3-port cals, specify up to 3 pairs.
- For 4-port cals, specify up to 6 pairs.

p1a, p1b  (Path1 - port A and port B)

p2a, p2b  (Path2 - port A and port B)

p3a, p3b  (Path3 - port A and port B)

**Examples**

'Measure isolation on all paths for the cal

SENSe:CORRection:COLLect:GUIDed:ISOLation:PATHs **ALL**

'Remove the port pairs 1-to-2 and 1-to-3 from the list of paths on which to measure isolation

sense:correction:collect:guided:isolation:paths **REMove**,1,2,1,3

**Query Syntax**

SENSe:CORRection:COLLect:GUIDed:ISOLation:PATHs?

**Note:** if isolation is not be measured on any of the paths, the query returns 0

**Return Type**

Numeric

**Default**

0 - Isolation not measured on any paths.
SENSe:CORRection:COLLect:GUIDed:METHod <char> Superseded


(Read-Write) Selects from one of several algorithms available for performing the THRU portion of a guided calibration. Learn more about THRU methods.

Parameters

<char> DEFAULT - Informs guided calibrations to use the default algorithm when computing the number of needed standards acquisition steps. (default selection if omitted.)

ADAP - Use the adapter removal algorithm

FLUSH - Use with insertable devices.

UNKN - Use the Unknown THRU algorithm with calibrations for non-insertable devices.

DEFINED - Use the THRU definition that you stored in the cal kit file, or ECal module.

TRL - Select TRL Cal Type for guided cals. Valid for "TRL ready" Cal Kits with properly assigned TRL cal classes.

SOLT - Select SOLT Cal Type for guided cals. Valid for any kit with properly assigned SOLT cal classes.

Examples

SENSe:CORRection:COLLect:GUIDed:METH ADAP
sense:correction:collect:guided:method unkn

Query Syntax

SENSe:CORRection:COLLect:GUIDed:METHod?

Return Type

Character

Default

SENSe:CORRection:COLLect:GUIDed:PACQuire <std>
(Read-Write) Show the Cal Window, and optionally one or more other specific windows before acquiring a Cal standard. This command will cause the Cal Window to display the specific measurements that are to be made for that particular Cal standard.

Parameters

<std>  Choose from: STAN1, STAN2, STAN3, through STAN40.

Examples

SENS:CORR:COLL:GUID:PACQuire STAN2
sense:correction:collect:guided:pacquire STAN5

See an example that uses this command.

Query Syntax

SENSe:CORRection:COLLect:GUIDed:PACQuire?

Return Type

Character

Default  Not Applicable

SENSe:CORRection:COLLect:GUIDed:PATH:CMEThod
<port1>,<port2>,<caltype1>[,caltype2]>

Note: This command replaces SENS:CORR:COLL:GUID:METH.

(Read-Write) Specifies the calibration method for each port pair.

Note: Before using this command, first do the following:

- Set the connector types: SENS:CORR:COLL:GUID:CONN:PORTn
- Set or query the thru path pairs: SENS:CORR:COLL:GUID:THRU:PORT

After sending or querying this command, send the Thru method:

SENS:CORR:COLL:GUID:PATH:TMEThod

Parameters

<port1>  First port of the pair to be calibrated.
<port2>  Second port of the pair to be calibrated.
<caltype1>[caltype2]>  (String) Cal types for 1st and 2nd ports of the pair, enclosed in a single pair of quotes. NOT case-sensitive.

   caltype1  Cal type for the pair if caltype2 is not specified. Otherwise, Cal type for port 1. Choose from:
• “TRL”
• “SOLT”
• “QSOLTN”
• “EnhRespn”

For the last two arguments, replace N with the port to be used as the source port, which MUST be one of the port pair.

caltype2  Optional argument. Use only when performing an adapter removal cal on the pair. This argument specifies the Cal type on the second port. Caltype1 then specifies the Cal type of the first port.

Choose from the same arguments as caltype1.

Examples

```
SENS:CORR:COLL:GUID:PATH:CMEThod 2,3,"QSOLTN2"
```

sense:correction:collect:guided:path:cmethod 2,3,"solt,trl"

Query Syntax

SENSe:CORRection:COLLect:GUIDed:PATH:TMEThod? <port1>,<port2>

If only one caltype is returned then it’s NOT adapter removal.

Return Type  String

Default  Not Applicable

SENSe:CORRection:COLLect:GUIDed:PATH:TMEThod
<br/>&lt;port1&gt;,&lt;port2&gt;,&lt;thruType1[,thruType2]&gt;

Note: This command replaces SENS:CORR:COLL:GUID:METH.

(Read-Write) Specifies the calibration THRU method for each port pair.

Note: Before using this command, first do the following:

- Set the connector types: SENS:CORR:COLL:GUID:CONN:PORTn
- Set or query the thru path pairs: SENS:CORR:COLL:GUID:THRU:PORT
- Set or query the Cal Type: SENS:CORR:COLL:GUID:PATH:CMEThod

Learn more about thru methods.

Parameters
<port1> First port of the port pair to be calibrated.
<port2> Second port of the port pair to be calibrated.
<thruType1[,thruType2]> (String) Thru methods for 1st and 2nd ports of the pair, enclosed in a single pair of quotes. NOT case-sensitive.

**thruType1** Calibration thru method for the pair if thruType2 is not specified. Otherwise, thru method for port 1.

Choose from:

- **“Defined Thru”** A thru type for which there is a stored definition in the Cal Kit.
- **“Zero Thru”** Zero length thru, also known as flush-thru.
- **“Undefined Thru”** A thru type for which there is NOT a stored definition in the Cal Kit. Also known as Unknown Thru. Valid ONLY for SOLT cal type.
- **“Undefined Thru using a Defined Thru”** Using an ECal module, measure the internal thru using the "Undefined Thru" method.

**thruType2** Optional argument. Use only when performing an adapter removal cal on the pair as determined by SENS:CORR:COLL:GUID:PATH:CMEThod. The only valid arguments for ThruType1&2 is "Defined Thru, Defined Thru".

**Examples**

<table>
<thead>
<tr>
<th>Query Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>sense:correction:collect:guided:path:tmethod 2,3,&quot;Defined Thru,Defined Thru&quot;</td>
</tr>
</tbody>
</table>

**Return Type** String

**Default** Not Applicable

**SENSe:CORRection:COLLect:GUIDed:SAVE [bool]**
(Write-only) Completes the guided cal by computing the error correction terms, turning Correction ON, and saving the calibration to a Cal Set.

If all of the required standards have not been measured, the calibration will not complete properly.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[bool]</td>
<td>Optional argument. If unspecified, the default behavior is the current PNA preference setting of <code>SENSe:CORRection:PREFerence:CSET:SAVE</code></td>
</tr>
<tr>
<td><strong>False</strong> (0)</td>
<td>Save cal data ONLY to a Cal Register.</td>
</tr>
<tr>
<td><strong>True</strong> (1)</td>
<td>Save cal data to a Cal Register and a User Cal Set. The filename is automatically generated.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENSe:CORRection:COLLect:GUIDed:SAVE
sense2:correction:collect:guided:save 0
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

`SENSe:CORRection:COLLect:GUIDed:SAVE:CSET <cal set name or guid>`

(Write-only) Completes the guided cal by computing the error correction terms, turning Correction ON, and saving the calibration to the specified Cal Set. This command performs the same function as `SENSe:CORRection:COLLect:GUIDed:SAVE`, but this command allows the name or GUID of the Cal Set to be specified.

- Use this command instead of the optional name or GUID argument in `SENSe:CORR:COLL:GUID:INIT`.
- Use `SENSe:CORRection:CSET` commands to get names or GUIDs of existing Cal Sets.
- The cal data is also saved to the channel Cal Register.
- If all of the required standards have not been measured, the calibration will not complete properly.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cal set name or guid&gt;</code></td>
<td><code>String</code> - Name or GUID of an existing Cal Set to be overwritten. If specifying a GUID, curly brackets must be included.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENSe:CORRection:COLLect:GUIDed:SAVE:CSET "MyCalSet"
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable
SENSe:CORRection:COLLect:GUIDed:STEPs?

(Read-only) Returns the number of measurement steps required to complete the current guided calibration. This command is sent after the SENS:CORR:COLL:GUID:INIT, SENS:CORR:COLL:GUID:CONN:PORT and SENS:CORR:COLL:GUID:CKIT:PORT commands.

**Examples**

```
SEN:COR:COL:GUID:STEP?
sense2:correction:collect:guided:steps?
```

**Return Type** Numeric

**Default** Not Applicable

SENSe:CORRection:COLLect:GUIDed:THRU:PORTs <t1a, t1b, t2a, t2b, t3a, t3b...>

(Read-Write) For calibrating more than 2-ports ONLY. Specifies the port pairs for the thru connections of the calibration.

Use the query to learn the port pairs that the PNA suggests.

Use the set command to override these thru pairs.

**Parameters**

- **<t1a,...>** Specify Port numbers in pairs:
  - For 3-port cals, specify two or three pairs.
  - For 4-port cals, specify from three up to six pairs.

  t1a, t1b  (Thru1 - port A and port B)
  t2a, t2b  (Thru2 - port A and port B)
  t3a, t3b  (Thru3 - port A and port B)
  ...and so forth up to six pairs.

**Examples**

```
SEN:COR:COL:GUID:THRU:PORT 1,2,1,3,1,4  '4-port measurement
sense:correction:collect:guided:thru:ports 1,2,2,3  '3-port measurement
```

**Query Syntax** SENSe:CORRection:COLLect:GUIDed:THRU:PORTs?

**Return Type** Numeric

**Default** Port pairings that were used in the previous cal.
Last modified:

1-Nov-2007   Added PAcquire command
April 14, 2007 Added Cal Set by name and isolation commands
8-Mar-2007    Added CMethod and TMethod
23-Oct-2006   Fixed wording in Conn:Port:Sel
**Sense:Correction:CKIT Commands**

Manages the list of cal kits that are installed in the PNA.

- Click on a **blue** keyword to view the command details.
- New [See Calibrating the PNA Using SCPI](#)
- Learn about Modifying Cal Kits
- Synchronizing the PNA and Controller
- SCPI Command Tree

**SENSe:CORRection:CKIT:CLEAR[:IMMediate]**

*(Write-only)* Deletes ALL installed cal kits.

- **Examples**
  - SENS:CORR:CKIT:CLE

- **Query Syntax** Not Applicable

- **Default** Not Applicable

**SENSe:CORRection:CKIT:COUNt?**

*(Read-only)* Returns the number of installed cal kits.

- **Examples**
  - SENS:CORR:CKIT:COUNt?

- **Query Syntax** SENS:CORR:CKIT:COUNt?

- **Return Type** Numeric

- **Default** Not Applicable
**SENSe:CORRection:CKIT:ECAL<mod>:CLIST?**

*(Read-only)* Returns a list of characterizations stored in the specified ECal module.

**Parameters**

- `<mod>` ECAl module from which to read user characterization numbers. If unspecified, value is set to 1.

**Examples**

Module 1 contains User Characterizations 1 and 3.

SENSe:CORRection:CKIT:ECAL:CLIST?

'Returns the following (0 always indicates the factory characterization):

0,1,3

**Return Type** Numeric list, separated by commas.

**Default** Not Applicable

---

**SENSe:CORRection:CKIT:ECAL<mod>:INFormation? [<char>]**

*(Read-only)* Reads the user-characterization information from the specified ECal module. This command returns the same values as SENS:CORR:COLL:CKIT:INF?

**Parameters**

- `<mod>` ECAl module to read characterizations from. Choose from:
  
  1 through 8. If unspecified, value is set to 1.

- `<char>` Optional argument. Specifies which characterization within the ECal module to read information from. If not specified, value is set to CHAR0.

Choose from:

- **CHAR0** Factory characterization (data that was stored in the ECal module by Agilent)
- **CHAR1** User characterization #1
- **CHAR2** User characterization #2
- **CHAR3** User characterization #3
- **CHAR4** User characterization #4
- **CHAR5** User characterization #5

**Examples**

SENSe:CORRection:CKIT:ECAL:LIST?

(Read-only) Returns a list of index numbers to be used for referring to the ECal modules that are currently attached to the PNA.

**Examples**

```
SENS:CORR:CKIT:ECAL:LIST?

'If 2 modules are attached to the PNA
'then the returned list will be:

1, 2
```

**Return Type** Numeric list, separated by commas.

**Default** Not Applicable

---

SENSe:CORRection:CKIT:ECAL[num]:PATH:COUNt? <path>

(Read-only) Returns the number of unique states that exist for the specified path name on the selected ECal module.

This command performs exactly the same function as CONT:ECAL:MOD:PATH:COUNt?

Use the CONT:ECAL:MOD:PATH:STAT command to set the module into one of those states.

Use SENS:CORR:CKIT:ECAL:PATH:DATA? to read the data for a state.

**Parameters**

- **[num]** Optional argument. USB number of the ECal module. If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use SENS:CORR:CKIT:ECAL:LIST? to determine how many, and SENS:CORR:CKIT:ECAL:INF? to verify their identities.

- **<path>** Name of the path for which to read number of states. Choose from:

  Reflection paths
Transmission paths

- AB
- AC (4-port modules)
- AD (4-port modules)
- BC (4-port modules)
- BD (4-port modules)
- CD (4-port modules)

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CONT:ECAL:MOD:PATH:COUNt?</code></td>
<td>Control ECal module path count</td>
</tr>
<tr>
<td><code>control:ecal:module2:path:count?</code></td>
<td>Control ECal module path count</td>
</tr>
</tbody>
</table>

**Return Type**

- **Integer**

**Default**

- Not Applicable

---

**SENSe<ch>:CORRection:CKIT:ECAL[num]:PATH:DATA? <path>, <stateNum>,<char>|**

(Read-only) Returns the data for a state from the memory of the selected ECal module. The returned data is interpolated if necessary to have the same stimulus values as the specified channel `<ch>`. 

- For a reflection path state, the data is reflection S-parameter data. The number of values equals the number of stimulus points on the channel multiplied by 2 (because they are complex numbers).
- For a transmission path state, the data is all 4 S-parameters of the state. The number of values returned is 4 times that of a reflection state.

The data is returned in the same format as `CALC:DATA:SNP?`

**Note:** This command returns SNP data without header information, and in columns, not in rows as .SnP files. This means that the data returned from this command sends all frequency data, then all Sx1 magnitude data, then all Sx1 phase data, and so forth.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
[num] Optional argument. USB number of the ECal module. If unspecified (only one ECal module is connected to the USB), <num> is set to 1. If two or more modules are connected, use SENS:CORR:CKIT:ECAL:LIST? to determine how many, and SENS:CORR:CKIT:ECAL:INF? to verify their identities.

<path> Name of the path for which to read number of states. Choose from:

Reflection paths

- A
- B
- C (4-port modules)
- D (4-port modules)

Transmission paths

- AB
- AC (4-port modules)
- AD (4-port modules)
- BC (4-port modules)
- BD (4-port modules)
- CD (4-port modules)

<stateNum> Number of the state to set. Refer to the following table to associate the <stateNum> with a state in your ECal module.

In addition, CONT:ECAL:MOD:PATH:COUNt? returns the number of states in the specified ECal module.
<table>
<thead>
<tr>
<th>&lt;stateNum&gt;</th>
<th>N4432A and N4433A States</th>
<th>N4431A States</th>
<th>N469x States**</th>
<th>8509x States</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**One-Port Reflection States**

<table>
<thead>
<tr>
<th></th>
<th>Open</th>
<th>Open</th>
<th>Impedance 1</th>
<th>Open</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Short</td>
<td>Short</td>
<td>Impedance 2</td>
<td>Short</td>
</tr>
<tr>
<td>3</td>
<td>Impedance 1</td>
<td>Impedance 1</td>
<td>Impedance 3</td>
<td>Impedance 1</td>
</tr>
<tr>
<td>4</td>
<td>Impedance 2</td>
<td>Impedance 2</td>
<td>Impedance 4</td>
<td>Impedance 2</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>Impedance 5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td>Impedance 6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td>Impedance 7</td>
<td></td>
</tr>
</tbody>
</table>

**Two-Port Transmission States**

<table>
<thead>
<tr>
<th></th>
<th>Thru</th>
<th>Thru</th>
<th>Thru</th>
<th>Thru</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Confidence</td>
<td>Confidence</td>
<td>Confidence</td>
<td>Confidence</td>
</tr>
</tbody>
</table>

** The following modules have only FOUR Impedance states (1, 2, 3, 4): N4690B, N4691B, N4692A, N4696B.**

<char> Optional argument. Specifies which characterization within the ECal module to read information from. If not specified, value is set to CHAR0.

Choose from:

- **CHAR0** Factory characterization (data that was stored in the ECal module by Agilent)
- **CHAR1** User characterization #1
- **CHAR2** User characterization #2
- **CHAR3** User characterization #3
- **CHAR4** User characterization #4
- **CHAR5** User characterization #5
<table>
<thead>
<tr>
<th>Examples</th>
<th>SENS:CORR:CKIT:ECAL1:PATH:DATA?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return Type</strong></td>
<td>S1P or S2P</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

**SENSe:CORRection:CKIT:IMPort <string>**

(Write-only) Imports the specified cal kit (.ckt file) and appends the imported kit to the end of the list of kits whenever the file import succeeds.

**Parameters**
- `<string>`: Path and cal kit name.

**Examples**
SENSe:CORRection:CKIT:IMPort "C:/Program Files/Agilent/Network Analyzer/Documents/85033D.ckt"

**Query Syntax**
Not Applicable

**Default**
Not Applicable

**SENSe:CORRection:CKIT:INITialize[:IMMediate]**

(Write-only) Restores all default factory installed cal kits.

**Parameters**

**Examples**
SENSe:CORRection:CKIT:INITialize

**Query Syntax**
Not Applicable

**Default**
Not Applicable

**SENSe:CORRection:CKIT:LOAD <string>**

(Write-only) Loads the specified collection of cal kits from a .wks file. You can make your own collection of cal kits from the Advanced Modify Cal Kit menu.

**Parameters**
- `<string>`: Path and file name of the cal kit collection.

**Examples**
SENSe:CORRection:ckit:load "C:/Program Files/Agilent/Network Analyzer/PnaCalKits/factory/wMyCalKits.wks"

**Query Syntax**
Not Applicable

**Default**
Not Applicable
Last modified:

10/16/06   Modified Ecal:Data to include <ch>
Sense:Correction:Collect:Ckit Commands

Use to change the definitions of calibration kit standards.

SENSe:CORRection:COLLect:CKIT

<table>
<thead>
<tr>
<th>CONN</th>
<th>DESCl</th>
<th>NAMe</th>
<th>OLIST?</th>
<th>RL Set</th>
<th>STANdard</th>
</tr>
</thead>
<tbody>
<tr>
<td>INF?</td>
<td>OLAb</td>
<td>ORDer</td>
<td>[SELect]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHAR</td>
<td>FMAX</td>
<td>IMP</td>
<td>[SELect]</td>
<td>TZRreal</td>
<td></td>
</tr>
<tr>
<td>-C0 DELay</td>
<td>FMIN</td>
<td>-L0 LOSS</td>
<td>SDES</td>
<td>TYpe</td>
<td>TZimag</td>
</tr>
<tr>
<td>-C1</td>
<td></td>
<td>-L1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-C2</td>
<td></td>
<td>-L2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-C3</td>
<td></td>
<td>-L3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD</td>
<td>CAT?</td>
<td>DELete</td>
<td>FNAme</td>
<td>SNAme</td>
<td></td>
</tr>
</tbody>
</table>

Click on a blue keyword to view the command details.

Most of these commands act on the currently selected standard from the currently selected calibration kit.

- To select a Calibration kit, use SENS:CORR:COLL:CKIT:SEL.
- To select a Calibration standard, use SENS:CORR:COLL:CKIT:STAN:SEL
- See an example program that CREATES a New Cal Kit
- See an example program that MODIFIES an Existing Cal Kit
- Learn about Modifying Cal Kits
- Synchronizing the PNA and Controller
- SCPI Command Tree

Note: You should provide data for every definition field - for every standard in your calibration kit. If a field is not set, the default value may not be what you expect.

SENSe:CORRection:COLLect:CKIT:CONNector:ADD
<familly>,<start>,<stop>,<z0>,<gender>,<media>,<cutoff>
Creates a new connector. The connector is automatically added to the list of available connectors for the currently selected cal kit. If a connector includes both male and female connectors, each connector must be added separately.

**Parameters**

- **<family>** (String) Name of connector family. Limited to 50 characters.
- **<start>** Start frequency
- **<stop>** Stop frequency
- **<z0>** Characteristic Impedance of the connector in ohms.
- **<gender>** Connector gender. Choose from:
  - MALE
  - FEMALE
  - NONE
- **<media>** Media of the connector. Choose from:
  - COAX - coaxial
  - WAVE - waveguide
- **<cutoff>** Cutoff frequency of the connector (waveguide only).

**Examples**

```plaintext
SENS:CORR:COLL:CKIT:CONN:ADD "PSC 1.8 mm", 0.0, 999.9 GHz, 50, FEMALE, COAX, 0.0
SENS:CORR:COLL:CKIT:CONN:ADD "PSC 1.8 mm", 0.0, 999.9 GHz, 50, MALE, COAX, 0.0
```

**Query Syntax**

Not applicable

**Default**

Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:CONNector:CATalog?**

(Read-only) Returns a comma-separated list of all connectors defined within the currently selected cal kit. The returned string includes the connector family name followed by the connector gender, if any. Kits may include a primary connector family name and additional connector family names.

Connector family names are case sensitive. A connector family named "PSC 2.4" is different from a connector family named "psc 2.4".

Learn more about [Connector Family Name](#)

**Examples**

```plaintext
'RETURNED STRING
"Type-N (50) male, Type-N (50) female"
```

**Default**

Not Applicable
SENSe:CORRection:COLLect:CKIT:CONNector:DELeTe

(Write-only) Deletes the primary connector family name from the selected kit. The PNA allows multiple connector families for each kit. If a kit includes multiple connector families, only the first listed (primary) connector family name is deleted.

Once the connector family is deleted, the connector may not be assigned to any new or existing standard within the kit.

The previously defined standards retain their association to the deleted connector name. To reassign standards to a new connector family name, use SENS:CORR:COLL:CKIT:CONN:SNAMe.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SENS : CORR : COLL : CKIT : CONN : DEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Syntax</td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SENSe:CORRection:COLLect:CKIT:CONNector:FNAME <name>

(Read-Write) Replaces the primary connector family name from the selected kit with a new connector family name. The connector family name is replaced in all standards in the kit that share that name. The PNA allows multiple connector families for each kit. If a kit includes multiple connector families, only the first listed (primary) connector family name is replaced. Use the query form of this command to return the name of the primary connector family.

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;name&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples</th>
<th>SENS : CORR : COLL : CKIT : CONN : FNAME 'MYPSC35'</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sense:correction:collect:ckit:connector:name 'My Type N'</td>
<td></td>
</tr>
</tbody>
</table>

| Default | Not Applicable |

SENSe:CORRection:COLLect:CKIT:CONNector:SNAMe <family>,<gender>,<port>
Assigns a family name to the currently selected standard from the currently selected kit. Specify each port of a 2-port standard individually. Use the query form of this command to read the connector family name assigned to the current standard. The name is not assigned unless the connector family name is previously defined within the selected kit.

**Parameters**

- **<family>** String. Connector family name.
- **<gender>** Connector gender. Choose from:
  - MALE
  - FEMALE
  - NONE
- **<port>** Number of the connector port to be assigned the connector family name. 2-port standards such as a thru line must be assigned separately. It is not relevant which connector is port 1 or port 2.
  - 1 Specifies a 1-port standard or the first port of a 2-port standard.
  - 2 Specifies the second port of a 2-port standard.

**Examples**

```
SENS:CORR:COLL:CKIT:CONN:SNAME "Type-N (50)", MALE, 1
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:CONNector:SNAME?
```

**Return Type** String

**Default** Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:DESCRIPTION <string>**

(Read-Write) Modifies the cal kit description field of the selected kit. This description appears in the [Edit PNA Cal Kit dialog box](#).

**Parameters**

- **<string>** Description of the cal kit. Limited to 50 characters.

**Examples**

```
SENS:CORR:COLL:CKIT:DESC "My New CalKit"
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:DESCRIPTION?
```

**Return Type** String

**Default** Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:INFormation? <module>[,char]**
(Read Only) Reads characterization information from an ECal module.

Parameters

<module> Specifies which ECal module to read from. Choose from:
- ECAL1
- .through.
- ECAL8

[char] Optional argument.

Specifies which characterization within the ECal module to read information from. If this argument is not used, the default is CHAR0. CHAR1 through CHAR5 are for user characterizations that may have been written to the module by the User Characterization feature on the PNA. Choose from:
- CHAR0 Factory characterization (data that was stored in the ECal module by Agilent)
- CHAR1 User characterization #1
- CHAR2 User characterization #2
- CHAR3 User characterization #3
- CHAR4 User characterization #4
- CHAR5 User characterization #5

Examples

SENS:COR:COLL:CKIT:INF? ECAL4
sense:correction:collect:ckit:information? ecal2, char1

Example return string:

ModelNumber: 85092-60007, SerialNumber: 01386, ConnectorType: N5FN5F, PortAConnector: Type N (50) female, PortBConnector: Type N (50) female, MinFreq: 30000, MaxFreq: 9100000000, NumberOfPoints: 250, Calibrated: July 4 2002

Return Type Character

Default Not Applicable

SENSe:CORRection:COLLect:CKIT:NAME <name>
(Read-Write) Sets a name for the selected calibration kit.

**Parameters**

- `<name>`: Calibration Kit name. Any string name, can include numerics, period, and spaces; any length (although the dialog box display is limited to about 30 characters).

**Examples**

```
SENSe:CORRection:COLLect:CKIT:NAME 'MYAPC35'
sense:correction:collect:ckit:name 'mytypen'
```

**Query Syntax**

`SENSe:CORRection:COLLect:CKIT:NAME?`

**Return Type**

String

**Default**

Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:OLAbel<class> <name>**

(Read-Write) Sets the label for the calibration class designated by `<class>`. The label is used in the prompts for connecting the calibration standards associated with that `<class>`.

**Parameters**

- `<class>`: Number of the calibration class. Choose a number between: 1 and 18. The `<class>` numbers are associated with the following calibration Classes:

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S11A Reflection standard</td>
</tr>
<tr>
<td>2</td>
<td>S11B Reflection standard</td>
</tr>
<tr>
<td>3</td>
<td>S11C Reflection standard</td>
</tr>
<tr>
<td>4</td>
<td>S21T Thru/Delay standard</td>
</tr>
<tr>
<td>5</td>
<td>S22A Reflection standard</td>
</tr>
<tr>
<td>6</td>
<td>S22B Reflection standard</td>
</tr>
<tr>
<td>7</td>
<td>S22C Reflection standard</td>
</tr>
<tr>
<td>8</td>
<td>S12T Thru/Delay standard</td>
</tr>
</tbody>
</table>

**3-port analyzers only**

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>S33A Reflection standard</td>
</tr>
<tr>
<td>10</td>
<td>S33B Reflection standard</td>
</tr>
<tr>
<td>11</td>
<td>S33C Reflection standard</td>
</tr>
<tr>
<td>12</td>
<td>S32T Thru/Delay standard</td>
</tr>
</tbody>
</table>
13 S23T  Thru/Delay standard
14 S31T  Thru/Delay standard
15 S13T  Thru/Delay standard

**TRL Calibrations**

16 TRL "T"  Thru standard
17 TRL "R"  Reflect standard
18 TRL "L"  Line standard

<name>  Label for the calibration class. Must be enclosed in quotes. Any string between 1 and 12 characters long. Cannot begin with a numeric.

**Examples**

SENS:CORR:COLL:CKIT:OLAB3 'LOADS'
sense:correction:collect:ckit:olabel4 'Thru'

**Return Type**  String

**Default**  Not Applicable

**SENSe:CORRection:COLLect:CKIT:OLIST[class]?**

*(Read-only)* Returns seven values of standards that are assigned to the specified class.

**Parameters**

<class>  Number of the calibration class to be queried. The <class> numbers are associated with the following calibration Classes:

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S11A Reflection standard</td>
</tr>
<tr>
<td>2</td>
<td>S11B Reflection standard</td>
</tr>
<tr>
<td>3</td>
<td>S11C Reflection standard</td>
</tr>
<tr>
<td>4</td>
<td>S21T Thru/Delay standard</td>
</tr>
<tr>
<td>5</td>
<td>S22A Reflection standard</td>
</tr>
<tr>
<td>6</td>
<td>S22B Reflection standard</td>
</tr>
<tr>
<td>7</td>
<td>S22C Reflection standard</td>
</tr>
<tr>
<td>8</td>
<td>S12T Thru/Delay standard</td>
</tr>
</tbody>
</table>
3-port analyzers ONLY
4-port analyzers use S11 and S22 classes (see example program)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>S33A</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>10</td>
<td>S33B</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>11</td>
<td>S33C</td>
<td>Reflection standard</td>
</tr>
<tr>
<td>12</td>
<td>S32T</td>
<td>Thru/Delay standard</td>
</tr>
<tr>
<td>13</td>
<td>S23T</td>
<td>Thru/Delay standard</td>
</tr>
<tr>
<td>14</td>
<td>S31T</td>
<td>Thru/Delay standard</td>
</tr>
<tr>
<td>15</td>
<td>S13T</td>
<td>Thru/Delay standard</td>
</tr>
</tbody>
</table>

**TRL Calibrations**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>TRL &quot;T&quot;</td>
<td>Thru standard</td>
</tr>
<tr>
<td>17</td>
<td>TRL &quot;R&quot;</td>
<td>Reflect standard</td>
</tr>
<tr>
<td>18</td>
<td>TRL &quot;L&quot;</td>
<td>Thru standard</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:CORR:COLL:CKIT:OLIST8?
```

Always returns 7 standard numbers. Unassigned standards return 0

**Return Type**

Numeric; returns the <class> number of the selected standard.

**Default**

Not Applicable


*(Read-Write)* Sets a standard number to a calibration class. Does NOT set or dictate the order for measuring the standards. For more information, see Assigning Standards to a Calibration Class

**Parameters**

- `<class>`: Number of the calibration class that is assigned to `<standard>`. Choose a number between: 1 and 18. The `<class>` numbers are associated with the following calibration Classes:
<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S11A</td>
</tr>
<tr>
<td>2</td>
<td>S11B</td>
</tr>
<tr>
<td>3</td>
<td>S11C</td>
</tr>
<tr>
<td>4</td>
<td>S21T</td>
</tr>
<tr>
<td>5</td>
<td>S22A</td>
</tr>
<tr>
<td>6</td>
<td>S22B</td>
</tr>
<tr>
<td>7</td>
<td>S22C</td>
</tr>
<tr>
<td>8</td>
<td>S12T</td>
</tr>
</tbody>
</table>

3-port analyzers ONLY

4-port analyzers use S11 and S22 classes *(see example program)*

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>S33A</td>
</tr>
<tr>
<td>10</td>
<td>S33B</td>
</tr>
<tr>
<td>11</td>
<td>S33C</td>
</tr>
<tr>
<td>12</td>
<td>S32T</td>
</tr>
<tr>
<td>13</td>
<td>S23T</td>
</tr>
<tr>
<td>14</td>
<td>S31T</td>
</tr>
<tr>
<td>15</td>
<td>S13T</td>
</tr>
</tbody>
</table>

**TRL Calibration**

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>TRL &quot;T&quot;</td>
</tr>
<tr>
<td>17</td>
<td>TRL &quot;R&quot;</td>
</tr>
<tr>
<td>18</td>
<td>TRL &quot;L&quot;</td>
</tr>
</tbody>
</table>

*2082*
Standard number to be assigned to the class; Choose a standard between 1 and 8. One standard is mandatory; up to six additional standards are optional.

**Examples**
Assigns standard 3 to S11A class:

```
SENS:CORR:COLL:CKIT:ORD1  3
```

Assigns standard 2 and 5 to S21T class:

```
sense:correction:collect:ckit:order4  2,5
```

**Query Syntax**
SENSe:CORRection:COLLect:CKIT:ORDer<class>?

'Returns only the first standard assigned to the specified class. To query the remaining standards, use `SENSe:CORRection:COLLect:CKIT:OLIST[1-15]'?

**Return Type**
Numeric

**Default**
Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:RESet <num> - Superseded**

This command is replaced by `Sens:Corr:Ckit:Init`.

*(Write-only)* Resets the selected calibration kit to factory default definition values.

**Parameters**

- `<num>`: The number of the calibration kit to be reset. Choose any integer between 1 and 8

**Examples**

```
SENS:CORR:COLL:CKIT:RESet 1
sense:correction:collect:ckit:reset 4
```

**Query Syntax**
Not Applicable

**Default**
Not Applicable

---

**SENSe<cnum>:CORRection:COLLect:CKIT[:SELect] <num>**
(Read-Write) Selects (makes active) a calibration kit for **performing** a calibration or for **modifying** standards. All subsequent "CKIT" commands that are sent apply to this selected calibration kit. Select a calibration standard using `SENS:CORR:COLL:CKIT:STAN <num>`

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.

- `<num>` The number of the calibration kit. Choose from:
  - Use `SENS:CORR:COLL:CKIT:RESet` to restore Cal Kits to default values.

**Name**

1. User Defined 1
2. User Defined 2
3. User Defined 3
   - "
   - "
   - 48 User Defined 48
   - 49 User Defined 49
   - 50 User Defined 50
   - 99 ECAL module

**Examples**

```plaintext
SENS:CORR:COLL:CKIT 2
sense2:correction:collect:ckit:select 7
```

**Query Syntax**

`SENS<cn>USe<cn>:CORR:COLL:CKIT?`

**Return Type** Numeric

**Default** Last kit selected

---

`SENS:CORR:COLL:CKIT:STANdard:C0 <num>`
(Read-Write) Sets the C0 value (the first capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

**Parameters**

|<num>| Value for C0 in femtofarads (1E-15) |

**Examples**
The following commands set C0 = 15 femtofarads:

```
SENS:CORR:COLL:CKIT:STAN:C0 15
sense:correction:collect:ckit:standard:c0 15
```

**Query Syntax**
SENSe:CORRection:COLLect:CKIT:STANdard:C0?

**Return Type**
Numeric

**Default**
Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard:C1 <num>**

(Read-Write) Sets the C1 value (the second capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

**Parameters**

|<num>| Value for C1 |

**Examples**
The following two commands set C1 = 15:

```
SENS:CORR:COLL:CKIT:STAN:C1 15
sense:correction:collect:ckit:standard:c1 15
```

**Query Syntax**
SENSe:CORRection:COLLect:CKIT:STANdard:C1?

**Return Type**
Numeric

**Default**
Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard:C2 <num>**
**SENSa:CORRection:COLLect:CKIT:STANdard:C2 <num>**

*(Read-Write)* Sets the C2 value (the third capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

**Parameters**

- `<num>` Value for C2.

**Examples**
The following two commands set C2:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

**Query Syntax**
SENSe:CORRection:COLLect:CKIT:STANdard:C2?

**Return Type**
Numeric

**Default**
Not Applicable

**SENSa:CORRection:COLLect:CKIT:STANdard:C3 <num>**

*(Read-Write)* Sets the C3 value (the fourth capacitance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

**Parameters**

- `<num>` Value for C3.

**Examples**
The following two commands set C3

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:CORR:COLL:CKIT:STAN:C3 15</td>
<td>Value for C3</td>
</tr>
<tr>
<td>sense:correction:collect:ckit:standard:c3 15</td>
<td>Value for C3</td>
</tr>
</tbody>
</table>

**Query Syntax**
SENSe:CORRection:COLLect:CKIT:STANdard:C3?

**Return Type**
Numeric

**Default**
Not Applicable

**SENSa:CORRection:COLLect:CKIT:STANdard:CHARacter <char>**
Sets the media type of the selected calibration standard.

**Parameters**

- **<char>** Media type of the standard. Choose from:
  - Coax - Coaxial Cable
  - Wave - Waveguide

**Examples**

SENS:CORR:COLL:CKIT:STAN:CHAR COAX
sense:correction:collect:ckit:standard:character wave

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:STANdard:CHARacter?

**Return Type**

Numeric

**Default**

Coax

---

**SENSe:CORRection:COLLect:CKIT:STANdard:DELay <num>**

Sets the electrical delay value for the selected standard.

**Parameters**

- **<num>** Electrical delay in picoseconds

**Examples**

The following two commands set delay to 50 picoseconds

SENS:CORR:COLL:CKIT:STAN:DEL 50e-12
sense2:correction:collect:ckit:standard:delay 50ps

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:STANdard:DELay?

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard:FMAX <num>**
(Read-Write) Sets the maximum frequency for the selected standard.

**Parameters**

<num> Maximum frequency in Hertz.

**Examples**

```
SENS:CORR:COLL:CKIT:STAN:FMAX 9e9
sense:correction:collect:ckit:standard:fmax 9Ghz
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:FMAX?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard:FMIN <num>**

(Read-Write) Sets the minimum frequency for the selected standard.

**Parameters**

<num> Minimum frequency in Hertz.

**Examples**

```
SENS:CORR:COLL:CKIT:STAN:FMIN 1e3
sense:correction:collect:ckit:standard:fmin 1khz
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:FMIN?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard:IMPedance <num>**

(Read-Write) Sets the characteristic impedance for the selected standard.

**Parameters**

<num> Impedance in Ohms

**Examples**

```
SENS:CORR:COLL:CKIT:STAN:IMP 75
sense:correction:collect:ckit:standard:impedance 50.3
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:IMPedance?
```

**Return Type**

Numeric

**Default**

50
SENSe:CORRection:COLLect:CKIT:STANdard:L0 <num>

(Read-Write) Sets the L0 value (the first inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for L0 in femtohenries (1E-15)

Examples

The following two commands set L0=15 femtohenries:

SENS:CORR:COLL:CKIT:STAN:L0 15
sense:correction:collect:ckit:standard:l0 15

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:L0?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:L1 <num>

(Read-Write) Sets the L1 value (the second inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at www.Agilent.com.

Parameters

<num> Value for L1.

Examples

The following two commands set L1=15:

SENS:CORR:COLL:CKIT:STAN:L1 15
sense:correction:collect:ckit:standard:l1 15

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard:L1?

Return Type

Numeric

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard:L2 <num>
**SENSe:CORRection:COLLect:CKIT:STANdard:L2 <num>**

*(Read-Write)* Sets the L2 value (the third inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at [www.Agilent.com](http://www.Agilent.com).

**Parameters**

- `<num>` Value for L2.

**Examples**
The following two commands set L2=15:

```plaintext
SENS:CORR:COLL:CKIT:STAN:L2 15
sense:correction:collect:ckit:standard:l2 15
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:L2?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard:L3 <num>**

*(Read-Write)* Sets the L3 value (the fourth inductance value) for the selected standard. For a detailed discussion of this value, search for App Note 8510-5B at [www.Agilent.com](http://www.Agilent.com).

**Parameters**

- `<num>` Value for L3.

**Examples**
The following two commands set L3=15:

```plaintext
SENS:CORR:COLL:CKIT:STAN:L3 15
sense:correction:collect:ckit:standard:l3 15
```

**Query Syntax**

```
SENSe:CORRection:COLLect:CKIT:STANdard:L3?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard:LABel <name>**
**SENSe:CORRection:COLLect:CKIT:STANdard:LABel**

*(Read-Write)* Sets the label for the selected standard. The label is used to prompt the user to connect the specified standard.

**Parameters**

- `<name>`: Label for the standard; Must be enclosed in quotes. Any string between 1 and 12 characters long. Cannot begin with a numeric.

**Examples**

- SENS:CORR:COLL:CKIT:STAN:LAB 'OPEN'
- sense:correction:collect:ckit:standard:label 'Short2'

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:STANdard:LABel?

**Return Type**

String

**Default**

Not Applicable

**SENSe:CORRection:COLLect:CKIT:STANdard:LOSS** `<num>`

*(Read-Write)* Sets the insertion loss for the selected standard.

**Parameters**

- `<num>`: Insertion loss in Gohms / sec. (GigaOhms per second of electrical delay)

**Examples**

- SENS:CORR:COLL:CKIT:STAN:LOSS 3.5e9
- sense:correction:collect:ckit:standard:loss 3

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:STANdard:LOSS?

**Return Type**

Numeric

**Default**

Not Applicable

**SENSe:CORRection:COLLect:CKIT:STANdard:REMove**

*(Write only)* Deletes the selected standard from the selected cal kit.

**Examples**


**Default**

Not Applicable

**SENSe:CORRection:COLLect:CKIT:STANdard:SDEScriptio**n `<string>`
(Read-Write) Modifies the description of the selected standard of the selected kit. This description appears in the edit kit dialog box.

Parameters

<string> Description of the standard.

Examples


Query Syntax

SENSe:CORRection:COLLect:STANdard:SDEScription?

Return Type

String

Default

Not Applicable

SENSe:CORRection:COLLect:CKIT:STANdard[:SELECT] <num>

(Read-Write) Selects the calibration standard. All subsequent "CKIT" commands to modify a standard will apply to the selected standard. Select a calibration kit using SENs:CORR:COLL:CKIT:SEL.

Parameters

<num> Number of the standard. Choose any number between: 1 and 30

Examples

SENS:CORR:COLL:CKIT:STAN 3
sense:correction:collect:ckit:standard:select 8

Query Syntax

SENSe:CORRection:COLLect:CKIT:STANdard[:SELect]?

Return Type

Numeric

Default

1

SENSe:CORRection:COLLect:CKIT:STANdard:TYPE <char>
(Read-Write) Sets the type for the selected standard.

**Parameters**

<char>  Choose from:

- OPEN
- SHORT
- LOAD
- SLOAD (sliding load)
- THRU (through)
- ARBI (arbitrary)

**Examples**

SENS:CORR:COLL:CKIT:STAN:TYPE LOAD
sense:correction:collect:ckit:standard:type short

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:STANdard:TYPE?

**Return Type**  Character

**Default**  Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard:TZReal <num>**

(Read-Write) Sets the TZReal component value of the Terminal Impedance for the selected standard.

**Note:** Only applicable when the Standard Type is set to ARBI

**Parameters**

<num>  Value for TZReal in Ohms

**Examples**

The following commands set TZReal=15 Ohms:

SENS:CORR:COLL:CKIT:STAN:TZReal 15
sense:correction:collect:ckit:standard:TZReal 15

**Query Syntax**

SENSe:CORRection:COLLect:CKIT:STANdard:TZReal?

**Return Type**  Numeric

**Default**  Not Applicable

---

**SENSe:CORRection:COLLect:CKIT:STANdard:TZImag <num>**
(Read-Write) Sets the TZImag component value of the Terminal Impedance for the selected standard.

**Note:** Only applicable when the Standard Type is set to **ARBI**

### Parameters

- `<num>` Value for TZImag in Ohms

### Examples

The following two commands set TZImag=15 Ohms:

- `SENS:CORR:COLL:CKIT:STAN:TZImag 15`
- `sense:correction:collect:ckit:standard:TZImag 15`

### Query Syntax

- `SENSe:CORRection:COLLect:CKIT:STANdard:TZImag?`

### Return Type

- Numeric

### Default

- Not Applicable

---

Last Modified:

- 19-Sep-2007   Added missing `<num>` argument
Sense:Correction:Cset Commands

Performs actions on calibration sets.

|SENSe:CORRection:CSET

- ACTivate
- CATalog?
- COPY
- CREate
- DATA
- DELete
- DESCription
- ETERm
- GUID
- NAME
- [SELect]
- SAVE
- STANdard

| TSET
|   | ALLPorts?
|   | TYPE?

| TYPE
|   | CATalog?

Click on a blue keyword to view the command details.

See Also

- Creating Cal Sets
SENSएनबीयसःCORRection:CSET:ACTivate <string>, <bool>

This command replaces SENS:CORR:CSET:GUID

(Read-Write) Selects and applies a Cal Set to the specified channel.

Use SENS:CORR:CSET:CAT? to list the Cal Sets.

Parameters

<cnun>  Any existing channel number. If unspecified, value is set to 1

<string>  Cal Set to make active. Specify the Cal Set by GUID or Name. Use SENS:CORR:CSET:CAT? to list the available Cal Sets in either format.

<bool>  Should the Cal Set stimulus values be applied to the channel. Choose from:

ON (1)  Apply the Cal Set stimulus values to the channel.

OFF (0)  Do NOT apply the Cal Set stimulus values. If the Cal Set stimulus values do not match the channel stimulus values, then the following will occur:

- If interpolation is ON, then interpolation will be attempted. This may fail if the channel frequency is outside the range of the Cal Set.
- If interpolation is OFF, the selection will be abandoned and an error is returned:

Examples

SENS:CORR:CSET:ACT "My2Port",1

sense:correction:cset:activate? name
returns
"My2Port"

Query Syntax  SENSएनबीयसःCORRection:CSET:ACTivate? [GUID][NAME]

Returns the name of the Cal Set that is applied to the specified channel. Choose from GUID or NAME to specify which string is returned. If unspecified, the GUID of the Cal Set is returned. If no Cal Set is applied to the specified channel, then "No Calset Selected" is returned.

Return Type  String

Default  Not Applicable
SENSe:CORRection:CSET:CATalog? [char]

(Read-only) Returns a list of Cal Sets.

Parameters

<char> Optional argument. The list is returned in one of the following formats. Both return comma-separated string lists.

GUID Cal Sets are listed by GUID (Default if unspecified).

NAME Cal Sets are listed by Name

Examples

SENSe:CORRection:CSET:CAT? 

'Returns: 
{FD6F863E-9719-11d5-8D6C-00108334AE96}, {1B03B2CE-971A-11d5-8D6C-00108334AE96} 

SENSe2:correction:csset:catalog? name

Default Not Applicable

SENSe<cnm>:CORRection:CSET:COPY <string>

(Write-only) Creates a new Cal Set and copies the current Cal Set data into it. Use this command to manipulate data on a Cal Set without corrupting the original cal data.

Parameters

<cnm> Channel number using the Cal Set to be copied. If unspecified, value is set to 1

<string> Name of the new Cal Set.

Examples

SENSe2:CORRection:CSET:COPY 'My2Port'

Query Syntax Not Applicable

Default Not Applicable

SENSe<cnm>:CORRection:CSET:CREate [name]
(Write-only) Creates an empty Cal Set and attaches it to the specified channel. This command is
ONLY necessary before remotely filling the Cal Set with error term data. (For Advanced Users).

A Cal Set is automatically created, applied to the channel, and saved at the completion of a guided cal
according to the preference setting SENS:CORR:PREF:CSET:SAVE.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `[name]` Optional argument. Name of the Cal Set. Spaces or punctuation are NOT
  allowed. If unspecified, a unique name is chosen in the form "Calset_N" where
  N is a unique number.

Examples

- SENS:CORR:CSET:CRE 'My2Port'

Query Syntax

- Not Applicable

Default

- Not Applicable

SENSe<cnm>:CORRection:CSET:DATA <eterm, portA, portB>,[<param>,,<block>]

(Read-Write) Read or Write a specific error term from/to the Cal Set currently attached to the
specified channel. (For Advanced Users). The command can be used only for the error terms listed.
See SENS:CORR:CSET:ETERM to get and put error term data using a string argument for all error
terms..

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<eterm, portA, portB>` Error Term, Port pair of the specified error term.

Although not all error terms use two port numbers, two are required by the
PNA in all cases. Each port number must be between 1 and the number of ports
on the PNA.

- `EDIR` - directivity

  portA: the port at which directivity is measured.

  portB: Not used, but must be a valid PNA port number.

- `ESRM` - source match

  portA: the port at which source match is measured.

  portB: Not used, but must be a valid PNA port number.
ERFT - reflection tracking

portA: the port at which reflection tracking is measured.
portB: Not used, but must be a valid PNA port number.

ELDM - load match

portA: the port at which load match is measured.
portB: the source port.

Load match is measured with a cable connected between the measured port (portA) and the source port (portB).

The cal system requires that the complete matrix of loadmatch arrays be filled. In most cases you can measure loadmatch once at a port, driven by any other port. Then use that data for all variations of the receive port. (The exception is the 3-port PNA models, which requires the loadmatch-measured port to be driven by every other port.)

For example: Measure the loadmatch at port2 while driving port1. Then upload this same data to the following arrays:

   ELDM,2,1,<data>
   ELDM,2,3,<data>
   ELDM,2,4,<data>

ETRT - transmission tracking

portA: the receive port
portB: the source port for this measurement

EXTLK - crosstalk

portA: the receive port
portB: the source port for this measurement

ERSPT - response tracking.

portA: Not used, but must be a valid PNA port number.
portB: Not used, but must be a valid PNA port number.
ERSPI - response isolation.

portA: Not used, but must be a valid PNA port number.

portB: Not used, but must be a valid PNA port number.

<param> <string> - required ONLY when Eterm is response tracking (ERSPT) or response isolation (ERSPI). Specify the S-parameter, ratio, or unratioed measurement for which the Eterm applies.

Logical receiver notation is allowed. Ratioed measurements do not require source port to be specified.

A full 4-port calibration requires the following terms be uploaded:

| PORT B |
|--------|--------|--------|--------|
| 1      | 2      | 3      | 4      |
| EDIR,1,1 | ELDM,1,2 | ELDM,1,3 | ELDM,1,4 |
| ERFT,1,1 | ETRT,1,2 | ETRT,1,3 | ETRT,1,4 |
| ESRM,1,1 | EXTLK,1,2 | EXTLK,1,3 | EXTLK,1,4 |
| 2      | 3      | 4      | 5      |
| EDIR,2,2 | ELDM,2,3 | ELDM,2,4 | ELDM,3,4 |
| ERFT,2,2 | ETRT,2,3 | ETRT,2,4 | ETRT,3,4 |
| ESRM,2,2 | EXTLK,2,3 | EXTLK,2,4 | EXTLK,3,4 |
| 3      | 4      | 5      | 6      |
| ELDM,3,2 | EDIR,3,3 | ELDM,3,4 | ELDM,4,4 |
| ETRT,3,2 | ERTT,3,3 | ERTT,3,4 | ERTT,4,4 |
| EXTLK,3,2 | ESRM,3,3 | ESRM,3,4 | ESRM,4,4 |

Reflection terms

Transmission terms

Examples

SENS:CORR:CSET:DATA EXTLK,3,1 'cross talk between port 3 receiver and port 1 source.

SENS:CORR:CSET:DATA ERSPT,1,1, "a3/b4" 'response tracking term for ratioed measurement of port 4 test receiver over port 3 reference receiver.
SENSe:CORRection:CSET:DELete <string>
(Write-only) Deletes a Cal Set from the set of available Cal Sets. This method immediately updates the Cal Set file on the hard drive. If the Cal Set is currently being used by a channel or does not exist, this request will be denied and an error is returned.

Parameters
<string> Cal Set to be deleted. Specify the Cal Set by GUID or Name. Use SENSe:CORR:CSET:CAT? to list the available Cal Sets in either format.

Examples
sense2:correction:cset:delete 'MyCalSet'

SENSe:cnum>:CORRection:CSET:DESCription <string>
(Read-Write) Sets or returns the descriptive string assigned to the selected Cal Set. Change this string so that you can easily identify each Cal Set. Apply and select the Cal Set using SENSe:CORR:CSET:ACT.

Parameters
<cnum> Any existing channel number. If unspecified, value is set to 1
<string> The descriptive string associated with the currently-selected Cal Set

Examples
SENSe:CORR:CSET:DESC 'MyCalSet'
sense2:correction:cset:description 'thisCalSet'

SENSe:cnum>:CORRection:CSET:ETERm <string>,<r, i [r,i]...>
(Read-Write) Sets or returns error term data for all PNA measurements.

Parameters

<cnump> Any existing channel number. If unspecified, value is set to 1
<string> (String) Error term to read or write. The error term is specified using the EXACT case-sensitive string displayed in the Cal Set Viewer utility. See SENS:CORR:CSET:DATA for a description of port numbers.

The following Noise figure error terms are listed for convenience:

- **RcvNoiseCorr_m_n** Noise correlation matrix of the noise receiver (a 2x2 complex matrix). The row and column indices m and n range from 1 to 2.
- **RcvT_m_n** T-matrix of the noise receiver (a 2x2 complex matrix). The row and column indices m and n range from 1 to 2.
- **GammaTuner_n** Reflection coefficient for impedance state n of the embedded noise tuner (Ecal module) in the port 1 source path. For the Agilent 4691 family of Ecal modules, n can range from 1 to 7.

<r,i> Real and Imaginary data pairs.

Examples

```
SENS:CORR:CSET:ETERM "Directivity(1,1)", 0.237,-1.422, 0.513, 0.895  ' set directivity(source error term for 2 points
SENS:CORR:CSET:ETERM? "Directivity(1,1)"   'read
```

Query Syntax  SENSE<cnump>:CORRection:CSET:ETERm? <string>

Return Type  Block data

Default  Not Applicable

SENSe<cnump>:CORRection:CSET:GUID <string>  Superseded
This command is replaced by `SENS:CORR:CSET:ACTivate`.

(Read-Write) Selects the Cal Set identified by the string parameter (GUID) and applies it to the specified channel.

- A Cal Set cannot be selected for a channel which is not ON.
- If the stimulus settings of the selected Cal Set differ from those of the selected channel, the instrument will automatically change the channel's settings to match the Cal Set.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<string>` GUID of the desired Cal Set. The curly brackets and hyphens must be included.

**Examples**

```
```

**Query Syntax**

`SENSe<cnum>:CORRection:CSET:GUID?`

Returns the GUID of the currently-selected Cal Set for the specified channel.

**Return Type** String

**Default** Not Applicable

---

`SENSe<cnum>:CORRection:CSET:NAME <string>`

(Read-Write) Sets or queries the name of the Cal Set currently applied to the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<string>` Name of the Cal Set. Spaces or punctuation NOT allowed.

**Examples**

```
SENS:CORR:CSET:NAME 'MyCalSet'
sense2:correction:cset:name 'thisCalSet'
```

**Query Syntax**

`SENSe<cnum>:CORRection:CSET:NAME?`

**Return Type** String

**Default** Not Applicable

---

`SENSe<cnum>:CORRection:CSET[:SELect] <char>` Superseded
This command is replaced by **MMEM:LOAD**

(Read-Write) Restores a state file from memory. The file name is "CSETx.cst" where x is the user number assigned to `<char>`, and .cst specifies a cal set and instrument state. This is not the same syntax as a file saved through the default choices from the front panel, which is "at00x.cst". For more information on the file naming syntax, see the **MMEMory** subsystem. Learn more about **Instrument/Cal States**.

**Note:** This command does NOT select a Cal Set for a channel. To select a Cal Set, use **SENS:CORR:CSET:ACTivate**

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
  - **DEF** - Presets the analyzer
  - **USER01** - Restores User01 calibration data
  - **USER02** - Restores User02 calibration data
  - through...
  - **USER10** - Restores User10 calibration data

**Examples**

```
SENS:CORR:CSET DEF
sense2:correction:cset:select user02
```

**Query Syntax**

```
SENSe<cnum>:CORRection:CSET[:SELect]?
```

**Return Type**

Character

**Default**

DEF

**SENSe<cnum>:CORRection:CSET:SAVE <char>**
This command is NOT necessary after completion of a calibration. A Cal Set is automatically created, applied to the channel, and saved at the completion of a guided cal according to the preference setting SENS:CORR:PREF:CSET:SAVE.

(Read Write)

Saves the channel's Cal Set to the PNA hard drive. For example, use this command after writing data to a Cal Set using SENS:CORR:CSET:DATA (For Advanced Users).

The file name is saved as "CSET.x.cst" where x is the user number assigned to <char>, and .cst specifies a Cal Set and instrument state. This is not the same syntax as a file saved through the default choices from the front panel, which is "at00x.cst". For more information on the file naming syntax, see the MMEMory subsystem. Learn more about Instrument/Cal States.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:
  - USER01
  - USER02...
  - USER10

**Examples**

```
SENS:CORR:CSET:SAVE USER03
sense2:correction:cset:save user09
```

**Query Syntax**

SENSe<cnum>:CORRection:CSET:SAVE?
Queries the last correction set saved.

**Return Type**

Character

**Default** Not applicable

```
SENSe<cnum>:CORRection:CSET:STANdard <string>
```
(Read-Write) Sets or returns standard data. Standard data is available for Unguided Cals ONLY.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<string>` (String) Cal standard acquisition data to read or write. The standard is specified using the EXACT case-sensitive string displayed in the Cal Set Viewer utility. See SENS:CORR:CSET:DATA for a description of port numbers.

**Examples**

```plaintext
SENS:CORR:CSET:STAN 'S11C(1,1)'
SENS:CORR:CSET:STAN? "S11C(1,1)"  "Read data"
```

**Query Syntax**

SENSe<cnum>:CORRection:CSET:STANdard? (string)

**Return Type**

Block data

**Default** Not Applicable

---

**SENSe:CORRection:CSET:TSET:ALLPorts? <cset>**

(Read-only) Reads the port mapping used for the specified Cal Set. The returned values are the physical ports. The POSITION of the returned values corresponds to the logical ports.

For example, with an N44xx test set, if the returned string is "PNA 1,TS 2,PNA 2, TS 4" this means:

- PNA 1 is assigned to logical port 1
- TS 2 is assigned to logical port 2
- PNA 2 is assigned to logical port 3
- TS 4 is assigned to logical port 4

**Parameters**

- `<cset>` (String) Name or GUID of the Cal Set. Use SENS:CORR:CSET:CAT? to read the list of available Cal Set names or GUIDs.

**Examples**

```plaintext
SENS:CORR:CSET:TSET:ALLP? "MyCalSet"
```

**Return Type**

String

**Default** Not Applicable

---

**SENSe:CORRection:CSET:TSET:TYPE? <cset>**
(Read-only) Reads the test set type (model) used for the specified Cal Set.

**Parameters**

<cset>  
(String) Name or GUID of the Cal Set. Use SENS:CORR:CSET:CAT? to read the list of available Cal Set names or GUIDs.

**Examples**

SENS:CORR:CSET:TSET:TYPE? "MyCalSet"
returns "N44xx"

**Return Type**  
String

**Default**  
Not Applicable

---

**SENS<ch>:CORRection:CSET:TYPE:CATalog? [format]**

(Read-only) Query the Cal Types available in the selected Cal Set. The output is a comma separated list of Guids or a Cal Type names. Learn more about applying Cal Types using SCPI.

Use CALC:CORR:TYPE to apply a Cal Type.

**Parameters**

<ch>  
Any existing channel number. If unspecified, value is set to 1

[format]  
(Optional) Format of the output of cal types. choose from:

- **NAME** - (default) returns a list of cal type string names.
- **GUID** - returns a list of cal type GUIDs

**Examples**

SENS:CORR:CSET:TYPE:CAT? NAME
SENS:CORR:CSET:TYPE:CAT?

**Return Type**  
String

**Default**  
Not Applicable

---

Last modified:

- 6-Mar-2008  
  Added CSET Delete by Name (8.0)
- 5-Mar-2008  
  Added Noise ETerm and Std commands (8.0)
- 9/12/06  
  MQ Added TSET commands for multiport.
Sense:Correction:Extension Commands

Performs and applies Port Extensions.

SENSe<cnum>:CORRection:EXTension:AUTO:CONFig <char>

(Read-Write) Sets the frequencies used to calculate Automatic Port Extension. Learn more about calculating Automatic Port Extension.

Parameters

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<char>`  Frequencies to be used:

  - **CSPN**  Use current frequency span.
  - **AMKR**  - Use active marker frequency.

Examples

```
SENSe:CORRection:EXTension:auto:config CSPN
SENSe:CORRection:EXTension:auto:config USPN
SENSe:CORRection:EXTension:auto:config AMKR
```

Query Syntax

```
SENSe<cnum>:CORRection:EXTension:AUTO:CONFig ?
```
SENSe\(<cnum>\):CORRection:EXTension:AUTO:DCOFfset <bool>

(Read-Write) Specifies whether or not to include DC Offset as part of automatic port extension. Learn more about Automatic DC Offset. Only allowed when SENS:CORR:EXT:AUTO:LOSS is set to ON.

Parameters

\(<cnum>\) Any existing channel number. If unspecified, value is set to 1
\(<bool>\) ON (or 1) - Includes DC Offset correction.

OFF (or 0) - Does NOT include DC Offset correction.

Examples

SENSe\(<cnum>\):CORR:EXT:AUTO:DCOF 1
sense2:correction:extension:auto:dcoffset off

Query Syntax

SENSe\(<cnum>\):CORRection:EXTension:AUTO:DCOFfset?

Return Type

Boolean

Default

OFF (0)

SENSe\(<cnum>\):CORRection:EXTension:AUTO:LOSS <bool>

(Read-Write) Specifies whether or not to include loss correction as part of automatic port extension. Learn more about Loss Compensation in port extension.

Parameters

\(<cnum>\) Any existing channel number. If unspecified, value is set to 1
\(<bool>\) ON (or 1) - Includes Loss correction.

OFF (or 0) - Does NOT include Loss correction.

Examples

SENSe\(<cnum>\):CORR:EXT:AUTO:LOSS 1
sense2:correction:extension:auto:loss off

Query Syntax

SENSe\(<cnum>\):CORRection:EXTension:AUTO:LOSS?

Return Type

Boolean

Default

OFF (0)

SENSe\(<cnum>\):CORRection:EXTension:AUTO:MEASure <char>
(Write-only) Measures either an OPEN or SHORT standard. When this command is sent, the PNA acquires the measurement with which to set automatic port extensions. Learn more about which standard to measure.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Standard to be measured. Choose from:
  - **OPEN** Measure OPEN standard
  - **SHORT** Measure SHORT standard

**Examples**

```plaintext
SENS:CORR:EXT:AUTO:MEAS OPEN
sense2:correction:extension:auto:measure short
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SENSe<cnum>:CORRection:EXTension:AUTO:PORT<n> <bool>**

(Read-Write) Enables and disables automatic port extensions on the specified port.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<n>` PNA Port number to enable or disable for automatic port extensions.
- `<bool>` ON (or 1) - Enable
  - OFF (or 0) - Disable

**Examples**

```plaintext
SENS:CORR:EXT:AUTO:PORT2 0
sense2:correction:extension:auto:port4 on
```

**Query Syntax**

`SENSe<cnum>:CORRection:EXTension:AUTO:PORT<n>?`

**Return Type**

Boolean

**Default**

All ports ON (enabled)

---

**SENSe<cnum>:CORRection:EXTension:AUTO:RESet**
(Write-only) Clears old port extension delay and loss data in preparation for acquiring new data. Send this command prior to sending a new series of SENS:CORR:EXT:AUTO:MEAS. If acquiring both OPEN and SHORT standards, do not send this command between those acquisitions.

**Parameters**

- `<cnum>`
  Any existing channel number. If unspecified, value is set to 1

**Examples**

```plaintext
SENS:CORR:EXT:AUTO:RES
sense2:correction:extension:auto:reset
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SENSe<cnum>:CORRection:EXTension:AUTO:STARt <value>**

(Read-Write) Set the start frequency for custom user span. Learn more about User Span.

**Parameters**

- `<cnum>`
  Any existing channel number. If unspecified, value is set to 1
- `<value>`
  User span start value. Must be within the frequency range of the active channel and less than the value set by SENS:CORR:EXT:AUTO:STOP.

**Examples**

```plaintext
SENS:CORR:EXT:AUTO:STAR 1E9
sense2:correction:extension:auto:start 200e6
```

**Query Syntax**

SENSe<cnum>:CORRection:EXTension:AUTO:STARt <value>?

**Return Type**

Numeric

**Default**

Start frequency of the current active channel.

---

**SENSe<cnum>:CORRection:EXTension:AUTO:STOP <value>**
(Read-Write) Set the stop frequency for custom user span. Learn more about User Span.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<value>` User span stop value. Must be within the frequency range of the active channel and greater than the value set by SENS:CORR:EXT:AUTO:STARt

**Examples**

```
SENS:CORR:EXT:AUTO:STOP 1E9
sense2:correction:extension:auto:stop 200e6
```

**Query Syntax**

```
SENSe<cnum>:CORRection:EXTension:AUTO:STOP <value>?
```

**Return Type**

Numeric

**Default**

Stop frequency of the current active channel.

---

**SENSe<cnum>:CORRection:EXTension:PORT<pnum>:FREQ<n> <value>**

(Read-Write) Sets and returns the frequency "Use" number (1|2) for the specified port number. Learn about Loss Compensation values.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number that will receive the freq/loss settings. If unspecified, value is set to 1.
- `<n>` Frequency "Use" number. Choose from 1 or 2. If unspecified, value is set to 1.
- `<value>` Frequency value. Choose a frequency within the frequency span of the PNA.

**Examples**

```
SENS:CORR:EXT:PORT1:FREQ1 10E9
sense2:correction:extension:port2:freq2:2E10
```

**Query Syntax**

```
SENSe<cnum>:CORRection:EXTension:PORT<pnum>:FREQ<n>?
```

**Return Type**

Numeric

**Default**

1 GHz

---

**SENSe<cnum>:CORRection:EXTension:PORT<pnum>:INCLude<n>:STATe <bool>**
(Read-Write) Sets and returns the ON/OFF state for the use of the **Loss** and **Freq** values for the specified port number.

**Learn about Loss Compensation values.**

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
- `<n>` Freq and Loss pair. Choose from 1 or 2. If unspecified, value is set to 1.
- `<value>` State of Freq and Loss values for port extension.

| 0 or OFF | Specified Freq and Loss values are OFF |
| 1 or ON  | Specified Freq and Loss values are ON |

**Examples**

```
SENS:CORR:EXT:PORT:INCL 0
sense2:correction:extension:port2:include:state on
```

**Query Syntax**

```
SENSe<cnum>:CORRection:EXTension:PORT<pnum>:INCLude:STATe?
```

**Return Type**

Boolean

**Default**

OFF

---

**SENSe<cnum>:CORR:EXT:PORT:<pnum>:LDC <value>**

(Read-Write) Sets and returns the Port Loss at DC value for the specified port number.

**Learn about Loss Compensation values.**

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port number to receive Loss value. If unspecified, value is set to 1.
- `<value>` Loss in dB. Choose a value between -90 and 90

**Examples**

```
SENS:CORR:EXT:PORT:LDC 1.5
sense2:correction:extension:port2:ldc .1
```

**Query Syntax**

```
SENSe<cnum>:CORR:EXT:PORT<pnum>:LDC?
```
**SENSe<cnum>:CORRection:EXTension:PORT<pnum>:LOSS<n> <value>**

(Read-Write) Sets and returns the Loss value for the specified port number.

**Learn about Loss Compensation values.**

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<pnum>`: Port Number that will receive the Freq/Loss settings. If unspecified, value is set to 1.
- `<n>`: Loss "Use" number. Choose from 1 or 2. If unspecified, value is set to 1.
- `<value>`: Loss in dB. Choose a value between -90 and 90

**Examples**

```
SENSe:CORR:EXT:PORT:LOSS1 1
SENSe:CORR:EXT:PORT:loss2 .1
```

**Query Syntax**

```
SENSe<cnum>:CORRection:EXTension:PORT<pnum>:LOSS<n>?  
```

**Return Type** Numeric

**Default** 0

---

**SENSe<cnum>:CORRection:EXTension:PORT<pnum>[[:TIME]] <num>**

---

Shepherd 100

SMA Female

SMA Female
(Read-Write) Sets the extension value at the specified port. Must also set SENS:CORR:EXT ON.

**Note:** This command affects ALL measurements on the specified channel.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pnum>` Port Number that will receive the extension. If unspecified, value is set to 1.
- `<num>` The port extension in seconds; may include suffix. Choose a number between: -1E18 and 1E18

**Examples**
- `SENS:CORR:EXT:PORT 2MS`  
  `sense2:correction:extension:port2 .00025`

**Query Syntax**
- `SENSe<cnum>:CORRection:EXTension:PORT<pnum> [:TIME]?

**Return Type** Numeric

**Default** 0

---

**SENSe<cnum>:CORRection:EXTension:RECeiver<Rnum>[[:TIME]] <num>** OBSOLETE

(Read-Write) This command has NO replacement and no longer works.

Sets the extension value at the specified receiver. Must also set SENS:CORR:EXT ON.

**Note:** Before using this command you must select a measurement using CALC:PAR:SEL. You can select one measurement for each channel.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<Rnum>` Number of the receiver that will receive the extension. If unspecified, value is set to 1
  Choose from:

  1 for Receiver A

  2 for Receiver B

- `<num>` The electrical length in seconds; may include suffix. Choose a number between:
  -10 and 10

**Examples**
- `SENS:CORR:EXT:REC 2MS`  
  `sense2:correction:extension:receiver2:time .00025`

**Query Syntax**
- `SENSe<cnum>:CORRection:EXTension:RECeiver<Rnum> [:TIME]?
SENSe<cnum>:CORRection:EXTension[:STATe] <ON | OFF>

(Read-Write) Turns port extensions ON or OFF.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<ON | OFF>`
  - ON (or 1) - turns port extensions ON.
  - OFF (or 0) - turns port extensions is OFF.

Examples

```
SENS:CORR:EXT ON
sense2:correction:extension:state off
```

Query Syntax

SENSe<cnum>:CORRection:EXTension[:STATe]?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF
Sense:Correction:Collect:Session Commands

The commands in this topic are common to perform both SMC and VMC calibrations. A calibration session is a term used to describe an instance of a SMC or VMC calibration. For more commands, see SESS:SMC and SESS:VMC.

Notes:

- Commands to read (STEP?) and describe (DESC?) each step are provided to facilitate a remote user interface.
- To perform a NON - SMC or VMC calibration, use either Guided or Mechanical commands.
- It is now possible to perform a remote Source Power Calibration for external FCA LO sources.

Click on a blue keyword to view the command details.

See Also

- SCPI SMC and VMC calibration examples.
- Learn about SMC and VMC calibrations
- Synchronizing the PNA and Controller
- SCPI Command Tree

SENSe<ch>:CORRection:COLLect:SESSion<n>:ACQuire <step>
(Write only) Acquire a calibration measurement.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Session number. Choose from 1 to 16.
- `<step>` Step number to acquire. Use `SENS:CORR:COLL:SESS:STEPS?` to find the number of steps required for the calibration.

**Examples**

```
SENSe2:CORR:COLL:SESS6:ACQ 5
```

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**SENSe<ch>:CORRection:COLLect:SESSion<n>:CKIT:PORT<p>:CATalog?**

*(Read only)* Returns a list of cal kits that are compatible with the connector on port `<p>`. The port connector type is set with `SENS:CORR:COLL:SESS:CONN:PORT`.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Session number. Choose from 1 to 16.
- `<p>` Port number of connector to query for compatible cal kits.
  1 - Input port of the mixer under test.
  2 - Output port of the mixer under test.
  3 - Output port of the calibration mixer.

*See the user interface (UI) equivalent of this command.*

**Examples**

```
```

**Return Type** Comma separated string values

**Default** Not Applicable

---

**SENSe<ch>:CORRection:COLLect:SESSion<n>:CKIT:PORT<p>[:SE lect] <calkit>**
(Read-Write) Set or return the Cal Kit for the specified port. Use `SENS::CORR::COLL::SESS::CKIT::PORT::CAT?` to list compatible Cal Kits.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16.
- `<p>` Port number for which to set cal kit.
  - 1 - Input port of the mixer under test.
  - 2 - Output port of the mixer under test.
  - Output port of MUT +1 - Output port of the calibration mixer. Generally this is 3.
  - See the user interface (UI) equivalent of this command.
- `<calkit>` Cal Kit Name

**Examples**

- `SENS::CORR::COLL::SESS::CKIT::PORT::SEL 85091A`
- `SENS2::CORR::COLL::SESS6::CKIT::PORT2::SEL?`

**Query Syntax**

`SENS<ch>::CORR::COLL::SESS<n>::CKIT::PORT<p>[:SEL]`?

**Return Type** String

**Default** Not Applicable

---

`SENS<ch>::CORR::COLL::SESSion<n>::CONN::PORT<p>[:SEL] <conn>`

(Read-Write) Set the connector type and sex for the specified port number. Catalog valid connector types using `SENS::CORR::COLL::GUID::CONN::CAT?`

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16.
- `<p>` Port number for which to specify a connector type.
- `<conn>` Name of the connector type

**Examples**

- `SENS2::CORR::COLL::SESS6::CONN::PORT1::SEL "N Type"

**Query Syntax**

`SENS<ch>::CORR::COLL::SESS<n>::CONN::PORT<p>[:SEL]?`

**Return Type** String
SENSe<ch>:CORRection:COLLect:SESSion<n>:DESC? <step>

(Read-only) Returns the connection prompt for the step. List the number of steps in the calibration using SENS:CORR:COLL:SESS:STEPS?.

**Parameters**
- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<n>`: Session number. Choose from 1 to 16.
- `<step>`: Step number

**Examples**
```
```

**Return Type** Numeric

**Default** Not Applicable

SENSe<ch>:CORRection:COLLect:SESSion<n>:DONE

(Write only) Ends the calibration sessions. Use **SAVE?** to calculate error terms and save the CalSet.

**Parameters**
- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<n>`: Session number. Choose from 1 to 16.

**Examples**
```
SENS1:CORR:COLL:SESS6:DONE
```

**Default** Not Applicable

SENSe<ch>:CORRection:COLLect:SESSion<n>:INI&Tiate <string>
(Write only) Initiates an SMC or VMC calibration session. Use the session number for subsequent SMC or VMC commands.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. If the session number already exists it will be terminated and a new session initiated.
- `<string>` Name of the calibration. Choose from:
  - "VMC" or "VectorMixerCal.VMCType"
  - "SMC" or "ScalarMixerCal.SMCType"

**Examples**

```
SENS1:CORR:COLL:SESS6:INITiate "VectorMixerCal.VMCType"
```

**Query Syntax** Not Applicable

**Default** Not Applicable

**SENSe<ch>:CORRection:COLLect:SESSion<n>:SAVE?**

(Read only) Finish the SMC or VMC calibration, compute error terms, populate and save the CalSet, and return the GUID of the Cal Set.

**Note:** The destination (Cal Register or User Cal Set) is determined by the setting of the `SENS:CORR:PREFER:CS:SAVE` command.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16.

**Examples**

```
SENS1:CORR:COLL:SESS6:SAVE?
```

**Return Type** String specifying the GUID of the CalSet produced by this session.

**Default** Not Applicable

**SENSe<ch>:CORRection:COLLect:SESSion<n>:STEPs?**
(Read-only) Returns the number of steps required by the Calibration.
To ensure this query always completes successfully, first send the write command:
SENS:CORR:COLL:SESS:STEP, then send the query.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number. Choose from 1 to 16.

Examples

SENS1:CORR:COLL:SESS6:STEPs?

Return Type Numeric

Default Not Applicable

Last Modified:

11-Dec-2007 Now can do external source LO cal
13-Aug-2007 New image and detail for <port> argument.
Performs scalar (SMC) calibration on a frequency converting device.

---

**SENSe:CORRection:COLLect:SESSion:SNC Commands**

Click on a blue keyword to view the command details.

**See Also**

- [Example Programs](#)
- [Learn about FCA Calibrations](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

**NOTE:** To configure a power meter and sensor see **SOURce:POWer** commands.

```plaintext
SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:ECAL:CHARacteriza <mod>,<char>
```
(Read-Write) Specifies the ECal module and characterization to be used for the SMC calibration.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<n>`  Session number. Choose from 1 to 16. [Learn about Cal sessions.](#)
- `<mod>`  1 - Electronic Calibration Module
- `<char>`  Specifies which characterization within the ECal module from which to read the confidence data.

  0  Factory characterization (data that was stored in the ECal module by Agilent). Default if not specified.

  1  User characterization #1

  2  User characterization #2

  3  User characterization #3

  4  User characterization #4

  5  User characterization #5

**Examples**

```
SENS:CORR:COLL:SESS:SMC:ECAL:CHAR 1, 2
```

**Query Syntax**


**Return Type**  Numeric

**Default**  1,0

---

```
SENSe<ch>:CORRection:COLLect:SESSion<n> ::SMC:FSIMulator:NETWork<x>:MODE <char>
```
(Read-Write) Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement. Learn more.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<x>` Apply network to input or output of mixer. Choose from:
  - **1** - Input of mixer
  - **2** - Output of mixer
- `<char>` Choose from:
  - **NONE** - Do nothing with effects of S2P file.
  - **EMBed** - Add effects of S2P file from the measurement results.
  - **DEEMbed** - Remove effects of S2P file from the measurement results.

**Examples**


**Query Syntax**

SENS<ch>:CORRection:COLLect:SESSion<n>:SMC:FSIMulator:NETWork<x>:MODE?

**Return Type** Character

**Default** NONE

**SENSe<ch>:CORRection:COLLect:SESSion<n> :SMC:FSIMulator:NETWork<x>:FLIEnam**

<string>
(Read-Write) Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement. Learn more.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number. Choose from 1 to 16. Learn about Cal sessions.

<x> Apply network to input or output of mixer. Choose from:

1 - Input of mixer

2 - Output of mixer

<string> Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

Examples

SENS:CORR:COLL:SESS:SMC:FSIM:NETW1:FIL "C:\Program Files\Agilent\Network Analyzer\Documents\WaveguideAdapt.S2P"

Query Syntax

SENS<ch>:CORRection:COLLect:SESSion<n>:SMC:FSIMulator:NETWork<x>:FILename?

Return Type String

Default Not Applicable

SENSe<ch>:CORR:COLL:SESSion<n>:SMC:PWRCal:SRCPort <string> Obsolete

(Read-Write) Specifies which port to calibrate.

Note: Beginning with Rev 6.0, this command is no longer necessary. Learn more. Because of improved calibration techniques, Both is always selected although a power meter measurement is performed only on port 1.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number. Choose from 1 to 16. Learn about Cal sessions.

<char> '1' Source port 1 (SMC forward direction)

'2' Source port 2 (SMC reverse direction)

'BOTH' (both forward and reverse directions)

Examples

SENS:CORR:COLL:SESS:SMC:PWRC:SRCP 'both'
SENSe2:CORR:COLL ect:SESSion6:SMC:PWRCal:SRCPort '2'

Query Syntax


(Read-Write) Sets ECAL Auto-Orientation ON or OFF. If setting auto-orientation OFF, you must manually specify the orientation of the ECAL module with *SENS:CORR:COLL:SESS:SMC:TWOP:ECAL:PORTmap*.

**Parameters**
- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<bool>`
  - 0 = Orientation OFF
  - 1 = Orientation ON

**Examples**

**Query Syntax**

**Return Type** Boolean
Default 1

---

**SENS<ch>:CORRection:COLLect:SESSion<n>:SMC:TWOPort:ECAL:PORTmap <mod>,<string>**

(Read-Write) Specifies the manual orientation (which ports of the module are connected to which ports of the PNA) when auto-orientation is OFF.

**Parameters**
- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<mod>` 1 - Electronic Calibration Module
- `<string>` Format in the following manner:
  
  Aw,Bx,Cy,Dz

  where

  - A, B, C, and D are literal ports on the ECAL module
- w, x, y, and z are substituted for PNA port numbers to which the ECAL module port is connected.

Ports of the module which are not used are omitted from the string.

**For example**, on a 4-port ECal module with

- port A connected to PNA port 2
- port B connected to PNA port 3
- port C not connected
- port D connected to PNA port 1

the string would be: A2, B3, D1

If either the receive port or source port (or load port for 2-port cal) of the CALC:PAR:SELected measurement is not in this string and orientation is OFF, an attempt to perform an ECal calibration will fail.

---

**Examples**

```
```

**Query Syntax**

```
```

**Return Type** String

**Default** "A1,B2"

---

**SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:TWOPort:METHod <string>**
(Read-Write) Specifies the guided ECal method for performing the thru portion of the calibration.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<n>`  Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<string>`  ECAL Method: Choose from:
  - 'DEFAULT' - Default
  - 'ADAP' - Adapter removal
  - 'FLUSH' - Flush Through
  - 'UNKN' - Unknown Thru

**Examples**

```
SENS:CORR:COLL:SESS:SMC:TWOPort:METH 'default'
```

**Query Syntax**


**Return Type**  String

**Default**  DEFAULT

**SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:TWOPort:OMITisolat <bool>**

(Read-Write) Select to omit or perform the isolation portion of the ECAL.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<n>`  Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<bool>`  1 - Omit isolation
  0 - Perform isolation

**Examples**

```
SENS:CORR:COLL:SESS:SMC:TWOPort:OMIT 1
```

**Query Syntax**

SENS:CORR:COLL:SMC:TWOPort:OMIT?

**Return Type**  Boolean

**Default**  1

**SENSe<ch>:CORRection:COLLect:SESSion<n >:SMC:TWOPort:OPTion <string>**
(Read-Write) Sets the SMC calibration to ECAL or MEChanical

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. [Learn about Cal sessions](#).
- `<char>` Choose from:
  - 'ECAL' Electronic Calibration Module
  - 'MECH' Mechanical Calibration Kit

**Examples**

```
SENS:CORR:COLL:SESS:SMC:TWOPort:OPTion 'ECAL'
```

**Query Syntax**

SENS:CORR:COLL:SESS:SMC:TWOPort:OPTion?

**Return Type** String

**Default** ECAL
Sense:Correction:Collect:Session:VMC Commands

Performs a vector (VMC) calibration on a frequency converting device.

SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:FSIMulator:NETWork<x>:MODE <char>

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about VMC Calibration
- Synchronizing the PNA and Controller
(Read-Write) Allows you to embed (add) or de-embed (remove) circuit network effects on the input and output of your mixer measurement. Learn more.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<x>` Apply network to input or output of mixer. Choose from:
  - 1 - Input of mixer
  - 2 - Output of mixer
- `<char>` Choose from:
  - **NONE** - Do nothing with effects of S2P file.
  - **EMBed** - Add effects of S2P file from the measurement results.
  - **DEEMbed** - Remove effects of S2P file from the measurement results.

**Examples**

```plaintext
```

**Query Syntax**

```plaintext
SENS<ch>:CORRection:COLLect:SESSion<n> :VMC:FSIMulator:NETWork<x>:MODE?
```

**Return Type** Character

Default NONE

---

**SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:FSIMulator:NETWork<x>:FILENAME <string>**

(Read-Write) Specifies the S2P filename to embed or de-embed on the input or output of your mixer measurement. Learn more.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<x>` Apply network to input or output of mixer. Choose from:
  - 1 - Input of mixer
  - 2 - Output of mixer
- `<string>` Filename of the S2P used for embedding or de-embedding. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

(Read-Write) Specifies the .S2P filename used for mixer characterization. Use the VMC:MIXer:CHARacterize:CAL:OPTion command to load the file for mixer characterization. Once loaded, use this command to query the current filename or set a new filename.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Session number. Choose from 1 to 16. Learn about Cal sessions.

<string> Filename of the S2P used for mixer characterization. Use the full path name, file name, and .S2P suffix, enclosed in quotes.

Examples


Query Syntax


Return Type String

Default C:/Program Files/Agilent/Network Analyzer/Documents/default.s2p

(Read-Write) Sets the mixer characterization method to ECal, Mechanical, or read from a file.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<char>`
  - **ECAL** - Electronic Calibration Module
  - **MECH** - Mechanical Calibration Kit
  - **FILE, <filename>** - Retrieve a mixer characterization file. Also specify the filename of the S2P used for mixer characterization. Use the full path name, file name, and .S2P suffix. Use the `VMC:CHARacterize:CAL:FILename` command to query the filename.

**Examples**

```plaintext
'or

file = "SENS:CORR:COLL:SESS6:VMC:MIXer:CHAR:CAL:FIL?" 'Read back the filename
```

**Query Syntax**


**Return Type** String

**Default** MECH


(Read-Write) Specifies the direction in which to characterize the calibration mixer. Learn more about the calibration mixer.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<bool>`
  - **False (0)** - Characterize the calibration mixer in the SAME direction as that specified in the mixer setup.
  - **True (1)** - Characterize the calibration mixer in the REVERSE direction as that specified in the mixer setup.

**Examples**

```plaintext
```

**Query Syntax**

SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:MIXer:ECAL:CHARacteriza <mod>,<char>

(Read-Write) Specifies the ECal module and characterization to be used for the mixer characterization portion of the calibration.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. [Learn about Cal sessions](#).
- `<mod>` 1 - Electronic Calibration Module
- `<char>` Characterization number in the specified ECAL module. Choose from:
  - 0  Factory characterization (data that was stored in the ECal module by Agilent). Default if not specified.
  - 1  User characterization #1
  - 2  User characterization #2
  - 3  User characterization #3
  - 4  User characterization #4
  - 5  User characterization #5

**Examples**  

```
SENSe:COR:COLL:SESS:VMC:MIX:ECAL:CHAR 1, 0
```

**Query Syntax**  


**Return Type** Numeric

- **Default** 1,0
**SENSe<ch>:CORRection:COLLect:SESSion<n>:VMC:OPERation <string>**

(Read-Write) Perform either full VMC calibration or mixer characterization only.

### Parameters

- **<ch>** Any existing channel number. If unspecified, value is set to 1
- **<n>** Session number. Choose from 1 to 16. Learn about Cal sessions.
- **<string>** 
  - `'CAL'` - full calibration and mixer characterization
  - `'CHAR'` - mixer characterization only (no reference mixer required) - Saves an .S2P file with the filename specified in `SENSe<ch>:CORRection:COLLect:SESSion<n>:VMC:CHARacterize:CAL:FILENAME <filename>`. If none is specified, a filename is automatically generated and can be queried using the filename command.

### Examples

```
SENSe:CORR:COLL:SESS:VMC:OPER 'CAL'
```

### Query Syntax

`SENSe:CORR:COLL:SESS:VMC:OPER?`

### Return Type

String

**Default** `CAL`
SENSe<ch>:CORRection:COLLect:SESSION<n>:VMC:TWOPort:ECAL:CHARacteriza <mod>,<char>

(Read-Write) Specifies the ECal module and characterization to be used for the VMC calibration.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. Learn about Cal sessions.
- `<mod>` 1 - Electronic Calibration Module
- `<char>` Characterization number in the specified ECAL module. Choose from:
  - 0 Factory characterization (data that was stored in the ECal module by Agilent). Default if not specified.
  - 1 User characterization #1
  - 2 User characterization #2
  - 3 User characterization #3
  - 4 User characterization #4
  - 5 User characterization #5

**Examples**

SENS:CORR:COLL:SESS:VMC:TWOP:ECAL:CHAR 1,1

**Query Syntax**


**Return Type** Integer

**Default** 1,0

---


---

2138
(Read-Write) Sets ECAL orientation for the VMC ECAL.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. [Learn about Cal sessions](#).
- `<bool>`  
  - `1` = ON  
  - `0` = OFF

**Examples**

```plaintext
```

**Query Syntax**

```plaintext
```

**Return Type** Integer

**Default** ON

---

**SENSe<ch>:CORRection:COLLect:SESSION<n> :VMC:TWOPort:ECAL:PORTmap <mod>, <string>**

(Read-Write) Specifies the manual orientation (which ports of the module are connected to which ports of the PNA) when orientation is turned off.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. [Learn about Cal sessions](#).
- `<mod>` 1 - Electronic Calibration Module
- `<string>` Port Map, formatted in the following manner:  
  ```plaintext
  Aw,Bx,Cy,Dz
  ```
  where:
  - A, B, C, and D are literal ports on the ECAL module.
  - w,x,y, z are substituted for PNA port numbers to which the ECAL module port is connected.
  - Ports of the module which are not used are omitted from the string.

For example, on a 4-port ECAL module with:
- port A connected to PNA port 2
- port B connected to PNA port 3
- port C not connected
- port D connected to PNA port 1

the string would be: A2,B3,D1

If either the receive port or source port (or load port for 2-port cal) of the measurement is not in this string and orientation is OFF, an attempt to perform an ECal will fail.

**Examples**
```
```

**Query Syntax**
```
```

**Return Type** string

**Default** "A1,B2"

---

**SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:TWOPort:METHod <string>**

*(Read-Write)* Specifies the guided ECal method for performing the thru portion of the calibration.

**Parameters**
- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Session number. Choose from 1 to 16. [Learn about Cal sessions](#).
- `<char>`
  - 'DEFAULT' - Default
  - 'ADAP' - Adapter removal
  - 'FLUSH' - Flush Through
  - 'UNKN' - Unknown Thru

**Examples**
```
SENS:CORR:COLL:SESS:VMC:TWOP:METH 'ADAP'
SENSe2:CORR:COLL:SESSion6:VMC:TWOPort:METHod 'FLUSH'
```

**Query Syntax**
```
SENS:CORR:COLL:SESS:VMC:TWOP:METH?
```

**Return Type** String

**Default** DEFAULT

---

**SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:TWOPort:OMItisolat <bool>**
(Read-Write) Select to omit or perform the isolation portion of the ECAL.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number. Choose from 1 to 16. Learn about Cal sessions.
<bool> 1 - omit isolation
          0 - perform isolation

Examples
SENS:CORR:COLL:SESS:VMC:TWOP:OMIT 1

Query Syntax
SENS:CORR:COLL:SESS:VMC:TWOP:OMIT?

Return Type Numeric

Default 1

SENSe<ch>:CORRection:COLLect:SESSion<n> :VMC:TWOPort:OPTion <string>

(Read-Write) Sets the 2-port calibration option to ECAL or MEChanical

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
<n> Session number. Choose from 1 to 16. Learn about Cal sessions.
<char> Choose from:
   'ECAL' Electronic Calibration Module
   'MECH' Mechanical Calibration Kit

Examples
SENS:CORR:COLL:SESS:VMC:TWOP:OPT 'MECH'
SENS:e2:CORR:COLL:SESSION6:VMC:TWOPort:OPTion 'ECAL'

Query Syntax
SENS:CORR:COLL:SESS:VMC:TWOP:OPT?

Return Type String

Default "MECH"

Last Modified:
Sense: Couple Commands

SENSe:COUPle

<table>
<thead>
<tr>
<th></th>
<th>PARameter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[STATe]</td>
</tr>
</tbody>
</table>

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Synchronizing the PNA and Controller
- SCPI Command Tree

SENSe<cnum>:COUPle <ALL | NONE>

(Read-Write) Sets the sweep mode as Chopped or Alternate.

Learn about Alternate Sweep

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <ALL | NONE> ALL - Sweep mode set to Chopped - reflection and transmission measured on the same sweep.

NONE - Sweep mode set to Alternate - reflection and transmission measured on separate sweeps. Improves Mixer bounce and Isolation measurements. Increases sweep time

Examples

SENS:COUP ALL
sense2:couple none

Query Syntax
SENS<cnum>:COUPle?

Return Type Character

Default ALL

SENSe<cnum>:COUPle:PARameter[:STATe] <bool>
(Read-Write) Turns ON and OFF Time Domain Trace Coupling. All of the measurements in the specified channel are coupled.

- To select Transform parameters to couple, use `CALC:TRAN:COUP:PAR`
- To select Gating parameters to couple, use `CALC:FILT:COUP:PAR`

Learn more about **Time Domain Trace Coupling**.

### Parameters

- `<cnum>`  
  Any existing channel number; if unspecified, value is set to 1.

- `<bool>`  
  **ON (or 1)** - Turns ON Time Domain Trace Coupling.
  
  **OFF (or 0)** - Turns OFF Time Domain Trace Coupling.

### Examples

```
SENS:COUP:PAR 0
sense2:couple:parameter:state on
```

### Query Syntax

```
SENSe<cnum>:COUPle:PARameter[:STATe]?
```

### Return Type

Boolean

### Default

ON (or 1)
Sense:FOM (Frequency Offset) Commands

Controls the frequency offset settings which cause stimulus and response frequencies to be different.

**Note**: These commands replace the previous FOM commands. Although the old commands will continue to work, they can NOT be mixed with these new commands.

![Command Diagram]

Click on a blue keyword to view the command details.

**See Also**

- FOM Example Program
- Learn about Frequency Offset
- Synchronizing the PNA and Controller
- SCPI Command Tree

**SENSe<cnum>:FOM[:STATe] <bool>**
(Read-Write) Turns Frequency Offset ON and OFF. Frequency offset settings are not enabled until this setting is ON.

Send this command (FOM ON) AFTER sending other FOM settings to avoid 'out-or-range' errors.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<bool>`: ON (or 1) - turns FOM ON. 
  OFF (or 0) - turns FOM OFF.

**Examples**

```plaintext
SENS:FOM 1
sense2:fom:state on
```

**Query Syntax**

SENSe<cnun>:FOM:STATe?

**Return Type**

Boolean

**Default**

OFF

---

**SENSe<cnun>:FOM:CATalog?**

(Read-only) Returns a comma-separated list of available range names in the PNA.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.

**Examples**

```plaintext
SENS:FOM:CAT?
returns
"Primary, Source, Receivers"
```

**Return Type**

String

**Default**

Not Applicable

---

**SENSe<cnun>:FOM:COUNt?**
(Read-only) Returns the number of valid ranges in the PNA.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.

Examples

```
SENS:FOM:COUN?
sense2:fom:count?
```

Query Syntax: `SENSe<cnum>:FOM:COUNt?`

Return Type: Numeric

**Default**: Not Applicable

---

`SENSe<cnum>:FOM:DISPlay:SELect <string>`

(Read-Write) Select the range to be displayed on the PNA x-axis. All traces in the channel have this same x-axis scaling.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<string>`: Range name. Case insensitive. Use `SENSe:FOM:CAT?` to see a list of available frequency range names.

Examples

```
SENS:FOM:DISPLAY:SELECT "source2"
sense2:fom:display:select "source"
```

Query Syntax: `SENSe<cnum>:FOM:DISPLAY:SELECT?`

Return Type: String

**Default**: Receivers

---

`SENSe<cnum>:FOM:RNUM? <string>`
(Read-only) Returns the number of a specified range name.
The FOM range items are typically numbered as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.
<string> Range name for which a number is being queried. Case insensitive. Use SENSE:FOM:CAT? to see a list of available range names.

Examples
SENS:FOM:RNUM? "receivers"
sense2:fom:rnum? "Source2"

Return Type Numeric

Default Not Applicable

SENSe<cnum>:FOM:RANGe<n>:COUPled <bool>

(Read-Write) Sets and returns the state of coupling (ON or OFF) of the specified range to the primary range.

Parameters

<cnum> Any existing channel number; if unspecified, value is set to 1.
<n> Range number to couple to primary range. An error is returned when attempting to couple to the Primary range (1).
<bool> ON (or 1) - Couple range to primary range.
OFF (or 0) - Do NOT couple to primary range.

Examples
SENS:FOM:RANG:COUP 1
sense2:fom:range2:coupled 0

Query Syntax SENSE<cnum>:FOM:RANGe<n>:COUPled?
Return Type Boolean

Default ON (or 1) Coupled
**SENSe<cnum>:FOM:RANGE<n>:FREQuency:CW <num>**

*(Read-Write)* Sets and returns the CW frequency.

This setting is valid for the primary range, or if the specified range is already uncoupled from the primary range and if the sweep type is CW.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<num>` CW frequency value in Hz. Choose any frequency within the range of the PNA.

**Examples**

```
SENS:FOM:RANGE:FREQ:CW 1e9
sense2:fom:range2:frequency:cw 10000000
```

**Query Syntax**

```
SENSe<cnum>:FOM:RANGE:<n>:FREQuency:CW?
```

**Return Type**

Numeric

**Default**

Center frequency of the PNA.

---

**SENSe<cnum>:FOM:RANGE<n>:FREQuency:DIVisor <num>**

*(Read-Write)* Sets and returns the divisor value.

This setting is valid only if the specified range is coupled to the primary range.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<num>` Divisor value (unitless).

**Examples**

```
SENS:FOM:RANGE:FREQ:DIV 3
sense2:fom:range2:frequency:divisor 0
```

**Query Syntax**

```
SENSe<cnum>:FOM:RANGE:<n>:FREQuency:DIVisor?
```

**Return Type**

Numeric

**Default**

1
SENSe<cnump>:FOM:RANGe<n>:FREQuency:MULTiplier <num>

(Read-Write) Sets and returns the multiplier value to be used when coupling this range to the primary range. This setting is valid only if the specified range is coupled to the primary range.

Parameters

<cnump> Any existing channel number; if unspecified, value is set to 1.
<n> Range number. If unspecified, value is set to 1.
<num> Multiplier value. (Unitless)

Examples

SENSe:FOM:RANG:FREQ:MULT 1
sense2:fom:range2:frequency:multiplier 2

Query Syntax

SENSe<cnump>:FOM:RANGe<n>:FREQuency:MULTiplier?

Return Type

Numeric

Default

1

SENSe<cnump>:FOM:RANGe<n>:FREQuency:OFFSet <num>

(Read-Write) Sets and returns the offset value to be used when coupling this range to the primary range. Learn more about offset value.

This setting is valid only if the specified range is coupled to the primary range.

Parameters

<cnump> Any existing channel number; if unspecified, value is set to 1.
<n> Range number. If unspecified, value is set to 1.
<num> Offset value. (Unitless)

Examples

SENSe:FOM:RANG:FREQ:OFFS 1E9
sense2:fom:range2:frequency:offset 10000000

Query Syntax

SENSe<cnump>:FOM:RANGe<n>:FREQuency:OFFSet?

Return Type

Numeric

Default

0

SENSe<cnump>:FOM:RANGe<n>:FREQuency:STARt <num>
(Read-Write) Sets and returns the Start value of frequency range. Also specify Stop frequency.

This setting is valid for the primary range, or if the specified range is already uncoupled from the primary range and if the sweep type is LOG or LINear.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<num>` Start value in Hz. Choose any frequency within the range of the PNA.

**Examples**

```
SENS:FOM:RANG:FREQ:STAR 1GHz
sense2:fom:range2:frequency:start 100000000
```

**Query Syntax**

```
SENSe<cnum>:FOM:RANGe<n>:FREQuency:STARt?
```

**Return Type** Numeric

**Default** Minimum frequency of the PNA.

---

**SENSe<cnum>:FOM:RANGe<n>:FREQuency:STOP <num>**

(Read-Write) Sets and returns the Stop value of frequency range. Also specify Start frequency.

This setting is valid for the primary range, or if the specified range is already uncoupled from the primary range and if the sweep type is LOG or LINear.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<num>` Stop value in Hz. Choose any frequency within the range of the PNA.

**Examples**

```
SENS:FOM:RANG:FREQ:STOP 1e12
sense2:fom:range2:frequency:stop 10000000000
```

**Query Syntax**

```
SENSe<cnum>:FOM:RANGe<n>:FREQuency:STOP?
```

**Return Type** Numeric

**Default** Maximum frequency of the PNA.

---

**SENSe<cnum>:FOM:RANGe<n>:NAME?**
(Read-only) Returns the name of range<n>.
The FOM range items are typically named as follows:

1. Primary
2. Source
3. Receivers
4. Source2 (if present)

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.

**Examples**

```
SENS:FOM:RANG:NAME?
sense2:fom:range2:name?
```

**Return Type** String

**Default** Not Applicable

---

**SENSe<cnum>:FOM:RANGE<n>:SWEep:TYPE <char>**

(Read-Write) Sets and returns the sweep type to be used with the specified range.

This setting is valid only if the specified range is already uncoupled from the primary range.

Learn aboutUnsupported Sweep Type combinations.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<char>` Sweep type. Choose from:
  - **CW** - Also specify CW frequency.
  - **LINear** - Also specify frequency Start/Stop or Center/Span
  - **LOG** - Also specify frequency Start/Stop or Center/Span
  - **POWer** - Also specify power Start/Stop or Center/Span
  - **SEGment** - Also specify segment sweep settings.
Examples
SENS:FOM:RANG:SWE:TYPE LOG
sense2:fom:range2:sweep:type linear:

Query Syntax
SENSe<cnum>:FOM:RANGe<n>:SWEep:TYPE?

Return Type
Character

Default
Linear

Last Modified:
5-Jul-2007 Fixed state link
8-Mar-2007 MQX New topic
**SENSe:FOM:RGn:SEG:ADD**

Constructs a segment table for a specified UNCOUPLED FOM range.

**Note:** Do NOT use **SENSe:SEG:ADD** commands for FOM segment sweep.

### See Also
- Other **SENSe:FOM** Commands
- Example Programs
- Synchronizing the PNA and Controller
- **SCPI Command Tree**
(Write-only) Adds a segment.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.
- `<s>`: Segment number to add. If unspecified, value is set to 1. Segment numbers must be sequential. If a new number is added where one currently exists, the existing segment and those following are incremented by one.

**Examples**

Two segments exist (1 and 2). The following command will add a new segment (1). The existing (1 and 2) will become (2 and 3) respectively.

```
sense2:fom:range2:segment:add
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SENSe<cnum>:FOM:RANGE<n>:SEGMENT<s>:BWIDth[:RESolution] <num>**

*(Read-Write)* Sets the IF Bandwidth for the specified segment. First set `SENSe:FOM:RANGE:SEGMENT:BWIDth:CONTrol ON`. All subsequent segments that are added assume the new IF Bandwidth value.

Valid either for Receiver range or for Primary range when coupled to Receiver.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.
- `<s>`: Segment number for which to set independent IF Bandwidth.
- `<num>`: IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. [See the list of valid IFBW values](#). If an invalid number is specified, the analyzer will round up to the closest valid number.

**Examples**

```
SENSe:FOm:RANGe:SEGMeNT:BWIDth 100
sense2:fom:range2:segment4:bwidth:resolution 1e3
```

**Query Syntax**

`SENSe<cnum>:FOM:RANGE<n>:SEGMENT<s>:BWIDth[:RESolution]?`

**Return Type**

Numeric

**Default**

See [Preset IFBW](#) for your PNA model.
SENSe<cnum>:FOM:RANGe<n>:SEGMent:BWIDth[:RESolution]:CONTrol <bool>

(Read-Write) Specifies whether the IF Bandwidth resolution can be set independently for each segment. When set, each segment added after this will be set to ON automatically.

Valid either for Receiver range or for Primary range. Primary range value is ignored unless Receiver is coupled to Primary.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.
- <bool> **ON** (or 1) - turns Bandwidth control ON. Bandwidth can be set for each segment
  **OFF** (or 0) - turns Bandwidth control OFF. Use the channel IF bandwidth setting instead.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:FOM:RANG:SEGM:BWIDth:CONT 0</td>
<td>Turns Bandwidth control OFF. Use channel IF bandwidth setting instead.</td>
</tr>
<tr>
<td>sense2:fom:range2:segment:bwidth:resolution:control 1</td>
<td>Turns Bandwidth control ON. Bandwidth can be set for each segment</td>
</tr>
</tbody>
</table>

Query Syntax

SENSe<cnum>:FOM:RANGe<n>:SEGMent:BWIDth[:RESolution]:CONTrol?

Return Type Boolean

Default OFF

SENSe<cnum>:FOM:RANGe<n>:SEGMent:COUNt?

(Read-only) Returns the number of segments that exist for the specified range.

Parameters

- <cnum> Any existing channel number; if unspecified, value is set to 1.
- <n> Range number. If unspecified, value is set to 1.

Examples

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:FOM:RANG:SEGM:COUN?</td>
<td>Returns number of segments for specified range</td>
</tr>
<tr>
<td>sense2:fom:range2:segment:count?</td>
<td>Returns number of segments for specified range</td>
</tr>
</tbody>
</table>

Return Type Numeric

Default Not Applicable

SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:DELe
(Write-only) Deletes the specified sweep segment.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.
- `<s>` Number of the segment to delete. If unspecified, value is set to 1.

Examples

```
SENS: FOM: RANG: SEGM3: DEL
sense2: fom: range2: segment4: delete
```

Query Syntax

Not Applicable

Default

Not Applicable

**SENSe<cnum>**:FOM:RANGe<n>:SEGMent:DELete:ALL

(Write-only) Deletes all sweep segments in the specified range.

Parameters

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Range number. If unspecified, value is set to 1.

Examples

```
SENS: FOM: RANG: SEGM: DEL: ALL
sense2: fom: range2: segment: delete: all
```

Query Syntax

Not Applicable

Default

Not Applicable

**SENSe<cnum>**:FOM:RANGe<n>:SEGMent<s>:FREQuency:CENTer <num>
(Read-Write) Sets and returns the center frequency for the specified sweep segment. Also specify segment frequency span.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.
- `<s>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Center Frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

**Examples**

```plaintext
SENS: FOM: RANG: SEGM: FREQ: CENT 1GHz
sense2: fom: range2: segment4: frequency: center 1e9
```

**Query Syntax**

`SENSe<cnum>: FOM: RANGe<n>: SEGMent<s>: FREQuency: CENTer?`

**Return Type**

Numeric

**Default**

Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

---

**SENSe<cnum>: FOM: RANGe<n>: SEGMent<s>: FREQuency: SPAN <num>**

(Read-Write) Sets and returns the span frequency for the specified sweep segment. Also specify segment center frequency.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.
- `<s>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Frequency span in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

**Examples**

```plaintext
SENS: FOM: RANG: SEGM: FREQ: SPAN 1GHz
sense2: fom: range2: segment4: frequency: span 1e9
```

**Query Syntax**

`SENSe<cnum>: FOM: RANGe<n>: SEGMent<s>: FREQuency: SPAN?`

**Return Type**

Numeric
Default  If first segment, frequency span of the analyzer. Otherwise 0.

**SENSe\(<cnum>\):FOM:RANGe\(<n>\):SEGMent\(<s>\):FREQuency:STARt \(<num>\)**

(Read-Write) Sets and returns the start frequency for the specified sweep segment. Also specify segment stop frequency.

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

**Parameters**

- \(<cnum>\) Any existing channel number; if unspecified, value is set to 1.
- \(<n>\) Range number. If unspecified, value is set to 1.
- \(<s>\) Segment number to modify. Choose any existing segment number.
- \(<num>\) Start frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

**Examples**

```
SENs:FOM:RANg:SEGm:FREQ:STARt 1GHz
sense2:fom:range2:segment4:frequency:start 1e9
```

**Query Syntax**  SENSE\(<cnum>\):FOM:RANGe\(<n>\):SEGMent\(<s>\):FREQuency:STARt?

**Return Type**  Numeric

**Default**  Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

**SENSe\(<cnum>\):FOM:RANGe\(<n>\):SEGMent\(<s>\):FREQuency:STOP \(<num>\)**
(Read-Write) Sets and returns the stop frequency for the specified sweep segment. Also specify segment start frequency.

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.
- `<s>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Stop frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

Examples

```
SENS:FOM:RANG:SEGM:FREQ:STOP 1GHz
sense2:fom:range2:segment4:frequency:stop 1e9
```

Query Syntax: `SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:FREQuency:STOP?`

Return Type: Numeric

Default: Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

---

`SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:POWer<p>[:LEVel] <num>`

(Read-Write) Sets the Port Power level for the specified sweep segment. First set `SENS:FOM:RANG:SEGM:POW:CONTrol ON`. When `port power is Coupled`, setting port power for one port will apply port power for all source ports. All subsequent segments that are added assume the new Power Level value.

Valid either for Source ranges or for Primary range when `coupled` to the source.

Parameters

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.
- `<s>`: Segment number to modify. Choose any existing segment number.
- `<p>`: Port number of the source. If unspecified, value is set to 1.
- `<num>`: Power level in dBm.
**Note:** The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, send SOUR:POW? MAX and SOUR:POW? MIN. *(SOUR:POW:ATT:AUTO must be set to ON).*

Actual achievable leveled power depends on frequency.

**Examples**

```
sense2: fom: range2: segment4: power2: level 5
```

**Query Syntax**

```
SENS<channel>: FOM: RANG<n>: SEGM<segment>: POW[p]: LEVEL?
```

**Return Type**

Numeric

**Default**

0

**SENS<channel>: FOM: RANG<n>: SEGM<segment>: POW<level>: CONTR <bool>**

*(Read-Write)* Specifies whether Power Level is to be set independently for each segment.

Valid either for Source ranges or for Primary range. Primary range value is ignored unless Source is **coupled** to Primary.

**Parameters**

- `<channel>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.
- `<bool>`: ON (or 1) - Power level will be set for each segment. OFF (or 0) - Use the channel power level setting.

**Examples**

```
SENS: FOM: RANG: SEGM: POW: CONTR 0
sense2: fom: range2: segment: power: control: on
```

**Query Syntax**

```
SENS<channel>: FOM: RANG<n>: SEGM<segment>: POW[p]: CONTR?
```

**Return Type**

Boolean

**Default**

OFF (or 0)

**SENS<channel>: FOM: RANG<n>: SEGM<s>: STATe <bool>**
(Read-Write) Turns the specified sweep segment ON or OFF.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.
- `<s>`: Segment number to be turned ON or OFF. Choose any existing segment number.
- `<bool>`: ON (or 1) - turns segment ON.
- OFF (or 0) - turns segment OFF.

**Examples**

```
SENS:FOM:RANG:SEGM 0
sense2:fom:range2:segment4:state on
```

**Query Syntax**

SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:SWEep:POINts?

**Return Type**

Boolean

**Default**

OFF (or 0)

---

SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:SWEep:POINts <num>

(Read-Write) Sets the number of data points for the specified sweep segment.

**Parameters**

- `<cnum>`: Any existing channel number; if unspecified, value is set to 1.
- `<n>`: Range number. If unspecified, value is set to 1.
- `<s>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Number of points in the segment. The total number of points in all segments cannot exceed 20001. A segment can have as few as 1 point.

**Examples**

```
SENS:FOM:RANG:SEGM:SWE:POIN 101
sense2:fom:range2:segment4:sweep:points 201
```

**Query Syntax**

SENSe<cnum>:FOM:RANGe<n>:SEGMent<s>:SWEep:POINts?

**Return Type**

Numeric

**Default**

21
SENSe\(<cnum>:FOM:RANGe<n>:SEGMen{s}:SWEep:TIME <num>\)

*(Read-Write)* Sets the time the PNA takes to sweep the specified segment.
Valid ONLY for receiver ranges.

**Parameters**

- \(<cnum>\) Any existing channel number; if unspecified, value is set to 1.
- \(<n>\) Range number. If unspecified, value is set to 1.
- \(<s>\) Segment number for which to set sweep time.
- \(<num>\) Sweep time in seconds. Choose a number between 0 and 100

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:FOM:RANG:SEG:SWE:TIME 1</td>
<td></td>
</tr>
<tr>
<td>sense2:fom:range2:segment3:sweep:time .1</td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe\(<cnum>:FOM:RANGe<n>:SEGMen{s}:SWEep:TIME?\)

**Return Type** Numeric

**Default** Not Applicable

SENSe\(<cnum>:FOM:RANGe<n>:SEGMen{s}:SWEep:TIME:CONTrol <bool>\)

*(Read-Write)* Specifies whether Sweep Time can be set independently for each sweep segment.
Valid either for Receiver ranges or for Primary range. Primary range value is ignored unless Receiver is coupled to Primary.

**Parameters**

- \(<cnum>\) Any existing channel number; if unspecified, value is set to 1.
- \(<n>\) Range number. If unspecified, value is set to 1.
- \(<bool>\) ON (or 1) - Sweep time will be set for each segment.
  OFF (or 0) - Use the channel sweep time setting.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:FOM:RANG:SEG:SWE:TIME:CONT 1</td>
<td></td>
</tr>
<tr>
<td>sense2:fom:range2:segment3:sweep:time:control off</td>
<td></td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe\(<cnum>:FOM:RANGe<n>:SEGMen{s}:SWEep:TIME:CONTrol?\)

**Return Type** Boolean
Default OFF

Last Modified:

21-Jun-2007 Increased max number of points
Sense:Frequency Commands

Sets the frequency sweep functions of the analyzer.

SENSe: FReQuency

CENTER [CW | FIXed] SPAN START STOP

Click on a blue keyword to view the command details.

See Also

- Example using some of these commands.
- Learn about Frequency Sweep
- Synchronizing the PNA and Controller
- SCPI Command Tree

SENSe<cnum>:FREQuency:CENTer <num>

(Read-Write) Sets the center frequency of the analyzer.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Center frequency. Choose any number between the minimum and maximum frequency limits of the analyzer. Units are Hz.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```plaintext
SENSe:FREQ:CENT 1000000
sense2:frequency:center 1mhz
sense2:frequency:center 1e6
```

**Query Syntax**

SENSe<cnum>:FREQuency:CENTer?

**Return Type** Numeric

**Default** Center of the analyzer's frequency span

SENSe<cnum>:FREQuency[:CW | :FIXed] <num>
(Read-Write) Sets the Continuous Wave (or Fixed) frequency. Must also send `SENS:SWEEP:TYPE CW` to put the analyzer into CW sweep mode.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` CW frequency. Choose any number between the `minimum` and `maximum` frequency limits of the analyzer. Units are Hz.

This command will accept `MIN` or `MAX` instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```
SENS:FREQ 1000000
SENS:FREQ:CW MIN
sense2:frequency:fixed 1mhz
```

**Query Syntax**

`SENSe<cnum>:FREQuency[:CW | :FIXed]?

**Return Type**

Numeric

**Default** 1 GHz

---

**SENSe<cnum>:FREQuency:SPAN <num>**

(Read-Write) Sets the frequency span of the analyzer.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Frequency span in Hz. Choose any number between 0 (minimum) and the `maximum` frequency span of the analyzer.

This command will accept `MIN` or `MAX` instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```
SENS:FREQ:SPAN 1000000
sense2:frequency:span max
```

**Query Syntax**

`SENSe<cnum>:FREQuency:SPAN?

**Return Type**

Numeric

**Default** Maximum frequency span of the analyzer
(Read-Write) Sets the start frequency of the analyzer.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Start frequency. Choose any number between the MIN and MAX frequency limits of the analyzer. Units are Hz

If FREQ:START is set greater than FREQ:STOP, then STOP is set equal to START.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```
SENS:FREQ:STAR 1000000
sense2:frequency:start MIN
```

**Query Syntax**

`SENSe<cnum>:FREQuency:STARt?`

If Sweep type is segment, this query returns the start frequency of the first segment.

**Return Type** Numeric

**Default** Minimum frequency of the analyzer

---

**SENSe<cnum>:FREQuency:STOP <num>**

(Read-Write) Sets the stop frequency of the analyzer.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Stop frequency. Choose any number between: the minimum and maximum frequency limits of the analyzer. Units are Hz.

If FREQ:STOP is set less than FREQ:START, then START will be set equal to STOP.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```
SENS:FREQ:STOP 1000000
sense2:frequency:stop max
```

**Query Syntax**

`SENSe<cnum>:FREQuency:STOP?`

If Sweep type is segment, this query returns the stop frequency of the last segment.

**Return Type** Numeric

**Default** Maximum frequency of the analyzer
SENSe:GCSetup Commands

Controls the Gain Compression configuration.

SENSe:GCSetup:

AMODe <char>

COMPression:

| ALGorithm <char>
| BACKoff:LEVel <num>
| DELTa:X <num>
| DELTa:Y <num>
| INTerpolation <bool>
| LEVel <num>

EOSoperation <string>

PMAP

| INPut?
| OUTPut?

POWer:

| LINear:INPut:LEVel <num>
| REVerse:LEVel <num>
| STARt:LEVel <num>
| STOP:LEVel <num>

SAFE:

| CPADjustment <num>
| ENABLe <bool>
| FPADjustment <num>
| FTHReshold <num>

SFA?

SMARt:
| MITerations <num> |
| SITerations <bool> |
| STIME <num> |
| TOLERance <num> |

SWEep:
| FREQuency:POINts <num> |
| POWer:POINts <num> |

Click on a blue keyword to view the command details.

**Other Gain Compression commands**

The calibration commands listed in this topic are supplemental to the Guided Cal commands.

- **CALC:CUSTom:DEFine** - creates a gain compression measurement.
- **GCA Calibration** uses the [Guided Calibration commands](#), except for the following:
  - **Sens:Corr:GCS:Power** - sets power level for Source Power Cal

See Also

- **Example Program**: Create and Cal a Gain Compression Measurement
- **Learn about Gain Compression Application**
- **Synchronizing the PNA and Controller**
- **SCPI Command Tree**

---

**SENSe<ch>:GCSetup:AMODE <char>**
(Read-Write) Set and read the method by which gain compression data is acquired.

Parameters

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<char>` Choose from:
  - **PFREQuency** - 2D Power Per Frequency
  - **FPOWer** - 2D Frequency Per Power
  - **SMARtsweep** - Smart Sweep

Examples

```
SENS:GCS:AMOD SMAR
sense:gcsetup:amode pfrequency
```

Query Syntax

```
SENSe<ch>:GCSetup:AMODe ?
```

Return Type

Character

Default

SMARtsweep

---

**SENSe<ch>:GCSetup:COMPression:ALGorithm <char>**

(Read-Write) Set and read the algorithm method used to compute gain compression.

Parameters

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<char>` Algorithm method. Choose from:
  - **CFLG** - compression from linear gain
  - **CFMG** - compression from maximum gain
  - **BACKoff** - compression from BackOff
  - **XYCOM** - X/Y Compression

Examples

```
SENS:GCS:COMP:ALG BACK
sense:gcsetup:compression:algorithm XYcom
```

Query Syntax

```
SENSe<ch>:GCSetup:COMPression:ALGorithm?
```

Return Type

Character

Default

CFLG
SENSe<ch>:GCSetup:COMPression:BACKoff:LEVel <num>

(Read-Write) Set and read value for the BackOff compression algorithm.

Parameters

- <ch>  Any existing GCA channel. If unspecified, value is set to 1.
- <num>  Backoff value in dB. Choose from 30 to (-30)

Examples

- SENS:GCS:COMP:BACK:LEV 10
- sense:gcsetup:compression:backoff:level 5

Query Syntax  SENS<ch>:GCSetup:COMPression:BACKoff:LEVEL?

Return Type  Numeric

Default  10

SENSe<ch>:GCSetup:COMPression:DELTA:X <num>

(Read-Write) Set and read the 'X” value in the delta X/Y compression algorithm.

Parameters

- <ch>  Any existing GCA channel. If unspecified, value is set to 1.
- <num>  X value in dB. Choose from 30 to (-30)

Examples

- SENS:GCS:COMP:DEL:TX 9
- sense:gcsetup:compression:delta:X 8

Query Syntax  SENS<ch>:GCSetup:COMPression:DELTA:X?

Return Type  Numeric

Default  10

SENSe<ch>:GCSetup:COMPression:DELTA:Y <num>
(Read-Write) Set and read the 'Y' value in the delta X/Y compression algorithm.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.
<num> Y value in dB. Choose from 30 to (-30)

Examples

SENS:GCS:COMP:DELT:Y 9
sense:gcsetup:compression:delta:Y 8

Query Syntax
SENSe<ch>:GCSetup:COMPression:DELTa:Y?

Return Type Numeric

Default 9

SENSe<ch>:GCSetup:COMPression:INTerpolation <bool>

(Read-Write) Sets whether or not interpolation should be performed on 2D measured compression data. Applies ONLY to 2D acquisition modes.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.
<bool> Choose from:

ON or (1) Interpolate the results
OFF or (0) Do NOT interpolate the results but return the value closest to compression.

Examples
SENS:GCS:COMP:INT 1
sense:gcsetup:compression:interpolation off

Query Syntax
SENSe<ch>:GCSetup:COMPression:INTerpolation?

Return Type Boolean

Default OFF

SENSe<ch>:GCSetup:COMPression:LEVEL <num>
**SENS:GCS:COMP:ALG CFLG**

(Read-Write) Set and read the desired gain reduction (from reference gain).

This value is used for Compression from Linear Gain and Compression from Maximum Gain.

Use **SENS:GCS:COMP:ALG CFLG** to set this compression algorithm.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Compression level in dB. Choose a value greater than 0.1 dB.

**Examples**

```
SENS:GCS:COMP:LEV 1
sense:gcsetup:compression:level 3
```

**Query Syntax**

SENSe<ch>:GCSetup:COMPression:LEVel?

**Return Type**

Numeric

**Default**

1

---

**SENSe<ch>:GCSetup:EOSoperation <char>**

(Read-Write) Set and read the This setting is used to protect a sensitive device from too much power during the sweep retrace. Other instrument settings or channels may over-ride this setting. Learn more.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<char>` End Of Sweep operation. Choose from:
  - STANdard Use the default PNA method. Learn more.
  - POFF Always turn power OFF while waiting.
  - PSTArt Sweep Start power
  - PSTOp Sweep Stop power.

**Examples**

```
SENS:GCS:EOS PSTA
sense:gcsetup:eosoperation standard
```

**Query Syntax**

SENSe<ch>:GCSetup:EOSoperation?

**Return Type**

Character

**Default**

STANdard
SENSe<ch>:GCSetup:PMAP <in>,<out>
(Write-only) Set and read the DUT-to-PNA port mapping for the Gain Compression measurement.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

<in> PNA port which is connected to the DUT input.

<out> PNA port which is connected to the DUT output.

Examples

SENSe:GCS:PMAP 1,2

SENSe:GCS:PMAP 2,1

Query Syntax Not Applicable

Default 1,2

SENSe<ch>:GCSetup:PMAP:INPut?
(Read-only) Read the PNA port number to be connected to the DUT Input.

Use SENSe:GCS:PORTMap to set the port mapping.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

Examples

SENSe:GCS:PMAP:INPut?

SENSe:GCS:PMAP:INPut?

Query Syntax Not Applicable

Return Type Numeric

Default 1

SENSe<ch>:GCSetup:PMAP:OUTPut?
(Read-only) Read the PNA port number to be connected to the DUT Output.

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.

Examples

SENSe:GCS:PMAP:OUTPut?

SENSe:GCS:PMAP:OUTPut?

Query Syntax Not Applicable

Return Type Numeric

Default 2
SENSe<ch>:GCSetup:POWer:LINear:INPut:LEVel <num>

(Read-Write) Set and read the input power at which Linear Gain and all S-parameters are measured.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Input power level in dBm. Choose a value from +30 to (-30).

**Examples**

```
SENS:GCS:POW:LIN:INP:LEV 0
sense:gcsetup:power:linear:input:level -10
```

**Query Syntax**

SENSe<ch>:GCSetup:POWer:LINear:INPut:LEVel?

**Return Type**

Numeric

**Default**

-25 dBm

SENSe<ch>:GCSetup:POWer:REVerse:LEVel <num>

(Read-Write) Set and read the reverse power level to the DUT. This is applied to the DUT output port when making reverse measurements like S22.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Reverse power level in dBm. Choose a value from +30 to (-30).

**Examples**

```
SENS:GCS:POW:REV:LEV 0
sense:gcsetup:power:reverse:level -5
```

**Query Syntax**

SENSe<ch>:GCSetup:POWer:REVerse:LEVel?

**Return Type**

Numeric

**Default**

-5

SENSe<ch>:GCSetup:POWer:STARt:LEVel <num>
(Read-Write) Set and read the start power level.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Start power level in dBm. Choose a value from +30 to (-30).

**Examples**

```
SENS:GCS:POW:STAR:LEV 0
sense:gcsetup:power:start:level -5
```

**Query Syntax**  
`SENSe<ch>:GCSetup:POWer:STARt:LEVel?`

**Return Type**  
Numeric

**Default**  
-30

---

`SENSe<ch>:GCSetup:POWer:STOP:LEVel <num>`

(Read-Write) Set and read the stop power level.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Stop power level in dBm. Choose a value from +30 to (-30).

**Examples**

```
SENS:GCS:POW:STOP:LEV 0
sense:gcsetup:power:stop:level -5
```

**Query Syntax**  
`SENSe<ch>:GCSetup:POWer:STOP:LEVel?`

**Return Type**  
Numeric

**Default**  
-5

---

`SENSe<ch>:GCSetup:SAFE:CPADjustment <num>`
(Read-Write) Set and read the Safe Sweep COARSE power adjustment. Learn more.

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> Coarse power adjustment setting in dBm. Choose a value from +30 to (-30).

Examples

```
SENS:GCS:SAFE:CPAD 2
sense:gcsetup:safe:cpadjustment 3.5
```

Query Syntax

SENSe<ch>:GCSetup:SAFE:CPADjustment?

Return Type

Numeric

Default

3.0

---

SENSe<ch>:GCSetup:SAFE:ENABle <bool>

(Read-Write) Set and read the (ON | OFF) state of Safe Sweep mode. Learn more

Parameters

- <ch> Any existing GCA channel. If unspecified, value is set to 1.
- <num> (Boolean) - Safe Sweep state. Choose from:
  - OFF (or 0) - Disable Safe Sweep
  - ON (or 1) - Enable Safe Sweep

Examples

```
SENS:GCS:SAFE:ENAB 0
sense:gcsetup:safe:enable 1
```

Query Syntax

SENSe<ch>:GCSetup:SAFE:ENABle?

Return Type

Boolean

Default

0

---

SENSe<ch>:GCSetup:SAFE:FPADjustment <num>
(Read-Write) Set and read the Safe Sweep FINE power adjustment. Learn more

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.
<num> Fine power adjustment setting in dBm. Choose a value from +30 to (-30).

Examples

- SENS:GCS:SAFE:FPAD 2
- sense:gcsetup:safe:fpadjustment .5

Query Syntax SENSE<ch>:GCSetup:SAFE:FPADjustment?

Return Type Numeric

Default 1.0 dBm

SENSe<ch>:GCSetup:SAFE:FTHReshold <num>

(Read-Write) Set and read the compression level in which Safe Sweep changes from the COARSE power adjustment to the FINE power adjustment. Learn more

Parameters

<ch> Any existing GCA channel. If unspecified, value is set to 1.
<num> Threshold setting in dB. Choose a value from +30 to (-30).

Examples

- SENS:GCS:SAFE:FTHR .1
- sense:gcsetup:safe:fthreshold .5

Query Syntax SENSE<ch>:GCSetup:SAFE:FTHReshold?

Return Type Numeric

Default 0.75 dB

SENSe<ch>:GCSetup:SFA?
(Read-only) Returns a comma-separated list of the frequency indexes that were out of tolerance for SMART Sweep mode, or at the power limit for 2D acquisition modes. Zero (0) is the first frequency data point.

Must be Single triggered. Invalid results occur if the GCA channel is continuously sweeping.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.

**Examples**

- `SENS:GCS:SFA?`
- `sense:gcsetup:sfa?`

**Return Type** Comma-separated list of frequency indexes.

**Default** Not Applicable

---

**SENS<ch>:=GCSetup:SMART:MITerations <num>**

(Read-Write) Set and read the maximum permitted number of iterations which SMART Sweep may utilize to find the desired compression level, to within the specified tolerance.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<num>` Maximum number of iterations. Choose a value between 1 and 10e8

**Examples**

- `SENS:GCS:SMAR:MIT 5`
- `sense:gcsetup:smart:iterations 3`

**Query Syntax** `SENS<ch>:=GCSetup:SMART:MITerations?`

**Return Type** Numeric

**Default** 20

---

**SENS<ch>:=GCSetup:SMART:SITerations <bool>**
(Read-Write) Set and read enable for showing intermediate results for each iteration of SMART Sweep

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<bool>` Choose from:
  - **ON** or **1** Compression traces are updated after each iteration.
  - **OFF** or **0** Compression traces are updated after ALL iterations are complete.

**Examples**

```
SENS:GCS:SMAR:SIT 1
sense:gcsetup:smart:siterations off
```

**Query Syntax**

SENSe<ch>:GCSetup:SMART:SIterations?

**Return Type**

Boolean

**Default**

OFF

---

**SENSe<ch>:GCSetup:SMARt:STIMe <num>**

(Read-Write) Set and read the amount of time SMART Sweep will dwell at the first point where the input power changes by the Backoff or X level. Applies only to SMART Sweep when Backoff or XY compression methods are selected. Learn more.

**Parameters**

- `<ch>` Any existing GCA channel. If unspecified, value is set to 1.
- `<bool>` Settling time in seconds. Choose any positive value.

**Examples**

```
SENS:GCS:SMAR:STIM 1
sense:gcsetup:smart:stime .1
```

**Query Syntax**

SENSe<ch>:GCSetup:SMART:STIMe?

**Return Type**

Numeric

**Default**

0

---

**SENSe<ch>:GCSetup:SMARt:TOLerance <num>**
(Read-Write) Set and read the acceptable range SMART Sweep will allow for the measured compression level.

**Parameters**

- `<ch>`  Any existing GCA channel. If unspecified, value is set to 1.
- `<num>`  Tolerance level in dBm. Choose a value between .01 and 10

**Examples**

```plaintext
SENS:GCS:SMAR:TOL .1
sense:gcsetup:smart:tolerance .05
```

**Query Syntax**  `SENSe<ch>:GCSetup:SMART:TOLERance?`

**Return Type**  Numeric

**Default**  .05

---

**SENSe<ch>:GCSetup:SWEep:FREQuency:POINts <num>**

(Read-Write) Set and read the number of data points in each frequency sweep.  Learn more

**Parameters**

- `<ch>`  Any existing GCA channel. If unspecified, value is set to 1.
- `<num>`  Frequency points. Do not exceed the max number of data points.

**See Data Points Limit**

**Examples**

```plaintext
SENS:GCS:SWE:FREQ:POIN 201
sense:gcsetup:sweep:frequency:points 101
```

**Query Syntax**  `SENSe<ch>:GCSetup:SWEep:FREQuency:POINts?`

**Return Type**  Numeric

**Default**  201

---

**SENSe<ch>:GCSetup:SWEep:POWer:POINts <num>**

---

2181
(Read-Write) Set and read the number of data points in each power sweep. Applies ONLY to 2D acquisition modes.

### Parameters

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ch&gt;</td>
<td>Any existing GCA channel. If unspecified, value is set to 1.</td>
</tr>
<tr>
<td>&lt;num&gt;</td>
<td>Power points. Do not exceed the max number of data points.</td>
</tr>
</tbody>
</table>

See Data Points Limit

### Examples

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:GCS:SWE:POW:POIN 50</td>
</tr>
<tr>
<td>sense:gcsetup:sweep:power:points 21</td>
</tr>
</tbody>
</table>

### Query Syntax

SENSe<ch>:GCSetup:SWEep:POWer:POINts?

### Return Type

Numeric

### Default

26

Last Modified:

9-Nov-2007    MX New topic
Sense:IF Commands

Controls the IF source and gain settings for use with the E836x H11 Option.

See IF/DSP and Pulse commands for the PNA-X.

Click on a blue keyword to view the command details.

- See IF Access User Interface Settings
- Synchronizing the PNA and Controller
- SCPI Command Tree

**SENSe<cnun>:IF:FILTer:SAMPle:COUNt <num>**

(Read-Write) Sets and returns the number of taps in the IF filter. The IF filter sample count setting is only used when **SENSe:IF:FILTer:SAMPle:COUNt:MODE** is set to MANUAL. **Critical Note:**

**Parameters**

- `<cnun>`: Existing channel number to manipulate. If unspecified, `<cnun>` is set to 1.
- `<num>`: An integer value. (MIN and MAX return the minimum and maximum allowed values, respectively.)

**Examples**

SENSe:IF:FILT:SAMP:COUN 40
SENSe:IF:FILT:SAMP:COUN:MAX

**Query Syntax**

SENSe:IF:FILTer:SAMPle:COUNt?

**Return Type**

Numeric

**Default**

Instrument dependent.

**SENSe<cnun>:IF:FILTer:SAMPle:COUNt:MODE <char>**

2183
(Read-Write) Sets and returns the IF filter sample count mode to the specified value. When in MANUAL mode, the value specified for the **IF Filter sample count** is used as the number of taps in the IF filter. **Critical Note:**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Existing channel number to manipulate. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
<tr>
<td>&lt;char&gt;</td>
<td>Choose either AUTO or MANUAL.</td>
</tr>
</tbody>
</table>

**Examples**

```
sense2:if:filter:sample:count:mode AUTO
```

**Query Syntax**

```
SENS:IF:FILTer:SAMPle:COUNt:MODE?
```

**Return Type**

Character

**Default**

AUTO

---

**SENSe<cnum>:IF:FILTer:SAMPle:PERiod <num>**

(Read-Write) Sets and returns the IF filter sample period. The IF filter sample period setting is only used by the instrument when the **SENS:IF:FILT:SAMP:PER:MODE** is set to MANUAL. **Critical Note:**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Existing channel number to manipulate. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:IF:FILT:SAMP:PER 6 us
sense2:if:filter:sample:period maximum
```

**Query Syntax**

```
SENS:IF:FILTer:SAMPle:PERiod?
```

**Return Type**

Numeric

**Default**

Instrument dependent.

---

**SENSe<cnum>:IF:FILTer:SAMPle:PERiod:CATalog?**

(Read-Only) Returns the list of allowed IF filter sample periods for this instrument. **Critical Note:**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;cnum&gt;</td>
<td>Existing channel number to manipulate. If unspecified, &lt;cnum&gt; is set to 1.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:IF:FILT:SAMP:PER:CAT?
sense2:if:filter:sample:period:catalog?
```

**Query Syntax**

```
SENS:IF:FILTer:SAMPle:PERiod:CATalog?
```

**Return Type**

String

**Default**

AUTO
### SENSe<cnum>:IF:FILTER:SAMPLE:PERiod:MODE <char>

*(Read-Write)* Sets and returns the IF filter sample period mode to the specified value. **Critical Note:**

**Parameters**
- `<cnum>`: Existing channel number to manipulate. If unspecified, `<cnum>` is set to 1.
- `<char>`: Sample period mode. Choose from **AUTO** or **MANUAL**.

**Examples**
- `sense2:if:filter:sample:period:mode AUTO`

**Query Syntax**
- `SENS:IF:FILTER:SAMPLE:PERIOD:MODE?`

**Return Type**
- Character

**Default**
- AUTO

### SENSe<cnum>:IF:GAIN:ALL[:STATe] <char>

*(Write only)* Sets the gain state for ALL receivers to Auto or Manual.

**Parameters**
- `<cnum>`: Existing channel number to manipulate. If unspecified, `<cnum>` is set to 1.
- `<char>`: Choose from **AUTO** or **MANUAL**.

**Examples**
- `SENS:IF:GAIN:ALL AUTO`
- `sense:if:gain:all:state manual`

**Query Syntax**
- Not Applicable

**Return Type**
- Character

**Default**
- AUTO

### SENSe<cnum>:IF:GAIN:LEVel <id>, <level>


Manually sets the gain level for the specified receiver.

**Parameters**

- `<cnum>`: Existing channel number to manipulate. If unspecified, `<cnum>` is set to 1.
- `<id>`: Choose from: 'A', 'B', 'R1', 'R2'.

**Note:** The A and R1 receivers are always switched together. B and R2 are also always switched together. For example, if you specify 'A', R1 will also be switched.

- `<level>`: Gain level. Choose from:
  - 0 - Lowest gain setting
  - 1
  - 2 - Highest gain setting

**Examples**

- `SENS:IF:GAIN:LEVel 'A', 1`

**Query Syntax**

- `SENS<cnum>:IF:GAIN:LEVel? <id>`

**Return Type**

- Numeric

**Default**

- 0

---

**SENS<cnum>:IF:GAIN[:STATe]?, <id>**

*(Read only)* Returns the gain state for the specified receiver. Use `SENS:IF:GAIN:ALL` to set the gain state for all channels.

**Parameters**

- `<cnum>`: Existing channel number to manipulate. If unspecified, `<cnum>` is set to 1.
- `<id>`: Choose from: 'A', 'B', 'R1', 'R2'.

**Note:** The A and R1 receivers are always switched together. B and R2 are also always switched together. For example, if you specify 'A', R1 will also be switched.

**Examples**

- `SENS:IF:GAIN? 'A'`

**Return Type**

- Boolean

**Default**

- Not Applicable
SENSe<cnum>:IF:GATE:STATe <boolean>
(Read-Write) Sets or returns the IF filter gate state.

**Parameters**
- **<cnum>** Existing channel number to manipulate. If unspecified, <cnum> is set to 1.
- **<boolean>** Choose from ON or OFF

**Examples**
- `SENSe:IF:FILT:GATE:STAT ON`
- `sense2:if:filter:gate:state 0`

**Query Syntax**
SENSeIF:FILTer:SAMPle:COUNt:MODE?

**Return Type**
Boolean

**Default**
AUTO

---

SENSe<cnum>:IF:SOURce:PATH <id>, <char>
(Read Write) Sets the source path for the specified receiver. An error is returned if <id> is not valid, or if <char> is not valid for the specified <id>.

**Parameters**
- **<cnum>** Existing channel number to manipulate. If unspecified, <cnum> is set to 1.
- **<id>** Choose from: 'A', 'B', 'R1', 'R2'
- **<char>** Choose from:
  - **NORMAL** - the PNA decides the appropriate IF input paths.
  - **EXTERNAL** - always use the rear panel IF inputs.

**Note**: The A and R1 receivers are always switched together. B and R2 are also always switched together. For example, if you specify "A", R1 will also be switched.

**Examples**
- `SENSe:IF:SOUR:PATH 'A', Ext`

**Query Syntax**
SENSe<cnum>:IF:SOURce:PATH? <id>

**Return Type**
Character

**Default**
Normal
**Sense:IF Commands**

Controls the IF filter for use with the PNA X.

![IF Path Configuration](image)

- Click on a blue keyword to view the command details.
- Synchronizing the PNA and Controller
- SCPI Command Tree

**PNA-X DSP Block diagram**

All of the Sense:IF commands, except the frequency commands, make settings related to the DSP section of the IF Path.

See the entire IF Path Configuration.

**Note:** For any of the Filter “Stage” parameters to take effect, `SENS:IF:FILT:AUTO` must be set to OFF (MANUAL) and mode, and `SENS:IF:FILT:CMOD` must be set to OFF.
Critical Note: These commands act on the selected measurement. You can select one measurement for each channel using `Calc:Par:Select`.

**SENSe<cnum>:IF:FIlt:AUTO <bool>**

(Read-Write) Sets and returns whether the PNA configures the 3-stage digital filter settings or they will be configured manually. When making manual settings, also send `SENS:IF:FIlt:CMOD OFF` which routes the IF through the 3-stage filter.

Critical Note

**Parameters**
- **<cnum>** (Existing channel number. If unspecified, <cnum> is set to 1.)*
- **<bool>** (Boolean)

**Examples**
- `SENS:IF:FIlt:AUTO 1`
- `sense2:if:filter:auto 0`

**Query Syntax**
- `SENSe<cnum>:IF:FILT:AUTO?`

**Return Type**
- **Boolean**

**Default**
- **ON**

**SENSe:IF:FIlt:CMODe <bool>**

(Read-Write) Sets and returns the ADC capture mode modeled as a 2-pole switch in the above diagram. The switch either bypasses or routes the IF through the 3-stage digital filter.

Critical Note

**Parameters**
- **<bool>** (Boolean)

**Examples**
- `SENS:IF:FILT:CMOD 1`
- `sense2:if:filter:cmode 0`

**Query Syntax**
- `SENSe<cnum>:IF:FILT:CMODe?`
SENSe<cnum>:IF:FILTER:ERRors?
(Read-only) Returns the error string associated with the digital filters. The return string has three fields separated by commas: "stage1 status, stage2 status, stage3 status"

Each of these fields can contain one or more of the following error codes:

- **NO ERROR**
- **NUMBER-OF-COEFFICIENTS** - the number of coefficients is excessive for that filter-stage
- **COEFFICIENT VALUE** - one or more coefficients are out of range for that filter-stage
- **SUM-OF-COEFFICIENTS** - the sum of all coefficients is excessive for that filter-stage,
- **FREQUENCY** - the frequency for Stage 1 is out of range (only applies stage1 field),
- **PARAMETER** - one or more parameters are out of range (only applies to stage 3 field)

**Critical Note**

**Parameters**

- `<cnum>` Existing channel number. If unspecified, `<cnum>` is set to 1.

**Examples**

- 'example return strings''
  - NO ERROR, NO ERROR, NO ERROR
  - indicates no errors,
  - *SUM-OF-COEFFICIENTS, NO ERROR, NO ERROR
  - indicates that the sum of all filter coefficients exceed the maximum value for the Stage-1 filter,
  - *COEFFICIENT *SUM-OF-COEFFICIENTS, NO ERROR, *PARAMETER
  - indicates a problems with Stage 1 coefficients and a problem with one or more of the parameters associated with the Stage 3 filter.

**Return Type** String

**Default** Not Applicable

SENSe<cnum>:IF:FILTER:STAGE<n>:COEFFicients <coef>
(Read-Write) Sets and returns the digital filter coefficients of the specified stage.

**Critical Note**

**Parameters**
- `<cnum>`: Existing channel number. If unspecified, `<cnum>` is set to 1.
- `<n>`: Stage number. Choose 1 or 2
- `<coef>`: An array of real numbers. Filter coefficients

**Examples**
- `SENS:IF:FILT:STAG2:COEF 0,0.1,0.7,0.7,0.1`
- `sense2:if:filter:stage1:coefficients`

**Query Syntax**
- `SENSe<cnum>:IF:FILTer:STAGE<n>:COEFFicients?`

**Example**
- `SENS:IF:FILT:STAG2:COEF?`
- `sense2:if:filter:stage1:coefficients?`

**Return Type**
- Numeric
- **Default** Stage dependent

---

**SENSe<cnum>:IF:FILTer:STAGE<n>:COUNt? [char]**

(Read-only) Returns the number of taps in the digital filter of the specified stage. The filter sample count setting is only used when `SENSe:IF:FILTer:AUTO` is set to False (MANUAL).

**Critical Note**

**Parameters**
- `<cnum>`: Existing channel number. If unspecified, `<cnum>` is set to 1.
- `<n>`: Stage number. Choose 1 or 2
- `[char]`: Optional parameter. Choose from:
  - not specified - returns the current number of coefficients for the specified stage.
  - **MIN** - returns the minimum number of coefficients for the specified stage. Stage1: 10, Stage2: 1
  - **MAX** - returns the maximum number of coefficients for the specified stage. Stage1 & 2: 1024

**Examples**
- `SENS:IF:FILT:STAG2:COUN?`
- `sense2:if:filter:stage1:count? max`

**Return Type**
- Numeric
- **Default** Stage dependent
SENSe<cnump>:IF:FILTer:STAGE1:FREQuency <value>

(Read-Write) Sets and returns the Numerically Controlled Oscillator (NCO) frequency. This command is only used when SENSe:IF:FILTer:AUTO is set to False (Manual).

Critical Note

Parameters
  <cnump> Existing channel number. If unspecified, <cnump> is set to 1.
  <value> Stage 1 Frequency. Min value= 0 Hz, Max value= 15 MHz. Or programmatically use the Max and Min queries to determine the range of settable values.

Examples
SENSe<cnump>:IF:FILTer:STAGE1:FREQuency 9e6
SENSe:IF:FILTer:STAGE1:FREQuency 9.2e6

Query Syntax
SENSe<cnump>:IF:FILTer:STAGE1:FREQuency?
  'returns the current parameter value
SENSe<cnump>:IF:FILTer:STAGE1:FREQuency? Min
  'returns the minimum frequency value.
SENSe<cnump>:IF:FILTer:STAGE1:FREQuency? Max
  'returns the maximum frequency value.

Return Type
  Default 9 MHz

SENSe<cnump>:IF:FILTer:STAGE3:CATalog?

(Read-only) Returns a list of strings for the currently supported filter types that can be used for the stage 3 filter. This command is only used when SENSe:IF:FILTer:AUTO is set to False (Manual). See SENS:IF:FILT:STAGE3:TYPE for a list of currently supported filter types.

Critical Note

Parameters
  <cnump> Existing channel number. If unspecified, <cnump> is set to 1.

Examples
SENSe<cnump>:IF:FILTer:STAGE3:CATalog?
SENSe:IF:FILTer:STAGE3:CATalog?

Return Type
  Default Not Applicable
SENSe<cnum>:IF:FILTer:STAGe3:TYPE <value>
(Read-Write) Sets and returns the Stage 3 filter type. This command is only used when SENSe:IF:FILTer:AUTO is set to False (Manual).

Critical Note

Parameters

- <cnum> Existing channel number. If unspecified, <cnum> is set to 1.
- <value> (String) Filter type. Choose from:
  - "RECT" Rectangular Window Filter
  - "TUKEY" Tukey Filter
  - "PWIN" Pulse Window Filter

Examples
SENSe<cnm>:IF:FILT:STAGE3:TYPE RECT
SENSe<cnm>:IF:FILT:STAGE3:TYPE PWIN
sense2:if:filter:stage3:typ pwin

Query Syntax
SENSe<cnum>:IF:FILTer:STAGE3:TYPE?

Return Type
String

Default
"TUKEY"

SENSe<cnum>:IF:FILTer:STAGe3:PARameter <p>, <value>
(Read-Write) Sets and returns the Stage 3 filter parameters.

Must first select the filter type (SENSe:IF:FILTer:STAGe3:TYPE) before setting these parameters.

Use SENSe:IF:FILTer:STAGE3:PCAT? to return a list of the available parameters for the currently selected filter type.

Critical Note

Parameters

- <cnum> Existing channel number. If unspecified, <cnum> is set to 1.
- <p> (String) Filter parameter. Choose from:
  - "C" - Tap count (Tukey, RECT, PWIN)
  - "P" - Period (PWIN ONLY)
  - "D" - Delay (PWIN ONLY)
  - "W" - Width (PWIN ONLY)
  - "R" - Ramp Count (PWIN ONLY)
(String) Parameter Value for the specified stage 3 parameter. Use the query form to return the minimum and maximum values for the specified parameter.

**Examples**

SENS:IF:FILT:STAGE3:PAR "C", 64
sense2:if:filter:stage3:parameter "d", 0.5E-6

**Query Syntax**

returns the current parameter value

SENSS<cnm>:IF:FILTer:STAGE3:PARameter? <p>, Min
returns the minimum parameter value.

SENSS<cnm>:IF:FILTer:STAGE3:PARameter? <p>, Max
returns the maximum parameter value.

**Examples**

SENS:IF:FILT:STAGE3:PAR? "C"
sense2:if:filter:stage3:parameter? "d", min

**Return Type**

**Default**

RECT: C = 1

PWIN: C=1E6, P=10ms, D=50us, W=50us, R=7

TUKEY: C=1

---

**SENSS<cnm>:IF:FILTer:STAGE3:PCATalog?**

(Read-only) Returns a list of the available parameters for the currently selected filter type.

**Critical Note**

**Parameters**

<cnm> Existing channel number. If unspecified, <cnm> is set to 1.

**Examples**

SENS:IF:FILT:STAGE3:PCAT?
sense2:if:filter:stage3:pcatalog?

**Return Type**

**Default**

String

Not Applicable

---

**SENSS<cnm>:IF:FREQuency:AUTO <bool>**
(Read-Write) Sets and returns the method for specifying the way the IF Frequency is determined.

**Critical Note**

**Parameters**
- `<cnum>` Existing channel number. If unspecified, `<cnum>` is set to 1.
- `<bool>` (Boolean)
  - **ON** (or 1) - Automatic. PNA determines the setting for the IF frequency.
  - **OFF** (or 0) - Manual. Use `SENS:IF:FREQ` to set the frequency.

**Examples**
- `SENS:IF:FREQ:AUTO 1`
- `sense2:if:frequency:auto 0`

**Query Syntax**
- `SENS<cnum>:IF:FREQuency:AUTO?`

**Return Type**
- Boolean

**Default**
- ON

---

`SENS<cnum>:IF:FREQuency[:VALue] <value>`

(Read-Write) Sets and returns the IF frequency for ALL receiver paths being used for the specified channel. To set this frequency, `SENS:IF:FREQ:AUTO` must be set to OFF (Manual).

Also returns the maximum and minimum allowable frequency settings.

**Critical Note**

**Parameters**
- `<cnum>` Existing channel number. If unspecified, `<cnum>` is set to 1.
- `<value>` (Numeric) Frequency value.

**Examples**
- `SENS:IF:FREQ`
- `sense2:if:frequency`

**Query Syntax**
- `SENS<cnum>:IF:FREQuency?`
  - 'returns the current frequency setting
- `SENS<cnum>:IF:FREQuency? Max`
  - 'returns the maximum allowable frequency setting
- `SENS<cnum>:IF:FREQuency? Min`
  - 'returns the minimum allowable frequency setting

**Return Type**
- Numeric

**Default**
- 9 MHz
Last Modified:

18-Jan-2007    MX New topic
Sense:Mixer Commands

Performs Mixer setup and configuration.

SENSe:MIXer:

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about the Frequency Converter Application
- Synchronizing the PNA and Controller
- SCPI Command Tree

Note: If you are changing several mixer configuration settings, you can make all the changes first and then issue the Calculate and Apply commands as you would do from the user interface.

SENSe<ch>:MIXer:APPLy
Applies the mixer setup settings and turns the channel ON. (Performs the same function as the Apply button on the mixer setup dialog box).

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1

**Examples**

`SENS:MIX:APPL`

**Query Syntax** Not Applicable

**Default** Not Applicable

**SENSe<ch>:MIXer:AVOidspurs <bool>**

Sets and returns the state of the avoid spurs feature. Learn more about avoid spurs.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` Avoid spurs state. Choose from:
  - 0 - Avoid spurs OFF
  - 1 - Avoid spurs ON

**Examples**

`SENS:MIX:AVO`

`sense2:mixer:avoidspurs 1`

**Query Syntax** `SENSe<ch>:MIXer:AVOidspurs?`

**Return Type** Boolean

**Default** 0 (OFF)

**SENSe<ch>:MIXer:CALCulate <char>**

Calculates the Input, IF, or Output frequencies of the mixer setup and updates the channel settings.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Mixer port to be calculated. Choose from:
<table>
<thead>
<tr>
<th>&lt;char&gt;</th>
<th>1st or only stage requires:</th>
<th>In addition, 2nd stage requires:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>INPUT</strong></td>
<td>• Output Start and Stop frequencies</td>
<td>• IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>• LO frequency</td>
<td>• 2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>• Output sideband (High or Low)</td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td><strong>Both</strong></td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td><strong>OUTPUT</strong></td>
<td>• Input Start and Stop frequencies</td>
<td>• IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>• LO frequency</td>
<td>• 2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>• Output sideband (High or Low)</td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td><strong>LO_1</strong></td>
<td>• Input Start and Stop frequencies</td>
<td>• IF Start and Stop frequencies</td>
</tr>
<tr>
<td></td>
<td>• Output frequency</td>
<td>• 2nd LO frequency</td>
</tr>
<tr>
<td></td>
<td>• Output sideband (High or Low)</td>
<td>• IF sideband (High or Low)</td>
</tr>
<tr>
<td><strong>LO_2</strong></td>
<td>NA</td>
<td>• Input Start and stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1st LO start and stop frequencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Output frequency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IF sideband(High or Low)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Output sideband(High or Low)</td>
</tr>
</tbody>
</table>

**Examples**

SENS:MIX:CALC Output

**Query Syntax**

Not Applicable

**Default**

Not Applicable

SENSe<ch>:MIXer:LOAD <name>
(Write-only) Loads a previously-configured mixer attributes file (.mxr)

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<name>` Path and file name (including .mxr extension) to load.

**Examples**

```
SENSe:MIXer:LOAD "C:\Program Files\Agilent\Network Analyzer\Documents\Mixer\MyMixer.mxr"
```

**Default** Not Applicable

---

**SENSe<ch>:MIXer:SAVE <name>**

(Write-only) Saves the settings for the mixer/converter test setup to a mixer attributes file.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<name>` Path and file name (including .mxr extension) to save.

**Examples**

```
SENSe:MIXer:SAVE "C:\Program Files\Agilent\Network Analyzer\Documents\Mixer\MyMixer.mxr"
```

**Default** Not Applicable

---

**SENSe<ch>:MIXer:STAGe <n>**

(Read-Write) Number of IF stages of the mixer.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Number of stages. Choose either 1 or 2

**Examples**

```
SENSe:MIX:STAG 2
SENSe:MIXer:STAGE 1
```

**Query Syntax**

`SENSe<ch>:MIXer:STAGe?`

**Return Type** Numeric

**Default** 1
SENSe<ch>:MIXer:IF:FREQuency:DENominator <num>

(Read-Write) Sets or returns the denominator value of the IF Fractional Multiplier. See Note

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.
<num> Denominator value.

Examples

SENSe:MIX:IF:FREQ:DEN 5
SENSe2:MIXer:IF:FREQ:DENominator 4

Query Syntax

SENSe<ch>:MIXer:IF:FREQuency:DENominator?

Return Type

Numeric

Default

Not Applicable

SENSe<ch>:MIXer:IF:FREQuency:FIXed <num>

(Read-Write) Sets or returns the fixed frequency of the IF. See Note

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.
<num> Frequency.

Examples

SENSe:MIXer:IF:FREQ:FIXed 1e9
SENSe2:MIXer:IF:FREQ:FIXed 1000000000

Query Syntax

SENSe<ch>:MIXer:IF:FREQuency:FIXed?

Return Type

Numeric

Default

Not Applicable

SENSe<ch>:MIXer:IF:FREQuency:NUMerator <num>
**SENSe<ch>:MIXer:IF:FREQuency:NUMerator <num>**

(Read-Write) Sets or returns the numerator value of the IF Fractional Multiplier. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<num>` Numerator value of the IF fractional multiplier

**Examples**

```
SENSe:MIX:IF:FREQ:NUM 4
SENSe2:MIXer:IF2:FREQ:NUMerator 3
```

**Query Syntax**

`SENSe<ch>:MIXer:IF:FREQuency:NUMerator?`

**Return Type**

Numeric

**Default**

Not Applicable

**SENSe<ch>:MIXer:IF:FREQuency:SIDeband <char>**

(Read-Write) Sets or returns the value of the IF sideband. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Sideband value. Choose from
  - **LOW** - Low or Difference (-)
  - **HIGH** - High or Sum (+)

**Examples**

```
SENSe:MIX:IF:FREQ:SIDE LOW
SENSe2:MIXer:IF:FREQ:SIDEband HIGH
```

**Query Syntax**

`SENSe<ch>:MIXer:IF:FREQuency:SIDeband?`

**Return Type**

Character

**Default**

LOW

**SENSe<ch>:MIXer:IF:FREQuency:STARt <num>**

2202
**SENSe<ch>:MIXer:IF:FREQuency:STARt <num>**

(Read-Write) Sets or returns the IF start frequency value of the mixer. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<num>` IF Start Frequency value

**Examples**

```
SENS:MIX:IF:FREQ:STAR 1e9
SESe2:MIXer:IF:FREQ:STARt 1000000000
```

**Query Syntax**

`SENSe<ch>:MIXer:IF:FREQuency:STARt?`

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe<ch>:MIXer:IF:FREQuency:STOP <num>**

(Read-Write) Sets or returns the stop frequency value of the mixer IF frequency. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<num>` IF Stop Frequency value

**Examples**

```
SENS:MIX:IF:FREQ:STOP 2e9
SESe2:MIXer:IF:FREQ:STOP 2000000000
```

**Query Syntax**

`SENSe<ch>:MIXer:IF:FREQuency:STOP?`

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe<ch>:MIXer:INPut:FREQuency:DENominator <value>**
(Read-Write) Sets or returns the denominator value of the Input Fractional Multiplier. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Input denominator value.

**Examples**

- `SENS:MIX:INP:FREQ:DEN 5`
- `SENS2:MIXer:INPut:FREQ:DENominator 4`

**Query Syntax**

`SENSe<ch>:MIXer:INPut:FREQuency:DENominator?`

**Return Type**

Numeric

**Default** Not Applicable

---

`SENSe<ch>:MIXer:INPut:FREQuency:FIXed<value>`

(Read-Write) Sets or returns the fixed frequency of the input. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Input frequency.

**Examples**

- `SENSe:MIXer:INPut:FREQ:FIXed 1e9`
- `SENSe2:MIXer:INPut:FREQ:FIXed 1000000000`

**Query Syntax**

`SENSe<ch>:MIXer:INPut:FREQuency:FIXed?`

**Return Type**

Numeric

**Default** Not Applicable

---

`SENSe<ch>:MIXer:INPut:FREQuency:MODE <char>`
(Read-Write) Sets or returns the Input sweep mode.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Input sweep mode. Choose either **FIXED** or **SWEPT**

**Examples**

```
SENS:MIX:INP:FREQ:MODE FIXED
SENS:2:MIXer:INP:FREQ:MODE swept
```

**Query Syntax**

`SENSe<ch>:MIXer:INPut:FREQuency:MODE?`

**Return Type** Character

**Default** Fixed

---

`SENSe<ch>:MIXer:INPut:FREQuency:NUMerator <value>`

(Read-Write) Sets or returns the numerator value of the Input Fractional Multiplier. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Input numerator value.

**Examples**

```
SENS:MIX:INP:FREQ:NUM 3
SENSe:2:MIXer:INPut:FREQ:NUMerator 1
```

**Query Syntax**

`SENSe<ch>:MIXer:INPut:FREQuency:NUMerator?`

**Return Type** Numeric

**Default** Not Applicable

---

`SENSe<ch>:MIXer:INPut:FREQuency:STARt <value>`
(Read-Write) Sets or returns the Input start frequency value of the mixer.  

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Input Start frequency

**Examples**

```
SENS:MIX:INP:FREQ:STAR 1e9
SENS2:MIX:INP:FREQ:START  1000000000
```

**Query Syntax**

```
SENS<ch>:MIX:INP:FWQ:START?
```

**Return Type**

Numeric

**Default** Not Applicable

---

**SENS<ch>:MIX:INP:FWQ:STOP <value>**

(Read-Write) Sets or returns the Input stop frequency value of the mixer.  

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Input stop frequency

**Examples**

```
SENS:MIX:INP:FREQ:STOP 2e9
SENS2:MIX:INP:FREQ:STOP  2000000000
```

**Query Syntax**

```
SENS<ch>:MIX:INP:FWQ:STOP?
```

**Return Type**

Numeric

**Default** Not Applicable

---

**SENS<ch>:MIX:INP:POWer <value>**
(Read-Write) Sets or returns the value of the Input Power.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Input power in dBm.

**Examples**

```
SENS:MIX:INP:POW 9
SENSe2:MIXer:INPut:POWer 5
```

**Query Syntax**

`SENSe<ch>:MIXer:INPut:POWer?`

**Return Type**

Numeric

**Default** Not Applicable

---

**SENSe<ch>:MIXer:INPut:POWer:USENominal <bool>**

(Read-Write) Toggles the Nominal Incident Power setting ON and OFF. This setting is ONLY to be used with SMC measurements, not VMC. Learn more about Nominal Incident Power.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` (boolean) - Nominal Incident Power State. Choose from:
  - True (1) - Turn nominal incident power ON
  - False (0) - Turn nominal incident power OFF

**Examples**

```
SENS:MIX:INP:POW:USEN 1
SENSe2:MIXer:INPut:POWer:USENominal false
```

**Query Syntax**

`SENSe<ch>:MIXer:INPut:POWer:USENominal?`

**Return Type**

Boolean

**Default** False

---

**SENSe<ch>:MIXer:LO<n>:FREQuency:DENominator <value>**
(Read-Write) Sets or returns the denominator value of the LO Fractional Multiplier. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` LO stage number. Choose 1 or 2.
- `<value>` LO denominator.

**Examples**

```
SENS:MIX:LO:FREQ:DEN 5
SENSe2:MIXer:LO2:FREQ:DENominator 4
```

**Query Syntax**

SENS<ch>:MIXer:LO<n>:FREQuency:DENominator?

**Return Type** Numeric

**Default** 1

---

**SENS<ch>:MIXer:LO<n>:FREQuency:FIXed <value>**

(Read-Write) Sets or returns the fixed frequency of the specified mixer LO. See Note

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` LO stage number. Choose 1 or 2
- `<value>` LO frequency.

**Examples**

```
SENS:MIX:LO:FREQ:FIX 1e9
SENSe2:MIXer:LO2:FREQ:FIXed 1000000000
```

**Query Syntax**

SENS<ch>:MIXer:LO<n>:FREQuency:FIXed?

**Return Type** Numeric

**Default** Not Applicable

---

**SENS<ch>:MIXer:LO<n>:FREQuency:ILTI <bool>**
(Read-Write) Specifies whether to use the Input frequency that is greater than the LO or less than the LO. To learn more, see the mixer setup dialog box help.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` LO stage number. Choose 1 or 2
- `<bool>` True (1) - Use the Input that is Greater than the specified LO.
  False (0) - Use the Input that is Less than the specified LO.

**Examples**

```
SENS:MIX:LO1:FREQ:ILTI 1
sense2:mixer:lo2:frequency:ilti true
```

**Query Syntax**

```
SENSe<ch>:MIXer:LO<n>:FREQuency:ILTI?
```

**Return Type** Boolean

**Default** False

---

**SENSe<ch>:MIXer:LO<n>:FREQuency:MODE <char>**

(Read-Write) Sets or returns the LO sweep mode.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` LO stage number. Choose 1 or 2
- `<char>` LO sweep mode. Choose either FIXED or SWEPT

**Examples**

```
SENS:MIX:LO:FREQ:MODE FIXED
SENSe2:MIXer:LO2:FREQ:MODE swept
```

**Query Syntax**

```
SENSe<ch>:MIXer:LO<n>:FREQuency:MODE?
```

**Return Type** Character

**Default** Fixed

---

**SENSe<ch>:MIXer:LO<n>:FREQuency:NUMerator <value>**
**SENSe<ch>:MIXer:LO<n>:FREQuency:NUMerator <value>**

*(Read-Write)* Sets or returns the numerator value of the LO Fractional Multiplier. See Note

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> LO stage number. Choose 1 or 2

<value> LO Numerator.

**Examples**

SENS:MIX:LO:FREQ:NUM 5
SENSe2:MIXer:LO2:FREQ:NUMerator 4

**Query Syntax**

SENSe<ch>:MIXer:LO<n>:FREQuency:NUMerator?

**Return Type** Numeric

**Default** Not Applicable

**SENSe<ch>:MIXer:LO<n>:FREQuency:STARt <value>**

*(Read-Write)* Sets or returns the LO start frequency value. See Note

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> LO stage number. Choose 1 or 2

<value> LO Start Frequency in Hertz.

**Examples**

SENS:MIX:LO:FREQ:STAR 5E9

**Query Syntax**

SENSe<ch>:MIXer:LO<n>:FREQuency:START?

**Return Type** Numeric

**Default** Not Applicable

**SENSe<ch>:MIXer:LO<n>:FREQuency:STOP <value>**
(Read-Write) Sets or returns the LO stop frequency value. See Note

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> LO stage number. Choose 1 or 2

<value> LO Stop Frequency in Hertz.

Examples SENS:MIX:LO:FREQ:STOP 5E9

Query Syntax SENSE<ch>:MIXer:LO<n>:FREQuency:STOP?

Return Type Numeric

Default Not Applicable

SENSe<ch>:MIXer:LO<n>:NAME <value>

(Read-Write) Sets or returns the name of the PNA internal source or external source to use as the LO in an FCA measurement.

See Remotely Specifying a Source Port.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1.

<n> LO stage number. Choose 1 or 2.

<value> (string) - LO Source name. Use Source:CAT? to return a list of valid source ports. An external source must be configured and selected to be valid. Learn more about external source configuration.

Examples SENS:MIX:LO:NAME "MySource"

Query Syntax SENSE<ch>:MIXer:LO<n>:NAME?

Return Type String

Default Not Controlled

SENSe<ch>:MIXer:LO<n>:POWer <value>
(Read-Write) Sets or returns the value of the LO Power.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.
- `<n>`: LO stage. Choose 1 or 2.
- `<value>`: LO Power in dBm

**Examples**

```
SENS:MIX:LO:POW 9
```

**Query Syntax**

```
SENSe<ch>:MIXer:LO<n>:POWer?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe<ch>:MIXer:OUTPut:FREQuency:FIXed <value>**

(Read-Write) Sets or returns the output fixed frequency of the mixer. [See Note](#)

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1.
- `<value>`: Output fixed frequency in Hertz.

**Examples**

```
SENS:MIX:OUTP:FREQ:FIX 5e9
```

**Query Syntax**

```
SENSe<ch>:MIXer:OUTPut:FREQuency:FIXed?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

**SENSe<ch>:MIXer:OUTPut:FREQuency:MODE <char>**
Sets or returns the Output sweep mode.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Output sweep mode. Choose either **FIXED** or **SWEPT**

**Examples**

```
SENS:MIX:OUT:FREQ:MODE FIXED
SEnSe2:MIXer:OUTput:FREQuency:MODE swept
```

**Query Syntax**

```
SENSe<ch>:MIXer:OUTPut:FREQuency:MODE?
```

**Return Type** Character

**Default** Fixed

**(Read-Write)** Sets or returns the value of the output sideband. [See Note](#)

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` **Boolean** Sideband value. Choose from
  - **LOW** - Low or Difference (-)
  - **HIGH** - High or Sum (+)

**Examples**

```
SENS:MIX:OUTP:FREQ:SIDe LOW
SEnSe2:MIXer:OUTPut:FREQ:SIDeband HIGH
```

**Query Syntax**

```
SENSe<ch>:MIXer:OUTPut:FREQuency:SIDeband?
```

**Return Type** Character

**Default** LOW

**(Read-Write)** Sets or returns the start frequency.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<value>` Frequency value in Hz.
(Read-Write) Sets or returns the Output start frequency of the mixer. See Note

**Parameters**

| <ch> | Any existing channel number. If unspecified, value is set to 1. |
| <value> | Output start frequency |

**Examples**

```
SENS:MIX:OUTP:FREQ:STAR 1e9
SENSe2:MIXer:OUTPut:FREQ:STARt 1000000000
```

**Query Syntax**

```
SENSe<ch>:MIXer:OUTPut:FREQuency:STARt?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

(Read-Write) Sets or returns the Output stop frequency of the mixer. See Note

**Parameters**

| <ch> | Any existing channel number. If unspecified, value is set to 1. |
| <value> | Output stop frequency |

**Examples**

```
SENS:MIX:OUTP:FREQ:STOP 1e9
SENSe2:MIXer:OUTPut:FREQ:STOP 1000000000
```

**Query Syntax**

```
SENSe<ch>:MIXer:OUTPut:FREQuency:STOP?
```

**Return Type**

Numeric

**Default**

Not Applicable

---

Last Modified:

- 6-Mar-2008 | Added note to page top
- 23-Jul-2007 | Clarified LO Name command
Sense:Mixer:ELO Commands

Controls the Mixer with Embedded LO setting.

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Embedded LO Settings
- Synchronizing the PNA and Controller
Note: The Embedded LO DIAGnostic commands read data from the various broadband and precise tuning sweeps, similar to the textual and graphical data that are available in the user interface.

SENS<ch>:MIXer:ELO:LO:DELTa <num>

(Read-Write) Sets and returns LO Frequency Delta. There is usually no need to set this value. Read this value to determine the difference between the LO Frequency that is stated in the Mixer dialog box and the last measured LO Frequency.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` LO Frequency delta in Hertz.

Examples

SENS:MIX:ELO:LO:DELT 10.3

Query Syntax

SENS<ch>:MIXer:ELO:LO:DELTa?

Return Type

Numeric

Default

Not Applicable

SENS<ch>:MIXer:ELO:LO:RESet

(Write-only) Resets the LO Delta Frequency to 0 (zero).

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1

Examples

SENS:MIX:ELO:LO:RES

sense2:mixer:elo:lo:reset

Query Syntax

Not Applicable

Default

Not Applicable

SENS<ch>:MIXer:ELO:NORMALize:POINt <num>
(Read-Write) Sets and returns the sweep data point around which to perform broadband and precise tuning.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Mixer Sweep data point. Choose a data point number, between 1 and the max number of data points in the sweep, that has the least amount of expected noise.

**Examples**

```
SENSe:MIX:ELo:STATe 1
SENSe:MIX:ELo:STATe 0
SENSe:MIX:ELo:STATe 1
SENSe:MIX:ELo:STATe 0
```

**Query Syntax**

`SENSe<ch>:MIXer:ELO:STATe?`

**Return Type**

Numeric

**Default** Center point in the sweep span

---

**SENSe<ch>:MIXer:ELO:STATE <bool>**

(Read-Write) Sets and returns the ON |OFF state of Embedded LO.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` ON | OFF state. Choose from
  - 0 - Embedded LO OFF
  - 1 - Embedded LO ON

**Examples**

```
SENSe:MIX:ELo:STATe 1
SENSe:MIX:ELo:STATe 0
SENSe:MIX:ELo:STATe 1
SENSe:MIX:ELo:STATe 0
```

**Query Syntax**

`SENSe<ch>:MIXer:ELO:STATE?`

**Return Type**

Boolean

**Default** OFF

---

**SENSe<ch>:MIXer:ELO:TUNing:IFBW <num>**
(Read-Write) Sets and returns the IF Bandwidth for Broadband and Precise tuning sweeps.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` IF Bandwidth

**Examples**

```
SENS:MIX:ELO:TUN:IFBW 10kHz
sense2:mixer:elo:tuning:ifbw 20e3
```

**Query Syntax**

`SENSe<ch>:MIXer:ELO:TUNing:IFBW?`

**Return Type**

Numeric

**Default**

30kHz

---

**SENSe<ch>:MIXer:ELO:TUNing:INTerval <num>**

(Read-Write) Sets and returns how often a tuning sweep is performed.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Tuning sweep interval

**Examples**

```
SENS:MIX:ELO:TUN:INT 2
sense2:mixer:elo:tuning:interval 1
```

**Query Syntax**

`SENSe<ch>:MIXer:ELO:TUNing:INTerval?`

**Return Type**

Numeric

**Default**

1

---

**SENSe<ch>:MIXer:ELO:TUNing:ITERations <num>**
(Read-Write) Sets and returns the maximum number of tuning iterations to achieve the precise tolerance.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Number of tuning iterations. Choose a number between 1 and 100.

**Examples**

```
SENS:MIX:ELO:TUN:ITER 5
sense2:mixer:elo:tuning:iterations 3
```

**Query Syntax**

```
SENSe<ch>:MIXer:ELO:TUNing:ITERations?
```

**Return Type**

Numeric

**Default**

5

---

**SENSe<ch>:MIXer:ELO:TUNing:MODE <char>**

(Read-Write) Sets and returns the method used to determine the embedded LO Frequency.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<char>`: Tuning mode. Choose from:
  - **BROadband**: Both broadband and precise tuning
  - **PRECise**: Precise tuning only
  - **NONE**: No tuning; just apply the LO Frequency Delta value.

**Examples**

```
SENS:MIX:ELO:TUN:MODE BRO
sense2:mixer:elo:tuning:mode precise
```

**Query Syntax**

```
SENSe<ch>:MIXer:ELO:TUNing:MODE?
```

**Return Type**

Character

**Default**

BROadband

---

**SENSe<ch>:MIXer:ELO:TUNing:RESet**
(Write-only) Resets the tuning parameters to their default values.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

Examples

SENS:MIX:ELO:TUN:RES
sense2:mixer:elo:tuning:reset

Query Syntax Not Applicable

Default Not Applicable

---

**SENSe<ch>:MIXer:ELO:TUNing:SPAN <num>**

(Read-Write) Sets and returns the frequency span for the broadband tuning sweep.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1

<num> Broadband frequency span in Hz.

Examples

SENS:MIX:ELO:TUN:SPAN 1e6
sense2:mixer:elo:tuning:span 1mhz

Query Syntax SENSe<ch>:MIXer:ELO:TUNing:SPAN?

Return Type Numeric

Default 3 MHz

---

**SENSe<ch>:MIXer:ELO:TUNing:TOLerance <num>**
(Read-Write) Sets and returns the tuning tolerance for precise tuning.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<num>`  Tuning tolerance in Hz. Choose a number between .001 and 1e3.

**Examples**

```
SENS:MIX:ELO:TUN:TOL .5
sense2:mixer:elo:tuning:tolerance 1
```

**Query Syntax**  SENSE<ch>:MIXer:ELO:TUN:TolERance?

**Return Type**  Numeric

**Default**  1 Hz

SENSe<ch>:MIXer:ELO:DIAGnostic:CLEar

(Write-only) Clears current diagnostic information.

**Parameters**

- `<ch>`  Any existing channel number. If unspecified, value is set to 1

**Examples**

```
SENS:MIX:ELO:DIAG:CLEar
sense2:mixer:elo:diagnostic:clear
```

**Query Syntax**  Not Applicable

**Default**  Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:STATus?
(Read-only) Returns a string that describes the result of the last tuning sweeps.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```
SENS:MIX:ELO:DIAG:STAT?
sense2:mixer:elo:diagnostic:status
```

**Return Type** String

**Default** Not Applicable

---

**SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep:COUNt?**

(Read-only) Returns the number of tuning sweeps used for the latest embedded LO measurement.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```
SENS:MIX:ELO:DIAG:SWEep:COUNT
sense2:mixer:elo:diagnostic:sweep:count?
```

**Return Type** Numeric

**Default** Not Applicable

---

**SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:DATA?**
SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:DATA?

(Read-only) Returns an array of data that describes the data retrieved from the specified tuning sweep.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNt? to find the number of sweeps taken.

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:MIX:ELO:DIAG:SWE2:DATA?</td>
</tr>
<tr>
<td>sense2:mixer:elo:diagnostic:sweep1:data?</td>
</tr>
</tbody>
</table>

**Return Type** Array

**Default** Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:LO:DELTa?

(Read-only) Returns the LO frequency delta from the specified tuning sweep.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNt? to find the number of sweeps taken.

**Examples**

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>sense2:mixer:elo:diagnostic:sweep1:lo:delta?</td>
</tr>
</tbody>
</table>

**Return Type** Numeric

**Default** Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:MARKer:ANNotation?
(Read-only) Returns the Y-axis marker value from the specified tuning sweep. This command assumes that a marker was used. Use SENS:MIX:ELO:DIAG:SWE:MARK:STATe? to confirm if a marker was used for the tuning sweep.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNt? to find the number of sweeps taken.

**Examples**

```
SENS:MIX:ELO:DIAG:SWE2:MARKer:ANN?
```

**Return Type** Numeric

**Default** Not Applicable

---

**SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:MARKer:POSition?**

(Read-only) Returns the X-axis marker position from the specified tuning sweep.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNt? to find the number of sweeps taken.

**Examples**

```
SENS:MIX:ELO:DIAG:SWE2:MARKer:POS?
sense2:mixer:elo:diagnostic:sweep1:marker:position?
```

**Return Type** Numeric

**Default** Not Applicable

---

**SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:MARKer:STATe?**
(Read-only) Returns whether or not a marker was used for the specified tuning sweep.

Parameters

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<n>`  Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNt? to find the number of sweeps taken.

Examples

```
sense2:mixer:elo:diagnostic:sweep1:marker:position?
```

Return Type  Numeric

Default  Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:PARameter?

(Read-only) Returns the name of the parameter of the specified tuning sweep.

Parameters

- `<ch>`  Any existing channel number. If unspecified, value is set to 1
- `<n>`  Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNt? to find the number of sweeps taken.

Examples

```
SENS:MIX:ELO:DIAG:SWE2:PAR?
sense2:mixer:elo:diagnostic:sweep1:parameter?
```

Return Type  String - either "VC21" or "B,1"

Default  Not Applicable

SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:TITLe?
(Read-only) Returns the title of the specified tuning sweep.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Tuning sweep number. Use `SENS:MIX:ELO:DIAG:SWE:COUNT?` to find the number of sweeps taken.

**Examples**

- `SENS:MIX:ELO:DIAG:SWE2:TITL?`
- `sense2:mixer:elo:diagnostic:sweep1:title?`

**Return Type** String

**Default** Not Applicable

---

**SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:X:ANNotation?**

(Read-only) Returns the X-Axis annotation of the specified tuning sweep.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<n>` Tuning sweep number. Use `SENS:MIX:ELO:DIAG:SWE:COUNT?` to find the number of sweeps taken.

**Examples**

- `sense2:mixer:elo:diagnostic:sweep1:x:annotation?`

**Return Type** String - either "Hz" or "s"

**Default** Not Applicable

---

**SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:X:STARt?**
(Read-only) Returns the start value of the specified tuning sweep.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNT? to find the number of sweeps taken.

**Examples**

<table>
<thead>
<tr>
<th>Example 1</th>
<th>Example 2</th>
</tr>
</thead>
</table>

**Return Type** Numeric

**Default** Not Applicable

---

**SENSe<ch>:MIXer:ELo:DIAGnostic:SWEep<n>:X:STOP?**

(Read-only) Returns the stop value of the specified tuning sweep.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1

<n> Tuning sweep number. Use SENS:MIX:ELO:DIAG:SWE:COUNT? to find the number of sweeps taken.

**Examples**

<table>
<thead>
<tr>
<th>Example 1</th>
<th>Example 2</th>
</tr>
</thead>
</table>

**Return Type** Numeric

**Default** Not Applicable

---

**SENSe<ch>:MIXer:ELO:DIAGnostic:SWEep<n>:Y:ANNotation?**
(Read-only) Returns Y-axis annotation value of the specified tuning sweep.

**Parameters**

- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<n>`: Tuning sweep number. Use `SENS:MIX:ELO:DIAG:SWE:COUNT?` to find the number of sweeps taken.

**Examples**

```
sense2:mixer:elo:diagnostic:sweep1:y:annotation?
```

**Return Type**

String - either "U" or "Phase"

**Default**

Not Applicable
**Sense:Multiplexer Commands**

Controls External Test Sets (N44xx, E5091A, "Z", and "H" series).

Click on a blue keyword to view the command details.

Red commands are superseded.

**See Also**

- See an example program using these commands.
- Learn about External Test Set Control
- Synchronizing the PNA and Controller
- SCPI Command Tree

**SENSe:MULTiplexer<id>:ADDRes <address>**

*(Read-Write)* Sets and returns the address for the external test set at the specified ID. This command should be immediately preceded by the **SENSe:MULT:TYPE** command.

**Parameters**

- `<id>` Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the **SENSe:MULT:TYPE** command.
- `<address>` Integer The test set address.

- For a GPIB test set (N44xx and some specials), this is the GPIB address.
- For a test set I/O test set (some specials), it is the position of the test set in the chain (starting at 0).
- For USB test sets (E5091A), the address is set by DIP switches on the rear panel.
Examples

SENS:MULT1:TYPE "Z5623A_K66" ' use K66 test set, and reference it through ID 1
SENS:MULT1:ADDR 0 ' first test set in sequence
' All subsequent commands using SENS:MULT1 will refer to this test set

Query Syntax

SENSe:MULTiplexer<id>:ADDResS?

Return Type

Numeric

Default

Not Applicable

SENSe<cnum>:MULTiplexer<id>:ALLPorts <string>

(Read-Write) Sets or gets the port selections for all available ports on the specified channel.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1.

<id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the SENSe:MULT:TYPE command.

<string> Comma-separated list of port selections, one for each port. Each port selection must correspond to one of the values returned by SENSe:MULT:PORT:CAT?.

Do NOT include + and -.

Examples

' for channel 5 and test set 1, set port 1 to T1, port 2 to A, port 3 to R2+, port 4 to R3-.
SENS5:MULT1:ALLP "T1,A,R2,R3 "

Query Syntax

SENSe<cnum>:MULTiplexer<id>:ALLPorts?

Return Type

STRING

Default

Not Applicable

SENSe:MULTiplexer:CATalog?

(Read-Only) Returns a comma-separated list of the external test sets models that are currently supported. Choose one of these items to send SENSe:MULT1:TYPE.

Examples

SENS:MULT:CAT?

Return Type

String

Default

Not Applicable
SENSe:MULTiplexer<id>:COUNT?

(Read-Only) Returns the total number of ports of the specified test set. Returns 0 if no test set is connected (GPIB test sets only).

Parameters

<id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the SENSe:MULT:TYPE command.

Examples

SENS:MULT1:COUN?
sense:multiplexer2:count?

Return Type Numeric

Default Not Applicable

SENSe:MULTiplexer<id>:DISPLAY[:STATE] <bool>

(Read-Write) Turns ON and OFF the display of the test set control status bar. This status bar indicates the test set that is being controlled and the current port mappings. This setting is turned ON automatically when the test set is enabled.

Parameters

<id> Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the SENSe:MULT:TYPE command.

<bool> ON(1) Turns ON the display.

OFF (0) Turns OFF the display.

Examples

SENS:MULT1:DISP 1
sense:multiplexer2:display:state on

Query Syntax SENSe:MULTiplexer<id>:DISPLAY[:STATE]?

Return Type Boolean

Default OFF (0)

SENSe:MULTiplexer<id>:INCount?
(Read-Only) Returns the number of input ports for the specified test set.

- For test sets such as the E5091A that do NOT use jumper cables to route the stimulus and response signals, this command returns the number of test set ports that can be connected to the PNA.
- For test sets that DO use jumper cables to route the stimulus and response signals, such as the N44xx, the return value is not valid.

**Parameters**

- `<id>`: Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the `SENSe:MULT:TYPE` command.

**Examples**

```
SENS3:MULT1:INC? ' returns the number of input ports for test set 1 on channel 3
```

**Return Type** Numeric

**Default** Not Applicable

---

**SENSe<cnum>:MULTiplexer:LABel <string>**

(Read-Write) Sets and returns the display label for the testset on the specified channel. The label appears in a status bar at the bottom of the PNA display when `SENSe:MULT:DISP` is set to TRUE.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<string>`: Display label text.

**Examples**

```
SENS3:MULT:LAB 'High-power output'
```

**Query Syntax**

```
SENSe<cnum>:MULTiplexer:LABel?
```

**Return Type** String

**Default** Not Applicable

---

**SENSe<cnum>:MULTiplexer<id>:OUTPut[:DATa] <num>**

---

2232
(Read-Write) Sets or returns the control line value for the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<id>` Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the `SENSe:MULT:TYPE` command.
- `<numr>` An integer specifying the decimal value of the control line. Values are obtained by adding weights from the following table that correspond to individual lines.

<table>
<thead>
<tr>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>128</td>
</tr>
</tbody>
</table>

**Note:**

- The E5091A interprets SENS:MULT1:OUTP 0 as all lines OFF.
- All "Z"and "H" series test sets interpret SENS:MULT1:OUTP 0 as all lines ON.

Refer to your test set documentation for setting control line values.

**Examples**

`SENS3:MULT1:OUTP 48`  'For Z5623A K64, lines 5 and 6 are OFF; all other lines are set to ON state.

**Query Syntax**

`SENSe<cnum>:MULTiplexer<id>:OUTPut[:DATa]?`

**Return Type**

Numeric

**Default**

Not Applicable

`SENSe:MULTiplexer<id>:PORT<pnum>CATalog?`
(Read-Only) Returns a comma-separated list of valid port selections for the specified port.

**Parameters**

- `<id>`: Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the `SENSe:MULT:TYPE` command.

- `<pnum>`: Integer - Input port number for which to return valid Output port selections. Read the number of input ports for the test set using `SENSe:MULT:INCount`?

**Examples**

```
SENSe:MULT1:PORT3:CAT?  ' returns the valid port selections for port 3
```

**Return Type**

String

**Default**

Not Applicable

---

**SENSe<cnum>:MULTiplexer<id>:PORT<logi>SELect <phys>**

(Write-Only) Sets and returns a port mapping for a single port. If this command creates a conflict with an existing port, the PNA will resolve the conflict.

**Note:** This command is currently not supported for the Z5623AK44.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, value is set to 1.

- `<id>`: Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the `SENSe:MULT:TYPE` command.

- `<logi>`: Integer - Logical port number.

- `<string>`: String - Physical port number.

**Examples**

```
SENSe:MULT1:PORT3:SEL "4"  'sets logical port 3 to physical port 4.
```

**Return Type**

String

**Default**

Not Applicable

---

**SENSe:MULTiplexer<id>:STATE <bool>**
(Read-Write) Enables and disables (ON/OFF) the port mapping and control line output of the specified test set.

If the specified test set is not connected or not ON, then setting State ON will report an error. All other properties can be set when the test set is not connected.

When this command is set to ON, then the display of the test set status bar (SENS:MULT:DISP) is also set to ON.

**Parameters**

- `<id>`: Id of the external test set. If unspecified, Id is assumed to be 1. Must be previously set by the SENSE:TYPE command.
- `<bool>`: ON(1) Enables test set control. OFF (0) Disables test set control.

**Examples**

```
SENS:MULT:STAT 1
sense2:multiplexer2:state on
```

**Query Syntax**

SENSe<cnum>:MULTiplexer<id>:STATe?

**Return Type**

Boolean

**Default**

OFF (0)

---

**SENSe<cnum>:MULTiplexer<id>:TSET9:OUTPut[:DATA] <data> Superseded**

**Note:** This command is replaced with SENS:MULT:OUTP

(Read-Write) Sets the control lines of the specified E5091A. Control lines, provided through a E5091A front panel connector, are used to control external equipment such as a part handler. See your E5091A documentation to learn more about control lines.

**Parameters**

- `<cnum>`: Channel number of the measurement. If unspecified, value is set to 1.
- `<id>`: Id of the E5091A test set. Choose from 1 or 2. Learn how to set ID value.
- `<data>`: Data value used to set control lines. Values are obtained by adding weights from the following table that correspond to individual lines. HIGH =1; LOW=0.
<table>
<thead>
<tr>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>128</td>
</tr>
</tbody>
</table>

0 - Sets all lines low
255 - Sets all lines high

**Examples**

The following sets line 3 and 4 high. All other lines low.

`SENS:MULT1:TSET9:OUTP 12`

**Query Syntax**

`SENSe<cnum>:MULTiplexer<id>:TSET9:OUTPut[:DATA]?`

**Return Type**

Numeric

**Default**

0

---

**SENSe<cnum>:MULTiplexer<id>:TSET9:PORT1 <char> Superseded**

**Note:** This command is replaced with `SENS:MULT:ALLPorts` which sets ALL ports to the specified outputs.

(Read-Write) Switches Port 1 of the specified E5091A to one of the available outputs.

**Parameters**

- `<cnum>` Any existing channel number; if unspecified, value is set to 1.
- `<id>` Id of the E5091A test set. Choose from 1 or 2. [Learn how to set ID value](#).
- `<char>` Output port to be switched to. Choose from:

  A

  T1 - (If Port 2 already is connected to T1, then Port 2 will be switched to T2.)

**Examples**

`SENS:MULT1:TSET9:PORT1 A`

**Query Syntax**

`SENSe<cnum>:MULTiplexer<id>:TSET9:PORT1?`
SENSe<ch>:MULtiplexer<id>:TSET9:PORT2 <char> Superseded

Note: This command is replaced with SENS:MULT:ALLPorts which sets ALL ports to the specified outputs.

(Read-Write) Switches Port 2 of the specified E5091A to one of the available outputs.

Parameters

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<id>` Id of the E5091A test set. Choose from 1 or 2. Learn how to set ID value.
- `<char>` Output port to be switched to. Choose from:
  - T1 - If Port 1 already is connected to T1, then Port 1 will be switched to A.
  - T2

Examples SENS:MULT1:TSET9:PORT2 T2

Query Syntax SENSe<ch>:MULtiplexer<id>:TSET9:PORT2?

Return Type Character

Default T1

SENSe<ch>:MULtiplexer<id>:TSET9:PORT3 <char> Superseded

Note: This command is replaced with SENS:MULT:ALLPorts which sets ALL ports to the specified outputs.

(Read-Write) Switches Port 3 of the specified E5091A to one of the available outputs.

Parameters

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<id>` Id of the E5091A test set. Choose from 1 or 2. Learn how to set ID value.
- `<char>` Output port to be switched to. Choose from:
  - R1 (R1+)
  - R2 (R2+)
**SENSe<cnm>:MULTiplexer<id>:TSET9:PORT4 <char> Superseded**

**Note:** This command is replaced with `SENSe:MULT:ALLPorts` which sets ALL ports to the specified outputs.

(Read-Write) Switches Port 4 of the specified E5091A to one of the available outputs.

**Parameters**

- `<cnm>` Any existing channel number; if unspecified, value is set to 1.
- `<id>` Id of the E5091A test set. Choose from 1 or 2. [Learn how to set ID value](#).
- `<char>` Output port to be switched to. Choose from:
  - **R1** (R1-)
  - **R2** (R2-)
  - **R3** (R3-) If option 007 (7port), R2 is selected.

**Examples**

```
SENSe:MULT1:TSET9:PORT4 R2
```

**Query Syntax**

`SENSe<cnm>:MULTiplexer<id>:TSET9:PORT4?`

**Return Type**

Character

**Default**

R1

---

`SENSe:MULTiplexer<id>:TYPe <name>`
(Read-Write) Loads a configuration file for the specified type of external test set.

This command should be immediately followed by the SENSE:MULT:ADDRess command.

**Parameters**

- **<name>** String  The name of the type of test set. Must be one of the items in the list returned by the SENSE:MULT:CATalog? query.

- **<id>** Id of the external test set. Set by this command. Use consecutive values starting at 1.

**Examples**

SENSe:MULT1:TYPE "Z5623AK66"  ' use K66 test set, and reference it through ID 1

**Query Syntax**

SENSe:MULTiplexer<id>:TYPe?

**Return Type**  String

**Default**  Not Applicable

---

Last Modified:

17-Aug-2007  Modified Data command for differences in active high and low
SENSe:NOISe Commands

Controls the Noise Figure configuration and calibration.

```
SENSe:NOISe:
   AVERage <num>
   | STATe <bool>
   BWIDth <num>
   CALibration:METHOD <string>
   ENR:FILename <string>
   GAIN <num>
   IMPedance:COUNt <num>
   SOURce:
   | CKIT <string>
   | CONNector <string>
   TEMPerature:AMBient <num>
   TUNer:
   | ID  <string>
   | INPut <string>
   | OUTPut <string>
```

Click on a blue keyword to view the command details.

Other Noise Figure SCPI commands

The calibration commands listed in this topic are supplemental to the Guided Cal commands.

- **CALC:CUSTom:DEFine** - creates a noise figure measurement.
- **CONTrol:NOISe:SOURce** or **OUTPut:MANual:NOISe[:STATe]** - turns the Noise Source ON and OFF.
- **SENSe:PATH:CONF:ELEMent[:STATe]** - sets the port 1 and port 2 noise switches.
- **SENS:CORR:ENR:CAL:** - manage ENR data - usually not necessary.
SYST:PREF:ITEM:SWIT:DEF - Sets the default setting of the Noise Tuner switch

See Also

- Example: Create and Cal a Noise Figure Measurement
- Learn about Noise Figure Application
- Synchronizing the PNA and Controller
- SCPI Command Tree

**SENSe<ch>:NOISe:AVERage <num>**

*(Read-Write)* Set and read the averaging factor for the noise receiver. [Learn more](#)

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Averaging value. Choose any number from 1 to 99.

**Examples**

```
SENSe:NOISe:AVER 20
sense:noise:average 10
```

**Query Syntax**

SENSe:NOISe:AVERage?

**Return Type**

Numeric

**Default**

1

**SENSe<ch>:NOISe:AVERage:STATe <bool>**

*(Read-Write)* Turns noise averaging ON and OFF.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<bool>` Averaging state. Choose from

0 - Noise averaging OFF

1 - Noise averaging ON

**Examples**

```
SENSe:NOISe:AVER:STAT 0
sense:noise:average:state 1
```

**Query Syntax**

SENSe:NOISe:AVERage:STATe?

**Return Type**

Boolean
**SENSe<ch>:NOISe:BWIDth[:RESolution] <num>**

(Read-Write) Set and read the bandwidth of the noise receiver. [Learn more](#)

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Bandwidth value. Choose from:
  
  800 KHz, 2 MHz, 4 MHz, 8 MHz, or 24 MHz.

  Or the numerical equivalent, such as 8e6 and so forth.

  If the value does not match one of these, it is rounded up to the next legal value.

**Examples**

- `SENS:NOIS:BWID 2e6`
- `sense:noise:bwidth:resolution 8mhz`

**Query Syntax**

`SENSe:NOISe:BWIDth[:RESolution]?

**Return Type**

Numeric

**Default**

4MHz

---

**SENSe<ch>:NOISe:CALibration:METHod <string>**

(Read-Write) Set and read the method for performing a calibration on a noise channel.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<string>` Calibration method. Note case-sensitive. Choose from:
  
  - "VectorFull"
  - "SParameter"

**Examples**

- `SENS:NOIS:CAL:METH "Vector"
- `sense:noise:calibration:method "SParameter"

**Query Syntax**

`SENSe:NOISe:CALibration:METHod?`

**Return Type**

String
**SENSe<ch>:NOISe:ENR:FILename <string>**

(Read-Write) Set and read the path and name of the ENR file associated with the noise source.

**Parameters**
- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<string>` Full path, filename, and extension of the ENR file.

**Examples**

```plaintext
SENSe:NOISe:ENR:FIL "c:\ProgramFiles\Agilent\Network Analyzer\Documents\ENR\346C.enr"

sense:noise:enr:filename "c:\ProgramFiles\Agilent\Network Analyzer\Documents\ENR\346C.enr"
```

**Query Syntax**
SENSe:NOISe:ENR:FILename?

**Return Type** String

**Default** Not applicable

---

**SENSe<ch>:NOISe:GAIN <num>**

(Read-Write) Set and read the amount of gain for the noise receiver.

**Parameters**
- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Gain value. Choose from:
  - **0** - Low gain; select if the gain of your DUT is relatively high (>35 dB).
  - **15** - Medium gain; select if the gain of your DUT is about average (20 dB to 45 dB).
  - **30** - High gain; select if the gain of your DUT is relatively low (<30 dB).

**Learn more about Noise Receiver Gain setting.**

If the value does not match one of these, it is rounded up to the next legal value.

**Examples**

```plaintext
SENSe:NOISe:GAIN 15
sense:noise:gain 0
```

**Query Syntax**
SENSe:NOISe:GAIN?
SENSe<ch>:NOISe:IMPedance:COUNt <num>

(Read-Write) Sets the number of impedance states to use during calibrated measurements.

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
<num> Number of impedance states to use. Choose between 4 and the maximum number allowed by the noise tuner device. The more states that are used, the more accurate, and slower, the measurement. If the specified number exceeds the capability of the device, the measurement will use the maximum number of states the device allows.

Examples

SENSe:NOISe:IMPedance:COUNt 5
SENSe:NOISe:IMPedance:COUNt 7

Query Syntax

SENSe:NOISe:IMPedance:COUNt?

Return Type Numeric

Default 4

SENSe<ch>:NOISe:SOURce:CKIT <string>

(Read-Write) Set and read the Cal Kit that will be used for the Noise Source adapter.

An adapter is always necessary to connect a 346C Noise Source to the PNA port 2. Select a Cal Kit that is the same type and gender as the noise source connector.

If the Noise Source mates directly to PNA port 2, then set this type to "None".

Parameters

<ch> Any existing channel number. If unspecified, value is set to 1
<string> Cal Kit. Case sensitive.

To read possible cal kit strings for the adapter:

- Change the port connector type to that of the noise source using:
  SENS:CORR:COLL:GUID:CONN:PORT<n>
- Then read the possible cal kit strings for that port using:
**SENSe<ch>:NOISe:SOURce:CONNector <string>**

*(Read-Write)* Set and read the Noise Source connector type and gender. The Agilent 346C has an "APC 3.5 male" connector.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1
- `<string>` Noise source connector type and gender. Case sensitive.

Use **SENSe:CORR:COLL:GUID:CONN:CAT?** to read possible connector strings.

**Examples**

```
SENSe:NOIS:SOUR:CONN "APC 3.5 male"

sense:noise:source:connector "APC 3.5 female"
```

**Query Syntax** SENSE:NOIS:SOURe:CONNector?

**Return Type** String

**Default** Not applicable
(Read-Write) Sets the temperature at which the current noise measurement is occurring.

**Parameters**
- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Ambient temperature in Kelvin.

**Examples**
- SENS:NOIS:TEMP:AMB 292
- sense:noise:temperature 289

**Query Syntax**: SENSE:NOIS:TEMP:AMBient?

**Return Type**: Numeric
**Default**: 295

---

**SENSe<ch>:NOISe:TUNer:ID <string>**

(Read-Write) Set and read the identity of the noise tuner. This is an ECal model and serial number string. To read the identities of the connected ECal modules, use
SENSe:CORRection:CKIT:ECAL:LIST? and
SENSe:CORRection:CKIT:ECAL<mod>:INFormation?

**Parameters**
- `<ch>`: Any existing channel number. If unspecified, value is set to 1
- `<string>`: ECal model and serial number string. The ECal module must be connected when this command is sent.

**Examples**
- SENS:NOIS:TUN:ID "N4691-60004 ECal 02822"
- sense:noise:tuner:id ""N4691-60004 ECal 02822"

**Query Syntax**: SENSE:NOIS:TUNer:ID?

**Return Type**: String
**Default**: Not applicable

---

**SENSe<ch>:NOISe:TUNer:INPut <string>**
**SENSe<ch>:NOISe:TUNer:INPut <string>**

(Read-Write) Sets and reads the port of the ECal noise tuner that is connected to the PNA SOURCE OUT.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1

<string> ECal port identifier. Case sensitive.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:NOIS:TUN:INP &quot;B&quot;</td>
<td>Set input port to B</td>
</tr>
<tr>
<td>sense:noise:tuner:input &quot;A&quot;</td>
<td>Set input port to A</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe:NOISe:TUNer:INPut?

**Return Type**

String

**Default**

"B"

---

**SENSe<ch>:NOISe:TUNer:OUTPut <string>**

(Read-Write) Sets and reads the port of the ECal noise tuner that is connected to the CPLR THRU.

**Parameters**

<ch> Any existing channel number. If unspecified, value is set to 1

<string> ECal port identifier. Case sensitive.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:NOIS:TUN:OUTP &quot;B&quot;</td>
<td>Set output port to B</td>
</tr>
<tr>
<td>sense:noise:tuner:output &quot;A&quot;</td>
<td>Set output port to A</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe:NOISe:TUNer:OUTput?

**Return Type**

String

**Default**

"A"

---

Last Modified:

21-Jun-2007   MX New topic


**Sense:Offset Commands Superseded**

**Note:** These commands are replaced by the Sense:FOM commands which include the features of the new FOM dialog. Although these old commands will continue to work, they can NOT be mixed with the new commands.

Sets the offset frequency functions, causing the stimulus and response frequencies to be different.

Click on a blue keyword to view the command details.

**See Also**

- [Example Programs](#)
- [Learn about Frequency Offset](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)

---

**SENSe<cnum>:OFFSet:CW <bool>**

(Read-Write) Turns stimulus CW Override mode ON or OFF. Use this setting to establish a fixed (CW) stimulus frequency while measuring the Response over a swept frequency range.

**Parameters**

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <bool> ON (or 1) - turns CW override ON.
  
  Off (or 0) - turns CW override OFF.

**Examples**

```
SENS:OFFS:CW ON
sense2:offset:cw off
```

**Query Syntax**

SENSe<cnum>:OFFSet:CW?

**Return Type**

Boolean

**Default**

OFF

---

**SENSe<cnum>:OFFSet:DIVisor <num>**
(Read-Write) Specifies (along with the multiplier) the value to multiply by the stimulus.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Divisor value. Range is 1 to 1000

**Examples**

```
SENS:OFFS:DIV 3
sense2:offset:divisor 2
```

**Query Syntax**

`SENSe<cnum>:OFFSet:DIVisor?`

**Return Type**

Numeric

**Default**

1

---

**SENSe<cnum>:OFFSet:MULTiplier <num>**

(Read-Write) Specifies (along with the divisor) the value to multiply by the stimulus.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Multiplier value. Range is +/- 1000. Negative multipliers cause the stimulus to sweep in decreasing direction. For mixer measurements, this would be for setups requiring the RF frequency to be less than LO frequency

**Examples**

```
SENS:OFFS:MULT 2
sense2:offset:multiplier 4
```

**Query Syntax**

`SENSe<cnum>:OFFSet:MULTiplier?`

**Return Type**

Numeric

**Default**

1

---

**SENSe<cnum>:OFFSet:OFFSet <num>**
(Read-Write) Specifies an absolute offset frequency in Hz. For mixer measurements, this would be the LO frequency.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Offset frequency. Range is +/- 1000 GHz. Offsets can be positive or negative

**Examples**

```
SENS:OFFS:OFFS 1GHz
sense2:offset:offset 1e9
```

**Query Syntax**

`SENS<cnum>:OFFSET:OFFSet?`

**Return Type** Numeric

**Default** 0 Hz

---

**SENSSe<cnum>:OFFSet:STARt?**

(Read-Only) Returns the response start frequency

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```
SENS:OFFS:STAR?
sense2:offset:start?
```

**Return Type** Numeric

**Default** Not applicable

---

**SENSSe<cnum>:OFFSet:[STATe] <bool>**

2250
Enables Frequency Offset Mode on ALL measurements that are present on the active channel. This immediately causes the source and receiver to tune to separate frequencies. The receiver frequencies are specified with the other SENS:OFFSet commands. To make the stimulus settings use the SENS:FREQ commands.

Tip: To avoid unnecessary errors, first make other offset frequency settings, then set Frequency Offset ON.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<bool>`: ON (or 1) - turns Frequency Offset ON.
  OFF (or 0) - turns Frequency Offset OFF.

Examples

- SENS:OFFS ON
- sense2:offset:state off

Query Syntax

SENSe<cnum>:OFFSet:[STATe]?

Return Type: Boolean

Default: OFF (0)

---

SENSe<cnum>:OFFSet:STOP?

(Read-Only) Returns the response stop frequency.

Parameters

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1

Examples

- SENS:OFFS:STOP
- sense2:offset:stop

Return Type: Numeric

Default: Not applicable
Sense:Path:Configuration Commands

Controls the path configuration settings.

SENSe:PATH:CONFiguration

- CATalog?
- DELete
- DTEXt
- ELEMent
  - CATalog?
  - [STAte]
  - VALue:CATalog?
- NAME
- SESelect
- STORe

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Path Configuration
- Synchronizing the PNA and Controller
- SCPI Command Tree

SENSe:PATH:CONFiguration:CATalog?

(Read-only) Returns a list of configuration names stored in the PNA.

<table>
<thead>
<tr>
<th>Examples</th>
<th>SENS : PATH : CONF : CAT?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Type</td>
<td>Comma-separated list of double-quoted strings</td>
</tr>
<tr>
<td>Default</td>
<td>Not Applicable</td>
</tr>
</tbody>
</table>

SENSe:PATH:CONFiguration:DELete <string>
(Write-only) Deletes the specified configuration name from the PNA. The factory configurations cannot be deleted. This is the only method of distinguishing a factory configuration from a user-named configuration.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;string&gt;</td>
<td>Configuration name to be deleted.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:PATH:CONF:DEL "MyMixer"
```

**Return Type**

Not Applicable

**Default**

Not Applicable

---

**SENSe:PATH:CONFiguration:DTEXt <string>**

(Read-Write) Write and read descriptive text associated with the configuration. This text is displayed in the path configuration dialog. Text is generally used to describe external connections that must be made manually to complete the configuration setup.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;string&gt;</td>
<td>Descriptive text enclosed in quotes. Double quotes are not allowed within the descriptive text.</td>
</tr>
</tbody>
</table>

**Examples**

```
SENS:PATH:CONF:DTEX "Connect J1 jumper on the rear panel."
```

**Query Syntax**

```
SENS<ch>:PATH:CONFiguration:DTEXt?
```

**Return Type**

String

**Default**

Not Applicable

---

**SENS<ch>:PATH:CONFiguration:ELEMent:CATalog?**
(Read-only) Returns the names of configurable elements as a comma-delimited list of strings. See a list of configurable elements and settings for various PNA models.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.

**Examples**

```plaintext
SENS:PATH:CONF:ELEM:CAT?
'returns
"Combiner", "Src1", "Src2"
```

**Default** Not Applicable

**SENSe<ch>:PATH:CONFiguration:ELEMent[:STATE] <elem>, <setting>**

(Read-Write) Write or read the setting of a specified element in the current configuration. See a list of configurable elements and settings for various PNA models.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<elem>` Name of the element for which a setting is to be made.
- `<setting>` Element setting. Use **SENS:PATH:CONF:ELEM:VAL:CAT?** to return a list of valid settings for the specified element.

**Examples**

```plaintext
SENS:PATH:CONF:ELEM "Combiner", "Normal"
```

**Query Syntax**

```plaintext
SENSe<ch>:PATH:CONFiguration:ELEMent? "Combiner"
```

Returns the current state of the Combiner element.

**Return Type** String

**Default** Not applicable

**SENSe<ch>:PATH:CONFiguration:ELEMent:VALue:CATalog? <element>**
(Read-only) Returns the list of valid settings that can be used with the specified element. See a [list of configurable elements and settings](#) for various PNA models.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<element>` String. Element name for which to return valid settings.

**Examples**

```
'returns "Normal", "Reversed"
```

**Default** Not Applicable

**SENSe<ch>:PATH:CONFiguration:NAME?**

(Read-only) Returns the name of the current configuration only if NO individual element settings had been changed since selecting or storing a configuration. When element settings change, the path configuration name is cleared.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.

**Examples**

```
SENS:PATH:CONF:NAME?
'returns "Default"
```

**Return Type** String

**Default** Not Applicable

**SENSe<ch>:PATH:CONFiguration:SELect <string>**
(Write only) Loads the named configuration onto the specified channel.
Use SENS:PATH:CONF:CAT? to return the configuration names that are stored on the PNA.

**Parameters**

- `<ch>`: Any existing channel number; if unspecified, value is set to 1.
- `<string>`: Configuration name. "Default" is the default factory configuration.

**Examples**

```
SENS:PATH:CONF:SEL 'default'
sense2:path:configuration:select "MyMixer"
```

**Query Syntax**
Not Applicable

**Default**
"Default"

---

**SENSe<ch>:PATH:CONFiguration:STORe <name>**

(Write only) Saves the path configuration currently associated with channel `<ch>` to the specified configuration name.

**Parameters**

- `<ch>`: Any existing channel number; if unspecified, value is set to 1.
- `<name>`: String. Configuration name. Factory configurations can NOT be overwritten. Specifying the name of a pre-defined factory configuration will result in an error.

**Examples**

```
SENS:PATH:CONF:STOR "MyMixer"
```

**Query Syntax**
Not Applicable

**Default**
Not Applicable

---

Last Modified:

14-Dec-6   MX New Topic
**Sense:Power Command**

**Learn about Receiver Attenuation**

SENSe<cnm>:POWer:ATTenuation <rcvr>,<num>

*(Read-Write)* Sets the attenuation level for the specified receiver.

**Note:** Attenuation cannot be set with Sweep Type set to Power

**Parameters**

- **<cnm>**  Any existing channel number. If unspecified, value is set to 1
- **<rcvr>**  Receiver to get attenuation. Choose from:
  - ARECuver - receiver A
  - BRECuver - receiver B
  - CRECuver - receiver C
  - DRECuver - receiver D

Receiver attenuation can NOT be set using *logical receiver notation*.

- **<num>**  Attenuation value in dB. To determine how many receiver attenuators, the maximum receiver attenuation, and attenuation step size, for a PNA model, see PNA Models and Options.

  If a number other than these is entered, the analyzer will select the next lower valid value. For example, if 19 is entered for the E8361A, then 10 dB attenuation will be selected.

**Examples**

SENSe:POW:ATT AREC,10
sense2:power:attenuation breceiver,30

**Query Syntax**

SENSe<cnm>:POWer:ATTenuation? <rc>

**Return Type**  Numeric

**Default**  0

---

Last Modified:

25-Oct-2007  Edit range of values
**Sense:Pulse Commands**

Configures the 5 pulse generators in the PNA-X.

![SENSe:PULSe]

Click on a blue keyword to view the command details.

**Pulse Definitions**

- **D** = Delay; the time before each pulse begins
- **W** = Width; the time the pulse is ON
- **P** = Period; one complete pulse cycle
- Duty Cycle = \( \frac{W}{P} \)

**Important:** If \( D + W \) is greater than \( P \), then undefined PNA behavior results. There is NO error message or warning.

**See Also**

- PNA-X IF Path Block diagram
- SENS:IF configuration commands
- Example Programs
- PNA-X Pulse Application
- Synchronizing the PNA and Controller
- SCPI Command Tree
SENSe<ch>:PULSe<n>:DELa y <value>

(Read-Write) Sets the pulse delay. The amount of time before a new pulse begins.

See Pulse Definition diagram.

Parameters

<ch> Any existing channel number; if unspecified, value is set to 1.

<n> Pulse generator number. Choose from 0 to 4.

0 is the generator that pulses the ADC.

<value> Delay value in seconds. Choose a value from about 33ns to about 70 seconds.

Examples

```
SENS:PUL:DEL .5
```

Query Syntax

```
SENSe<ch>:PULSe<n>:DELa y?
```

Return Type

Numeric

Default

0

SENSe<ch>:PULSe<n>:DINCrement <value>

(Read-Write) Sets the pulse delay increment. The delay increments with each pulse by the <value> amount.

For example, in this diagram the delay starts as 1. On the second pulse, delay=2. On the third pulse, delay=3.

**Important:** If \( D + W \) is greater than \( P \), then undefined PNA behavior results. There is NO error message or warning. Delay includes the incremented value.

This is useful for pulse profiling.

See Pulse Definition diagram.

Parameters

<ch> Any existing channel number; if unspecified, value is set to 1.
<n> Pulse generator number. Choose from 0 to 4.
0 is the generator that pulses the ADC.

<value> Delay increment value in seconds.

Examples
SENS:PULS:DINC .5

Query Syntax
SENSe<ch>:PULSe<n>:DINCrement?

Return Type Numeric

Default 0

SENSe<ch>:PULSe:PERiod <value>
(Read-Write) Sets the pulse-period (1/PRF) for ALL pulse generators.

The resolution of the period is 16.667nS.

See Pulse Definition diagram.

Parameters

<ch> Any existing channel number; if unspecified, value is set to 1.

<value> Pulse period in seconds. Choose a value from about 33ns to about 70 seconds.

Examples
SENS:PULS:PERiod .5

Query Syntax
SENSe<ch>:PULSe:PERiod?

Return Type Numeric

Default 1e-3 sec

SENSe<ch>:PULSe<n>:STATE <bool>
(Read-Write) Turns the pulse output ON and OFF.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Pulse generator number. Choose from 0 to 4.
  - 0 is the generator that pulses the ADC.
- `<bool>` ON (or 1) - turns pulse output ON.
  - OFF (or 0) - turns pulse output OFF.

**Examples**

```
SENS:PULS:STAT 1
```

**Query Syntax**

`SENSe<ch>:PULSe:STATe?`

**Return Type**

Boolean

**Default**

OFF

---

`SENSe<ch>:PULSe<n>:WIDTh <value>

(Read-Write) Sets the pulse width. The amount of time that the pulse is ON.

See Pulse Definition diagram.

**Parameters**

- `<ch>` Any existing channel number; if unspecified, value is set to 1.
- `<n>` Pulse generator number. Choose from 0 to 4.
  - 0 is the generator that pulses the ADC.
- `<value>` Pulse width in seconds. Choose a value from about 33ns to about 70 seconds.

**Examples**

```
SENS:PULS:WIDT .5
```

**Query Syntax**

`SENSe<ch>:PULSe<n>:WIDTh?`

**Return Type**

Numeric

**Default**

1e-4 sec

---

Last Modified:

2-Jan-2007    MX New topic
Sense:Roscillator Command

Learn about the Reference Osc.

**SENSe:ROSCillator:SOURce?**

*(Read-only)* Applying a signal to the Reference Oscillator connector automatically sets the Reference Oscillator to EXTernal. This command allows you to check that it worked. EXT is returned when a signal is present at the Reference Oscillator connector. INT is returned when NO signal is present at the Reference Oscillator connector.

<table>
<thead>
<tr>
<th>Examples</th>
<th>Sense:roscillator:source?</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Return Type</th>
<th>Character</th>
</tr>
</thead>
</table>

**Default** Not applicable
Sense:Segment Commands

Defines the segment sweep settings. Enable segment sweep with SENS:SWE:TYPE SEGment.

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Segment Sweep
- Synchronizing the PNA and Controller
- SCPI Command Tree

SENSe<cnum>:SEGMennt<snum>:ADD

(Write-only) Adds a segment.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <snum> Segment number to add. If unspecified, value is set to 1. Segment numbers must be sequential. If a new number is added where one currently exists, the existing segment and those following are incremented by one.

Examples

Two Segments exist (1 and 2). The following command will add a new segment (1). The existing (1 and 2) will become (2 and 3) respectively.

SENSe2:segment1:ADD
SENSe1:ADD

Query Syntax

Not applicable. Use Sense:Segment:Count to determine the number of segments in a trace.

Default Not Applicable

SENSe<cnum>:SEGMennt:ARBitrary <ON | OFF>
(Read-Write) Enables you to setup a segment sweep with arbitrary frequencies. The start and stop frequencies of each segment can overlap other segments. Also, each segment can have a start frequency that is greater than its stop frequency which causes a reverse sweep over that segment. Learn more about Arbitrary Segment Sweep.

**Parameters**

- `<cnum>`  
  Any existing channel number. If unspecified, value is set to 1.

- `<ON | OFF>`  
  ON (or 1) - Allows the setup of arbitrary segment sweep.  
  OFF (or 0) - Prevents the setup of arbitrary segment sweep.

**Examples**

```plaintext
SENS:SEGM:ARB ON  
sense2:segment:arbitrary off
```

**Query Syntax**  
SENSe<cnum>:SEGMent:ARBitrary?

**Return Type**  
Boolean (1 = ON, 0 = OFF)

**Default**  
OFF

---

**SENSe<cnum>:SEGMen<snum>:BWIDth[:RESolution] <num>**

(Read-Write) Sets the IF Bandwidth for the specified segment. First set SENS:SEGM:BWIDth:CONTrol ON. All subsequent segments that are added assume the new IF Bandwidth value.

**Parameters**

- `<cnum>`  
  Any existing channel number. If unspecified, value is set to 1

- `<snum>`  
  Segment number to modify. Choose any existing segment number.

- `<num>`  
  IF Bandwidth in Hz. The list of valid IF Bandwidths is different depending on the PNA model. (Click to see the lists.) If an invalid number is specified, the analyzer will round up to the closest valid number.

  **Note**: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```plaintext
SENS:SEGM:BWID 1KHZ  
sense2:segment2:bwidth:resolution max
```

**Query Syntax**  
SENSe<cnum>:SEGMent<snum>:BWIDth[:RESolution]?

**Return Type**  
Numeric

**Default**  
See Preset IFBW for your PNA model.

---

**SENSe<cnum>:SEGMent:BWIDth[:RESolution]:CONTrol <ON | OFF>**
Specifies whether the IF Bandwidth resolution can be set independently for each segment.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>`
  - **ON** (or 1) - turns Bandwidth control ON. Bandwidth can be set for each segment
  - **OFF** (or 0) - turns Bandwidth control OFF. Use channel bandwidth setting

**Examples**

- `SENS:SEG:M:BWID:CONT ON`
- `sense2:segment:bwidth:control off`

**Query Syntax**

`SENS<cnum>:SEG:M:BWID[:RESolution]:CONTrol?`

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

**SENS<cnum>:SEG:M:COUNt?**

(Read-only) Queries the number of segments that exist in the specified channel.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

- `SENS<cnum>:SEG:M:COUNt?`
- `sense2:segment:count?`

**Return Type**

Numeric

**Default**

1 segment

---

**SENS<cnum>:SEG:M<snum>:DEL**

(Write-only) Deletes the specified sweep segment. When ALL segments are deleted, Sweep Mode (`SENS:SWE:M:ODE`) is automatically set to Linear because there are no segments to sweep.

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Number of the segment to delete. If unspecified, value is set to 1

**Examples**

- `SENS<cnum>:SEG:M<snum>:DEL`
- `sense2:segment2:delete`

**Query Syntax**

Not applicable

**Default**

Not Applicable
SENSe<cnum>:SEGMent:DELete:ALL

(Write-only) Deletes all sweep segments. When this command is executed, Sweep Mode (SENS:SWE:MODE) is automatically set to Linear because there are no segments to sweep.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

Examples

- SENS:SEGM:DEL:ALL
- sense2:segment:delete:all

Query Syntax Not applicable

Default Not Applicable

SENSe<cnum>:SEGMent<snum>:FREQuency:CENTer <num>

(Read-Write) Sets the Center Frequency for the specified segment. The Frequency Span of the segment remains the same. The Start and Stop Frequencies change accordingly.

Note: All previous segment's Start and Stop Frequencies that are larger than the new Start Frequency are changed to the new Start Frequency. All following segment's start and stop frequencies that are smaller than the new Stop Frequency are changed to the new Stop Frequency.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Segment number to modify. Choose any existing segment number.
- `<num>` Center Frequency in Hz. Choose any number between the `minimum` and `maximum` frequency of the analyzer.

Note: This command will accept `MIN` or `MAX` instead of a numeric parameter. See SCPI Syntax for more information.

Examples

- SENS:SEGM:FREQ:CENT 1MHZ
- sense2:segment2:frequency:center 1e9

Query Syntax SENSe<cnum>:SEGMent<snum>:FREQuency:CENTer?

Return Type Numeric

Default Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

SENSe<cnum>:SEGMent<snum>:FREQuency:SPAN <num>
(Read-Write) Sets the Frequency Span for the specified segment. The center frequency of the segment remains the same. The start and stop frequencies change accordingly.

**Note:** All previous segment's Start and Stop Frequencies that are larger than the new Start Frequency are changed to the new Start Frequency. All following segment's start and stop frequencies that are smaller than the new Stop Frequency are changed to the new Stop Frequency.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Segment number to modify. Choose any existing segment number.
- `<num>` Frequency Span in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

  **Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```
SENS:SEG:M:FREQ:SPAN 1MHZ
sense2:segment2:frequency:span max
```

**Query Syntax**

SENSe<cnum>:SEGment<snum>:FREQuency:SPAN?

**Return Type**

Numeric

**Default**

If first segment, frequency span of the analyzer. Otherwise 0.

---

**SENSe<cnum>:SEGment<snum>:FREQuency:START <num>**

(Read-Write) Sets the Start Frequency for the specified sweep segment.

**Notes**

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

To return the start and stop frequency of the entire sweep (all segments), Use SENS:FREQ:START? and SENS:FREQ:STOP?

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Segment number to modify. Choose any existing segment number.
- `<num>` Start Frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

  **Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.
**SENSe<cnum>:SEGMent<snum>:FREQuency:STARt?**

(Read-Write) Sets the Start Frequency for the specified sweep segment.

**Notes**

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

To return the start and stop frequency of the entire sweep (all segments), use `SENS:FREQ:STARt?` and `SENS:FREQ:STOP?`.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Stop Frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

**Examples**

```
SENS:SEG:M:FREQ:STAR 1MHZ
sense2:segment2:frequency:start minimum
```

**Query Syntax**

`SENSe<cnum>:SEGMent<snum>:FREQuency:STARt?`

**Return Type**

Numeric

**Default**

Stop Frequency of the previous segment. If first segment, start frequency of the analyzer.

---

**SENSe<cnum>:SEGMent<snum>:FREQuency:STOP <num>**

(Read-Write) Sets the Stop Frequency for the specified sweep segment.

**Notes**

All other segment Start and Stop Frequency values that are larger than this frequency are changed to this frequency.

To return the start and stop frequency of the entire sweep (all segments), use `SENS:FREQ:STARt?` and `SENS:FREQ:STOP?`.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<snum>`: Segment number to modify. Choose any existing segment number.
- `<num>`: Stop Frequency in Hz. Choose any number between the minimum and maximum frequency of the analyzer.

**Examples**

```
SENS:SEG:M:FREQ:STOP 1MHZ
sense2:segment2:frequency:stop maximum
```

**Query Syntax**

`SENSe<cnum>:SEGMent<snum>:FREQuency:STOP?`

**Return Type**

Numeric

**Default**

If first segment, stop frequency of the analyzer. Otherwise, start frequency of the segment.

---

**SENSe<cnum>:SEGMent<snum>:POWer[port][:LEVel] <num>**
(Read-Write) Sets the Port Power level for the specified sweep segment. First set SENS:SEGM:POW:CONTrol ON.

When port power is Coupled, setting port power for one port will apply port power for all source ports.
All subsequent segments that are added assume the new Power Level value.

Parameters

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<snum>`  Segment number to modify. Choose any existing segment number.
- `<port>` Port number of the source. If unspecified, value is set to 1.
- `<num>`  Power level.

Note: The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, send SOUR:POW? MAX and SOUR:POW? MIN. (SOUR:POW:ATT:AUTO must be set to ON).

Actual achievable leveled power depends on frequency.

Examples

```
SENS:SEGM:POW 0
sense2:segment2:power1:level -10
```

Query Syntax

```
SENSe<cnum>:SEGMent<snum>:POWer[<port>][]:LEVel? # Query Syntax
```

Return Type

Numeric

Default

0

SENSe<cnum>:SEGMent:POWer[:LEVel]:CONTrol <ON | OFF>

(Read-Write) Specifies whether Power Level can be set independently for each segment.

Parameters

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>`  ON (or 1) - turns Power Level control ON. Power level can be set for each segment. OFF (or 0) - turns Power Level control OFF. Use the channel power level setting.

Examples

```
SENS:SEGM:POW:CONT ON
sense2:segment:power:level:control off
```

Query Syntax

```
SENSe<cnum>:SEGMent:POWer[:LEVel]:CONTrol? # Query Syntax
```

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF
SENSe<cnum>:SEGMent<snum>[:STATe] <ON | OFF>

(Read-Write) Turns the specified sweep segment ON or OFF. At least ONE segment must be ON or Sweep Mode is automatically set to Linear.

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<snum>`  Segment number to be turned ON or OFF
- `<ON | OFF>`  ON (or 1) - turns segment ON. OFF (or 0) - turns segment OFF.

**Examples**

```plaintext
SENS:SEG ON
sense2:segment2:state off
```

**Query Syntax**

SENSe<cnum>:SEGMent<snum>[:STATe]?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

SENSe<cnum>:SEGMent<snum>:SWEep:POINts <num>

(Read-Write) Sets the number of data points for the specified sweep segment.

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<snum>`  Any existing segment number. If unspecified, value is set to 1
- `<num>`  Number of points in the segment. The total number of points in all segments cannot exceed 20001. A segment can have as few as 1 point.

**Note:** This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

```plaintext
SENS:SEG:SWE:POIN 51
sense2:segment2:sweep:points maximum
```

**Query Syntax**

SENSe<cnum>:SEGMent<snum>:SWEep:POINts?

**Return Type**

Numeric

**Default**

21

SENSe<cnum>:SEGMent<snum>:SWEep:TIME <num>

2270
(Read-Write) Sets the time the analyzer takes to sweep the specified sweep segment.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <snum> Any existing segment number.
- <num> Sweep time in seconds. Choose a number between 0 and 100

Note: This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

Examples

```
SENS:SEGM:SWE:TIME 1ms
sense2:segment2:sweep:time .001
```

Query Syntax

SENSe<cnum>:SEGMent<snum>:SWEep:TIME?

Return Type

Numeric

Default

Not Applicable

SENSe<cnum>:SEGMent:SWEep:TIME:CONTrol <ON | OFF>

(Read-Write) Specifies whether Sweep Time can be set independently for each sweep segment.

Parameters

- <cnum> Any existing channel number. If unspecified, value is set to 1
- <ON | OFF> ON (or 1) - turns Sweep Time control ON. Sweep Time can be set for each segment. OFF (or 0) - turns Sweep Time control OFF. Uses the channel Sweep Time setting.

Examples

```
SENS:SEG:M:SWE:TIM:CONT ON
sense2:segment:swep:time:control off
```

Query Syntax

SENSe<cnum>:SEGMent:SWEep:TIME:CONTrol?

Return Type

Boolean (1 = ON, 0 = OFF)

Default

OFF

SENSe<cnum>:SEGMent<snum>:X:SPACing <char>
(Read-Write) Sets X-axis spacing ON or OFF

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<snum>` Any existing segment number. (This parameter is ignored)
- `<char>`
  - **LINear** - turns X-axis point spacing OFF
  - **OBASe** - turns X-axis point spacing ON

**Examples**

<table>
<thead>
<tr>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SENS:SEGM:X:SPACing LIN</code></td>
</tr>
<tr>
<td><code>sense2:segment1:spacing obase</code></td>
</tr>
</tbody>
</table>

**Query Syntax**

```
SENSe<cnum>:SEGMent<snum>:X:SPACing?
```

**Return Type**

Character

**Default**

LINear

---

**Last Modified:**

21-Jun-2007  Increased max number of points
**Sense:Sweep Commands**

Specifies the sweep functions of the analyzer.

```
SENSe:SWEep:
   DWELI
      | AUTO
   GENeration
   GROups
      | COUNT
   MODE
   POINts
   SRCPort
   TIME
      | AUTO
   TRIGger
      | DELAY
      | MODE
      | POIN
   TYPE
```

Click on a blue keyword to view the command details.

**See Also**

- [Example Programs](#)
- **Example** Triggering the PNA using SCPI
- [Learn about Sweeping](#)
- [Synchronizing the PNA and Controller](#)
- [SCPI Command Tree](#)
SENSe<cnum>:SWEep:DWELl <num>
(Read-Write) Sets the dwell time between each sweep point.

- Dwell time is **ONLY** available with SENSe:SWEep:GENeration set to **STEPped**; It is **Not** available in ANALOG.
- Sending dwell = 0 is the same as setting SENSe:SWEep:DWELl:AUTO ON. Sending a dwell time > 0 sets SENSe:SWEep:DWELl:AUTO OFF.

**Parameters**

- `<num>` Any existing channel number. If unspecified, value is set to 1
- `num` Dwell time in seconds.

This command will accept MIN or MAX instead of a numeric parameter. See SCPI Syntax for more information.

**Examples**

<table>
<thead>
<tr>
<th>SCPI Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:SWEep:DWELl .1</td>
</tr>
<tr>
<td>sense2:sweep:dwell min</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnum>:SWEep:DWELl?

**Return Type**

Numeric

**Default**

0 - (Note: dwell time set to 0 is the same as dwell:auto ON)

SENSe<cnum>:SWEep:DWELl:AUTO <ON | OFF>
(Read-Write) Specifies whether or not to automatically calculate and set the minimum possible dwell time. Setting Auto ON has the same effect as setting dwell time to 0.

**Parameters**

- `<num>` Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>` ON (or 1) - turns dwell ON.
  OFF (or 0) - turns dwell OFF.

**Examples**

<table>
<thead>
<tr>
<th>SCPI Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENSe:SWEep:DWELl:AUTO ON</td>
</tr>
<tr>
<td>sense2:sweep:dwell:dwel auto off</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnum>:SWEep:DWELl:AUTO?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

SENSe<cnum>:SWEep:GENeration <char>
(Read-Write) Sets sweep as Stepped or Analog.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:

  - **STEPped** - source frequency is CONSTANT during measurement of each displayed point. More accurate than ANALog. Dwell time can be set in this mode.

  - **ANALog** - source frequency is continuously RAMPING during measurement of each displayed point. Faster than STEPped. Sweep time (not dwell time) can be set in this mode.

**Examples**

```plaintext
SENS:SWEP:GEN STEP
sense2: sweep:generation analog
```

**Query Syntax**

`SENSe<cnum>:SWEP:GENERation?`

**Return Type** Character

**Default** Analog

---

**SENSe<cnum>:SWEP:GROups:COUNt <num>**

(Read-Write) Sets the trigger count (groups) for the specified channel. Set trigger mode to group after setting this count.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Count (groups) number. Choose any number between: 1 and `2e6` (1 is the same as single trigger)

**Examples**

```plaintext
SENS:SWEP:GRO:COUNT 10
sense2: sweep:groups:count 50
```

**Query Syntax**

`SENSe<cnum>:SWEP:GROups:COUNt?`

**Return Type** Numeric

**Default** 1

---

**SENSe<cnum>:SWEP:MODE <char>**
(Read-Write) Sets the number of trigger signals the specified channel will ACCEPT.

See Triggering the PNA Using SCPI.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Trigger mode. Choose from:
  
  **HOLD** - channel will not trigger
  
  **CONTinuous** - channel triggers indefinitely
  
  **GROups** - channel accepts the number of triggers specified with the last `SENS:SWE:GRO:COUN <num>`. This is one of the PNA overlapped commands. Learn more.

  **SINGle** - channel accepts ONE trigger, then goes to HOLD.

  **Note:** Beginning with 7.50, the SINGle argument makes it no longer necessary to use `SENS:SWE:GRO:COUN 1`.

**Examples**

```plaintext
SENS:SWE:MODE CONT
sense2:sweep:mode hold
```

**Query Syntax**

SENSe<cnum>:SWEep:MODE?

**Return Type**

Character

**Default**

CONTinuous

SENSe<cnum>:SWEep:POINts <num>
(Read-Write) Sets the number of data points for the measurement.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Choose any number between 1 and **20001**.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See **SCPI Syntax** for more information.

**Examples**

```
SENS:SWE:POIN 51
sense2:sweep:points max
```

**Query Syntax**

```
SENSe<cnum>:SWEep:POINts?
```

**Return Type**

Numeric

**Default**

201

---

**SENSe<cnum>:SWEep:SRCPort <1 | 2> Superseded**

This command is superseded. The **Calc:Par:Def:Ext** and **Calc:Par:Mod:Ext** can now optionally include the source port.

(Read-Write) Sets the source port when making non S-parameter measurements. Has no effect on S-parameter measurements.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<1 | 2>` 1 - Source power comes out Port 1
  2 - Source power comes out Port 2

**Examples**

```
SENS:SWE:SRCP 1
sense2:sweep:srcport 2
```

**Query Syntax**

```
SENSe<cnum>:SWEep:SRCPorT?
```

**Return Type**

Character

**Default**

1

---

**SENSe<cnum>:SWEep:TIME <num>**
(Read-Write) Sets the time the analyzer takes to complete one sweep. If sweep time accuracy is critical, use ONLY the values that are attained using the up and down arrows next to the sweep time entry box. See Sweep Time.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Sweep time in seconds. Choose a number between **0** and **86,400** (24hrs).

To select the fastest sweep speed, either send MIN as an argument to this command, or send SENS:SWE:TIME:AUTO 1.

This command will accept **MIN** or **MAX** instead of a numeric parameter. See SCPI Syntax for more information.

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:SWE:TIME 1ms</td>
</tr>
<tr>
<td>sense2:sweep:time .001</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnum>:SWEep:TIME?

**Return Type**

Numeric

**Default**

NA

---

**SENSe<cnum>:SWEep:TIME:AUTO <ON | OFF>**

(Read-Write) Turns the automatic sweep time function ON or OFF.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>` **ON** (or 1) - turns the automatic sweep time ON.
  **OFF** (or 0) - turns the automatic sweep time OFF.

<table>
<thead>
<tr>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENS:SWE:TIME:AUTO</td>
</tr>
<tr>
<td>sense2:sweep:time: auto off</td>
</tr>
</tbody>
</table>

**Query Syntax**

SENSe<cnum>:SWEep:TIME:AUTO?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

---

**SENSe<cnum>:SWEep:TRIGger:DELay <num>**
(Read-Write) Sets and reads the trigger delay for all measurements in the specified CHANNEL. This delay is only applied while TRIG:SOURce EXTernal and TRIG:SCOP CURRent. After an external trigger is applied, the start of the sweep is delayed for the specified delay value plus any inherent latency.

To apply a trigger delay for all channels (Global), use TRIG:DEL.

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Trigger delay value in seconds. Range is from 0 to 107 seconds

**Examples**
- `SENS:SWEP:TRIG:DELAY .003`
- `sense2:sweep:trigger:delay 1`

**Query Syntax** `SENSSe<cnum>:SWEPep:TRIGger:DELay?`

**Return Type** Numeric

**Default** 0

**SENSe<cnum>:SWEPep:TRIGger:MODE <char>**

(Read-Write) Sets and reads the trigger mode for the specified channel. This determines what EACH signal will trigger. Learn more.

**Note:** Setting Point and Sweep mode forces Trigger:SCOPe = CURRent

**Parameters**
- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Trigger mode. Choose from:
  - CHANNEL - Each trigger signal causes ALL traces in that channel to be swept.
  - SWEEP - Each Manual or External trigger signal causes ALL traces that share a source port to be swept.
  - POINT - Each Manual or External trigger signal causes one data point to be measured.

**Examples**
- `SENS:SWEP:TRIG:MODE SWEEP`
- `sense2:sweep:trigger:mode point`

**Query Syntax** `SENSSe<cnum>:SWEPep:TRIGger:MODE?`

**Return Type** Character

**Default** Channel
SENSe<cnum>:SWEep:TRIGger:POINt <ON | OFF> Superseded

This command is replace with SENS:SWE:TRIG:MODE POINT

(Read-Write) Specifies whether the specified channel will measure one point for each trigger or all of the measurements in the channel. Setting any channel to POINt mode will automatically set the TRIGger:SCOPe = CURRent.

Parameters

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<ON | OFF>** ON (or 1) - Channel measures one data point per trigger.
  
  OFF (or 0) - All measurements in the channel made per trigger.

Examples

```
SENS:SWE:TRIG:POIN ON
sense2:sweep:trigger:point off
```

Query Syntax

SENSe<cnum>:SWEep:TRIGger:POINt?

Return Type

Boolean (1 = Point, 0 = Measurement)

Default

0 - Measurement

---

SENSe<cnum>:SWEep:TYPE <char>

(Read-Write) Sets the type of analyzer sweep mode.

Parameters

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<char>** Choose from:
  
  LINear | LOGarithmic | POWer | CW | SEGMen
t

Note: SWEep TYPE cannot be set to SEGMen if there are no segments turned ON. A segment is automatically turned ON when the analyzer is started.

Examples

```
SENS:SWE:TYPE LIN
sense2:sweep:type segment
```

Query Syntax

SENSe<cnum>:SWEep:TYPE?

Return Type

Character

Default

LINear

---

Last Modified:
<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>21-Jun-2007</td>
<td>Increased max number of points</td>
</tr>
<tr>
<td>18-Jun-2007</td>
<td>Added Single to Mode</td>
</tr>
<tr>
<td>24-Apr-2007</td>
<td>Clarified Sweep mode</td>
</tr>
</tbody>
</table>
**X Values Command**

**SENSe<cnum>:X[:VALues]?**

*(Read-only)* Returns the stimulus values for the specified channel. If the channel is sweeping the source backwards, the values will be in descending order.

**Parameters**

- `<cnum>`  
  Any existing channel number; if unspecified, value is set to 1.

**Examples**

- `SENSE:X?`
- `sense2:x:values?`

**Return Type**  
Depends on FORM:DATA command

**Default**  
Not applicable
Source Commands

Controls the power delivered to the DUT.

SOURce

  CATalog?

  POWER

   ALC[:MODE]

   CATalog?

   ATTenuation

   AUTO

   CENTer

   CORRection (More)

   COUPle

   DETector

   [LEVEL]

     [IMMEDIATE][AMPLitude]

     SLOPe

     STATe

   MODE

   SPAN

   START

   STOP

SOURce<cnum>:CATalog?

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Power Settings
- Synchronizing the PNA and Controller
- SCPI Command Tree
- Remotely Specifying a Source Port
(Read-Write) Returns a list of valid port names that can be controlled. Some ports only have string names, NOT numbers. All commands that require a port argument have provisions for specifying either a port number OR a string name.

See also: Remotely Specifying a Source Port.

Parameters

Examples

'Some PNA-X models return "Port 1,Port 2,Port 3,Port 4,Port 1 Src2"

Return Type
Comma-separated list of strings.

Default
Not applicable

SOURce<cnum>:POWer<port>:ALC[:MODE]:CATalog?, [src]

(Read-Write) Returns a list of valid ALC modes for the specified channel and port number. Use the returned values to set SOUR:POW:ALC:MODE.

Parameters

Examples

'PORT 1,PORT 2,PORT 3,PORT 4,PORT 1 SRC2"

Return Type
Comma-separated list of strings.

Default
Not applicable

SOURce<cnum>:POWer<port>:ALC[:MODE] <char>, [src]
(Read-Write) Sets and returns the ALC mode for the specified channel and port. Use SOUR:POW:ALC:MODE:CAT? to return a list of valid ALC modes for the PNA.

Learn more about ALC mode.

Parameters
<cnun> Any existing channel number. If unspecified, value is set to 1
<port> Port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
<char> ALC Mode.

For the PNA-X choose from:

- **INTernal** Standard ALC loop
- **OPENloop** No ALC loop


While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples
SOUR:POW:ALC INT
source2:power2:alc:mode openloop
source:power:alc:mode openloop,"Port 1 Src2"

Query Syntax
SOURce<cnun>:POWer<port>:ALC:MODE? [src]

Return Type
Character

Default
INTernal

---

**SOURce<cnun>:POWer<port>:ATTenuation <num>, [src]**

(Read-Write) Sets the attenuation level for the selected channel. Sending this command turns automatic attenuation control (SOUR:POW:ATT:AUTO) to OFF. If the ports are coupled, changing the attenuation on one port will also change the attenuation on all other ports. To turn port coupling OFF use SOURce:POWer:COUPle OFF.

**Note:** Attenuation cannot be set with Sweep Type set to Power
See **Sens:Power:ATT** to change receiver attenuation.

### Parameters

| <cnum> | Any existing channel number. If unspecified, value is set to 1 |
| <port> | Port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument. |
| <num> | Attenuation value. The range of settable values depends on the PNA model. To determine the valid settings, do one of the following: |

- See [PNA models and options](#) to see the range and step size for each model / option.
- Perform a query using MAX, then MIN, as an argument. Example: SOURce:POWer:ATT? Max However, this will not tell you the attenuation step size.

If an invalid attenuation setting is entered, the PNA will select the next lower valid value. For example, if 19 is entered, then for an E8361A, 10 dB attenuation will be selected.

**Note:** This command will accept **MIN** or **MAX** instead of a numeric parameter. See [SCPI Syntax](#) for more information.


While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

### Examples

SOUR:POW:ATT 10

source2:power2:attenuation maximum

source:power:att 20, "Port 1 Src2"

### Query Syntax

SOURce<cnum>:POWer<port>:ATTenuation? [min/max] [src] [min/max,src]

### Return Type

Numeric

**Default** 0

SOURce<cnum>:POWer<port>:ATTenuation:AUTO <bool>, [src]
(Read-Write) Turns automatic attenuation control ON or OFF. Setting an attenuation value (using SOURce:POWer:ATTenuation <num>) sets AUTO OFF.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<port>` Port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<bool>` **ON** (or 1) - turns coupling ON. The analyzer automatically selects the appropriate attenuation level to meet the specified power level.
- **OFF** (or 0) - turns coupling OFF. Attenuation level must be set using SOURce:POWer:ATTenuation <num>.

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```plaintext
SOUR:POW2:ATT:Auto On
source2:power:attentuation:auto off
sour:pow:att:auto 1, "Port 1 Src2"
```

**Query Syntax** SOURce<cnum>:POWer:ATTenuation:Auto? [src]

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** ON

---

**SOURce<cnum>:POWer:CENTer <num>**
(Read-Write) Sets the power sweep center power. Must also set: SENS:SWE:TYPE:POWer and SOURce:POWer:SPAN <num>.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Center power. Actual achievable leveled power depends on frequency.

**Examples**

```
SOUR:POW:CENT -15
source2:power:center -7
```

**Query Syntax**

SOURce<cnum>:POWer:CENTer?

**Return Type**

Numeric

**Default**

0 dBm

---

**SOURce<cnum>:POWer:COUPle <ON | OFF>**

(Read-Write) Turns Port Power Coupling ON or OFF.

**Parameters**

- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<ON | OFF>`: ON (or 1) - turns coupling ON. The same power level is used for both source ports.
- OFF (or 0) - turns coupling OFF. Power level can be set individually for each source port.

**Examples**

```
SOUR:POW:COUP ON
source2:power:couple off
```

**Query Syntax**

SOURce<cnum>:POWer:COUPle?

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

ON

---

**SOURce<cnum>:POWer:DETector <char> OBSOLETE**
The PNA models with external leveling are now OBSOLETE.

Sets the source leveling loop as Internal or External.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` **INTernal** - Internal leveling is applied to the source
- **EXTERNal** - External leveling is applied to the source through a rear-panel connector. ONLY provided on 3 GHz, 6 GHz, and 9 GHz PNA models.

**Examples**

```
SOUR:POW:DET INT
source2:power:detector external
```

**Query Syntax**

`SOURce<cnum>:POWer<port>[:LEVel][:IMMediate][:AMPLitude] <num>, [src]`

(Read-Write) Sets the RF power output level.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Port number of the PNA. If unspecified, value is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- `<num>` Source power in dBm.

**Note:** The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. (SOUR:POW:ATT:AUTO must be set to ON) Example: SOURce:POWer? Max

Actual achievable leveled power depends on frequency.


While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the `<port>` argument performs the same function. If both arguments are specified, [src] takes priority.
**Examples**

```
SOUR:POW1 5DB
source2:power:level:immediate:amplitude maximum
sour:pow 5, "Port 1 Src2"
```

**Query Syntax**

```
SOURce<cnum>:POWer[:LEVel][:IMMediate][:AMPLitude]? [src]
```

**Return Type**

Numeric

**Default**

0 dBm

---

**SOURce<cnum>:POWer[:LEVel]:SLOPe <num>**

*(Read-Write)* Sets the RF power slope value.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Slope value in db/GHz. Choose any integer between -2 and 2 (0 is no slope).

**Examples**

```
SOUR:POW:SLOP 2
source2:power:slope -2
```

**Query Syntax**

```
SOURce<cnum>:POWer[:LEVel]:SLOPe?
```

**Return Type**

Numeric

**Default**

0

---

**SOURce<cnum>:POWer[:LEVel]:SLOPe:STATe <ON | OFF>**

*(Read-Write)* Turns Power Slope ON or OFF.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<ON|OFF>`
  - **ON** (or 1) - turns slope ON.
  - **OFF** (or 0) - turns slope OFF.

**Examples**

```
SOUR:POW:SLOP:STAT ON
source2:power:slope:state off
```

**Query Syntax**

```
SOURce<cnum>:POWer[:LEVel]:SLOPe:STATe?
```

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF

---

**SOURce<cnum>:POWer<port>:MODE <state>, [src]**
(Read-Write) Sets the state of PNA source for the specified port.

Learn more about the PNA internal second source

**Parameters**

- **<cnum>** Any existing channel number. If unspecified, value is set to 1
- **<port>** Port number of the PNA. If unspecified, <port> is set to 1. To make settings for ports that are not simple numbers, use the [src] argument.
- **<state>** Source state. Choose from:
  - **AUTO** Source power is turned ON when required for a measurement.
  - **ON** Source power is always ON regardless of the measurement.
  - **OFF** Source power is always OFF regardless of the measurement.

**Note:** ON and OFF are valid only on PNA models with two sources.


While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

- **SOUR:POW:MODE ON**
- **source2:power4:mode OFF**
- **sour:pow:mode 1, "Port 1 Src2"**

**Query Syntax**

SOURce<cnum>:POWer<port>:MODE? [src]

**Return Type**

Character

**Default**

Auto
(Read-Write) Sets the power sweep span power. Must also set:

**SENS:SWE:TYPE POWer** and **SOURce:POWer:CENTer <num>**.

**Parameters**
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1.
- `<num>`: Span power. Actual achievable leveled power depends on frequency.

**Examples**
- `SOUR:POW:SPAN -15`
- `source2:power:span -7`

**Query Syntax**
- `SOURce<cnum>:POWer:SPAN?`

**Return Type**
- Numeric
- **Default** 0 dBm

---

**SOURce<cnum>:POWer:STARt <num>**

(Read-Write) Sets the power sweep start power. Must also set

**SENS:SWE:TYPE POWer** and **SOURce:POWer:STOP <num>**.

**Parameters**
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<num>`: Start power.

**Note:** The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. (**SOUR:POW:ATT:AUTO** must be set to ON) Example: `SOURce:POWer:STARt? MIN`

**Examples**
- `SOUR:POW:STAR -15`
- `source2:power:start -7`

**Query Syntax**
- `SOURce<cnum>:POWer:START?`

**Return Type**
- Numeric
- **Default** 0 dBm

---

**SOURce<cnum>:POWer:STOP <num>**
(Read-Write) Sets the power sweep stop power. Must also set: SENS:SWE:TYPE POWer and SOURce:POWer:START <num>.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<num>` Stop power.

**Note:** The range of settable power values depends on the PNA model and if source attenuators are installed. To determine the range of values, perform a query using MAX, then MIN, as an argument. (SOUR:POW:ATT:AUTO must be set to ON) Example: SOURce:POWer:STOP? MAX

Actual achievable leveled power depends on frequency.

**Examples**

```plaintext
SOUR:POW:STOP -15
source2:power:stop -7
```

**Query Syntax**

SOURce<cnum>:POWer:STOP?

**Return Type**

Numeric

**Default**

0 dBm

---

Last modified:

- 25-Oct-2007 Edit test for source and rec attenuators commands
- 27-Jun-2007 Edited wording on Source:Cat?
- 10/18/06 MQQ Added Mode command

2293
Source:Power:Correction Commands

Used to perform source power calibration on internal and external sources.

Click on a blue keyword to view the command details.

See Also

- Example program using these commands.
- Template for creating your own Power Meter Driver
- Learn about Source Power Cal
- Synchronizing the PNA and Controller
- SCPI Command Tree

Note: The SOURce:POWer:CORRection:COLLect:ACQuire command, used to step the PNA and read a power meter, cannot be sent over the GPIB unless the power meter is connected to a different GPIB interface. See the alternative methods described in the command details.

SOURce<cnum>:POWer<port>:CORRection:COLLect:ABORt
(Write-only) Aborts a source power calibration sweep that is in progress.

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<port>`  Port number to correct for source power. If unspecified, value is set to 1.

**Examples**

```
SOUR:POW:CORR:COLL:ABOR
source1:power2:correction:collect:abort
```

**Query Syntax**
Not Applicable

**Default**
Not Applicable

---

**SOURce<cnum>:POWer<port>:CORRection:COLLect[:ACQuire] <char>,<id>[,src]**

**Note:** With PNA Rev. 6.2, a new "id" argument has been added to this command, replacing **SOUR:POW:CORR:COLL:METH**.

(Write-only) Initiates a source power cal acquisition sweep using the power sensor attached to the specified channel (A or B) on the power meter, or using the specified PNA receiver.

**Note:** This command, when used with a power meter, cannot be sent over the GPIB unless the power meter is connected to a different GPIB interface. Use one of the following methods to perform this command or its equivalent:

- If present, use the **GPIB dedicated controller port**.
- Connect the power meter to the PNA using a **USB / GPIB interface** (Agilent 82357A).
- SCPI programming of the PNA using a LAN Client interface (**see example**)
- Send SCPI commands through the COM interface using the **SCPI String Parser** object.
- Directly control the Power Meter and PNA to step frequency; then acquire and store the Power reading. (**see example**).

**Parameters**

- `<cnum>`  Any existing channel number. If unspecified, value is set to 1
- `<port>`  Port number to correct for source power. If unspecified, value is set to 1.
- `<char>`  The following choices are new beginning PNA Rev 6.2. Choose from:
  - **PMETer** - Power Meter is used for all readings.
  - **PMReceiver** - Power meter for the first iteration; then use the reference receiver for remaining readings if necessary (same as "fast iteration" box checked on
dialog box

- RECeiver - Use PNA measurement receiver for all readings.

The following choices are superseded but still supported, in conjunction with SOUR:POW:CORR:COLL:METH.

- ASENSor - Sensor on power meter channel A
- BSENSor - Sensor on power meter channel B

<id> String. Not case sensitive. The power sensor or PNA receiver to use for measuring power.

- For PMETer or PMRECeiver: select "ASENSOR" or "BSENSOR".
- For RECeiver: select a PNA receiver to acquire readings using physical or logical receiver notation.
- For ASENSor or BSENSor, this argument is NOT required.


While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:CORR:COLL PMET  'acquires power meter readings using the A sensor.
source1:power2:correction:collect:acquire receiver,"a1"
  'acquires source cal readings using the reference receiver for port 1.
```

Query Syntax Not Applicable

Default Not Applicable

SOURce:POWer:CORRection:COLLect:AVERage[:COUNT] <num>
This command, along with SOUR:POW:CORR:COLL:AVER:NTOLerance, allows for settling of the power sensor READINGS.

Specifies the maximum number of power readings that are taken at each stimulus point to allow for measurement settling. Each reading is averaged with the previous readings at that stimulus point.

When this average meets the Average:NTOLerance value or this number of readings has been made, the average is returned as the valid reading.

This setting is not necessary when using a PNA receiver (SOUR:POW:CORR:COLL REC) to make the measurement.

Learn more.

**Parameters**

- `<num>`: Maximum number of readings to make to allow for settling. Choose any number between 3 and 25.

**Examples**

- SOUR:POW:CORR:COLL:AVER 2
- source:power:correction:collect:average:count 3

**Query Syntax**

SOURce:POWer:CORRection:COLLect:AVERage[:COUNt]?

**Return Type**

- Numeric

**Default**

3

---

SOURce:POWer:CORRection:COLLect:AVERage:NTOLerance `<num>`

(Read-Write) This command, along with SOUR:POW:CORR:COLL:AVER:COUNT, allows for settling of the power sensor READINGS.

Each power reading is averaged with the previous readings at each stimulus point. When the average meets this nominal tolerance value or the max number of readings has been made, the average is returned as the valid reading.

This setting is not necessary when using a PNA receiver (SOUR:POW:CORR:COLL REC) to make the measurement.

Learn more.

**Parameters**

- `<num>`: Power measurement settling tolerance value in dB. Choose any number between 0 and 5.

**Examples**

- SOUR:POW:CORR:COLL:AVER:NTOL .05
- source1:power2:correction:collect:average:ntolerance .003
**Query Syntax**  
`SOURce:POWer:CORRection:COLLect:AVERage:NTOLerance?`  

**Return Type**  
Numeric  

**Default**  
.050 dBm  

---

**SOURce<cnum>:POWer:CORRection:COLLect:DISPlay[:STATe] <ON | OFF>**  
(Read-Write) Enables and disables the display of power readings on the PNA screen. Send this command BEFORE you begin a source power cal acquisition. After the source power cal data is acquired, this setting is reset to ON.

**Parameters**

- `<cnum>`  
  Any existing channel number. If unspecified, value is set to 1

- `<ON|OFF>`  

  - **ON (1)**  
    Source power calibration dialog box is shown on the PNA screen. Power readings are plotted against the Tolerance value as limit lines.

  - **OFF (0)**  
    Source power calibration dialog box is NOT shown on the PNA screen.

**Examples**

```plaintext
SOUR:POW:CORR:COLL:DISP ON
source1:power2:correction:collect:display:state off
```

**Query Syntax**  
`SOURce:POWer:CORRection:COLLect:DISPlay[:STATe]?`  

**Return Type**  
Boolean (1 = ON, 0 = OFF)  

**Default**  
ON (1)  

---

**SOURce<cnum>:POWer:CORRection:COLLect:FCHeck[:STATe] <ON | OFF>**  
(Read-Write) Enables and disables frequency checking of source power cal acquisition sweeps. ONLY use when you have more than one power sensor.

**Parameters**

- `<cnum>`  
  Any existing channel number. If unspecified, value is set to 1

- `<ON|OFF>`  

  - **ON (1)**  
    turns source power cal frequency checking ON. A requested acquisition will only succeed for those frequency points which fall within a frequency range specified for the power sensor being used. An acquisition will pause in mid-sweep if the frequency is about to exceed the maximum frequency limit specified for that sensor. When the sweep is paused in this manner, a sensor connected to the other channel input of the power meter can be connected to the measurement port in place of the previous sensor, and used to complete the sweep. However, the maximum frequency specified for the

2298
second sensor would need to be sufficient for the sweep to complete. Frequency limits are specified using the `SOUR:POW:CORR:COLL:SEN` command.

**OFF (0)** - turns source power cal frequency checking OFF. An acquisition will use just one power sensor for the entire sweep, regardless of frequency.

**Examples**
```
SOUR:POW:CORR:COLL:FCH ON
source1:power2:correction:collect:fcheck:state off
```

**Query Syntax** `SOURce:POWer:CORRection:COLLect:FCheck[:STATe]?

**Return Type** Boolean (1 = ON, 0 = OFF)

**Default** OFF (0)

---

`SOURce:POWer:CORRection:COLLect:ITERation[:COUNt] <num>`

(Read-Write) This command, along with `SOUR:POW:CORR:COLL:ITER:NTOL` describes the number of adjustments to make to the source power.

Sets the maximum number of readings to take at each data point for iterating the source power. Power readings (performed by `SOUR:POW:CORR:COLL:AVER:COUNT` and `SOUR:POW:CORR:COLL:AVER:NTOLerance`) will continue to be made, and source power adjusted, until the measurement is within the iteration tolerance value or this max number of measurements has been met. The last value is the valid measurement for that data point.

*Learn more.*

**Parameters**

|--<num>| Maximum number of readings. Choose any number between 1 and 100. |

**Examples**
```
SOUR:POW:CORR:COLL:ITER 2
source:power:correction:collect:iteration 3
```

**Query Syntax** `SOURce:POWer:CORRection:COLLect:ITERation[:COUNt]?`

**Return Type** Numeric

**Default** 1

---

`SOURce:POWer:CORRection:COLLect:ITERation:NTOLerance <num>`
This command, along with SOUR:POW:CORR:COLL:ITER:COUNT, describes the number of adjustments to make to the source power.

Sets the maximum desired deviation from the sum of the test port power and the offset value. Power READINGS (performed by SOUR:POW:CORR:COLL:AVER:COUNT and SOUR:POW:CORR:COLL:AVER:NTOLerance) will continue to be made, and source power adjusted, until a measurement is within this tolerance value or the max number of measurements has been met. The last value is the valid measurement for that data point.

Learn more.

## Parameters
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Port number to correct for source power. If unspecified, value is set to 1.
- `<num>`: Tolerance value in dBm. Choose any number between 0 and 5

### Examples
```
SOUR:POW:CORR:COLL:ITER:NTOL .005
source:power:correction:collect:iteration:ntolerance .1
```

## Query Syntax
```
SOURce:POWer:CORRection:COLLect:ITERation:NTOLerance?
```

## Return Type
Numeric

**Default** .05

---

**SOURce<cnum>:POWer<port>:CORRection:COLLect:METHOD <char> Superseded**

This command is replaced with SOUR:POW:CORR:COLLect[:ACQuire] which now specifies the method and the device.

(Read-Write) Selects the calibration method to be used for the source power cal acquisition.

## Parameters
- `<cnum>`: Any existing channel number. If unspecified, value is set to 1
- `<port>`: Port number to correct for source power. If unspecified, value is set to 1.
- `<char>`: Choose from:
  - **NONE** - No Cal method
  - **PMETer** - Power Meter is used for all readings. (same as "fast iteration" box not checked on dialog box)
  - **PMReceiver** - Power meter for the first iteration; then use the reference receiver for remaining readings if necessary (same as "fast iteration" box

---
Examples

SOUR:POW:COLL:METH PMET
source1:power2:correction:collect:method pmreceiver

Query Syntax
SOURce:POWer:CORRection:COLLect:METHod?

Return Type
Character

Default
NONE

SOURce<cnum>:POW<port>:CORRection:COLLect:SAVE [<RREC>]

(Write-only) Applies the array of correction values after a source power calibration sweep has completed. The source power correction will then be active on the specified source port for channel <cnum>. This command does NOT save the correction values. To save correction values, save an instrument / calibration state (*.cst file) after performing a source power cal.

Parameters

<cnum> Any existing channel number. If unspecified, value is set to 1
<port> Port number to correct for source power. If unspecified, value is set to 1.
<RREC> Optional argument.

RRECeiver In addition to a source Power Cal, perform a calibration of the reference receiver used in the measurement. ONLY the Reference Receiver calibration is then saved to a Cal Set or Cal Register as specified by the current setting of SENS:CORR:PREF:CSET:SAVE.

Examples
SOUR:POW:COLL:SAVE
source1:power2:correction:collect:save rreceiver

Query Syntax
Not Applicable

Default
Not Applicable

SOURce<cnum>:POW<pmChan>:SENsor:[FRANge]<num1>,<num2>
Specifies the frequency range over which the power sensors connected to the specified channels (A and B) of the power meter can be used (minimum frequency, maximum frequency). If the power meter has only a single channel, that channel is considered channel A.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pmChan>` Power Meter channel. Choose from:
  - A - Channel A
  - B - Channel B
- `<num1>` Minimum frequency for the sensor. If a frequency unit is not specified, Hz is assumed. No limits are placed on this value.
- `<num2>` Maximum frequency for the sensor. If a frequency unit is not specified, Hz is assumed. No limits are placed on this value.

**Examples**

```
SOUR:POW:COR:COLL:ASEN 100E3, 3E9
source1:power2:correction:collect:bsensor:frange 10 MHz, 18 GHz
```

**Query Syntax**

```
SOURce:POWer:CORRection:COLLect:ASENsor[:FRANge]?
SOURce:POWer:CORRection:COLLect:BSENsor[:FRANge]?
```

**Return Type**

Numeric

**Default**

0,0

---

Specifies the reference cal factor for the power sensor connected to channel A or B of the power meter. If the power meter has only a single channel, that channel is considered channel A.

**Note:** If the sensor connected to the specified channel of the power meter contains cal factors in EPROM (such as the Agilent E-series power sensors), those will be the cal factors used during the calibration sweep. The reference cal factor value associated with this command, and any cal factors entered into the PNA for that sensor channel, will not be used.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<pmChan>` Power Meter channel. Choose from:
  - A - Channel A
  - B - Channel B
Reference cal factor in percent. Choose any number between 1 and 150.

Examples
SOUR:POW:CORR:COLL:ASEN:RCF 98.7
source1:power2:correction:collect:bsensor:rcfactor 105

Query Syntax
SOURce:POWer:CORRection:COLLect:ASENsor:RCFactor?
SOURce:POWer:CORRection:COLLect:BSENsor:RCFactor?

Return Type
Numeric
Default 100

SOURce<cnum>:POWer:CORRection:COLLect:<pmChan>SEN:SELect
(Read-Write) Sets and returns the power sensor channel (A or B) to be used. This performs the same function as the Use this sensor only checkbox in the Power Sensor Settings dialog.

Note: This write portion of this command is only necessary when performing an SMC calibration.

Parameters
  <cnum> Any existing channel number. If unspecified, value is set to 1
  <pmChan> Power Meter channel. Choose from:
    A - Channel A
    B - Channel B

Examples
SOUR:POW:CORR:COLL:<pmChan>SEN:SEL 'Write
source1:power2:correction:collect:bsensor:select? 1e9 'Read

Query Syntax
SOURce:POWer:CORRection:COLLect:ASENsor:SELect? <Frequency>
SOURce:POWer:CORRection:COLLect:BSENsor:SELect? <Frequency>

Returns a boolean 1 or 0 (true or false) indicating whether the sensor is to be used at the specified frequency.

If frequency checking is OFF, then the <Frequency> parameter is ignored. The query returns if the sensor is selected for ALL frequencies.

Return Type
Numeric
Default Not Applicable

SOURce<cnum>:POWer:CORRection:COLLect:TABLe:DATA <data>
(Read-Write) Read or write data into the selected table. Use SOUR:POW:COR:COLL:TABLE:SELect to select a table.

- If the selected table is a power sensor table, the data is interpreted as cal factors in **percent**.
- If the loss table is selected, the data is interpreted as loss in dB.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<data>` Data to write into the selected table.

**Examples**

```
SOURce:POWer:CORRection:COLLect:TABLE:DATA 0.12, 0.34, 0.56
```

**Query Syntax**

```
SOURce<cnum>:POWer:CORRection:COLLect:TABLE:DATA?
```

If the selected table is currently empty, no data is returned.

**Return Type** Numeric - one number per table segment.

**Default** Not Applicable

---

**SOURce<cnum>:POWer:CORRection:COLLect:TABLE:FREQuency <data>**

(Read-Write) Read or write frequency values for the selected table (cal factor table for a power sensor, or the loss compensation table).

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<data>` Frequency data to write into the selected table.

**Examples**

```
SOURce:POWer:CORRection:COLLect:TABLE:FREQuency 10E6, 1.5E9, 9E9
```

**Query Syntax**

```
SOURce<cnum>:POWer:CORRection:COLLect:TABLE:FREQuency?
```

If the selected table is currently empty, no data is returned.

**Return Type** Numeric - one number per table segment

**Default** Not Applicable

---

**SOURce<cnum>:POWer:CORRection:COLLect:TABLE:LOSS[:STATe] <ON | OFF>**
(Read-Write) Indicates whether or not to adjust the power readings using the values in the loss table during a source power cal sweep.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<ON|OFF>` **ON (or 1)** - turns use of the loss table ON.
  **OFF (or 0)** - turns use of the loss table OFF.

**Examples**

```
SOUR:POW:CORR:COLL:TABL:LOSS ON
source1:power2:correction:collect:table:loss:state off
```

**Query Syntax**

```
SOURce:POWer:CORRection:COLLect:TABLe:LOSS[:STATe]?
```

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF (0)

---

**SOURce<cnum>:POWer:CORRection:COLLect:TABLe:POINts?**

(Read-only) Returns the number of segments that are currently in the selected table.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1

**Examples**

```
SOUR:POW:CORR:COLL:TABL:POIN?
source1:power2:correction:collect:table:points?
```

**Return Type**

Numeric

**Default**

0

---

**SOURce<cnum>:POWer:CORRection:COLLect:TABLe[:SELect] <char>**
(Read-Write) Selects which table you want to write to or read from. Read or write using SOURce:POWer:CORRection:COLLect:TABLE:FREQuency and SOURce:POWer:CORRection:COLLect:TABLE:DATA

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<char>` Choose from:

  **NONE** - No table selected
  **ASEnsor** - Cal Factor table for Power Sensor A
  **BSEnsor** - Cal Factor table for Power Sensor B
  **LOSS** - Loss compensation table

**Examples**

```
SOUR:POW:CORR:COLL:TABLE:ASEN
source1:power2:correction:collect:table:select bsensor
```

**Query Syntax**

SOURce:POWer:CORRection:COLLect:TABLE[:SELect]?

**Return Type**

Character

**Default**

NONE

---

**SOURce<cnum>:POW<port>:CORRection:DATA <data>[,src]**

(Read-Write) Writes and reads source power calibration data.

When querying source power calibration data, if no source power cal data exists for the specified channel and source port, then no data is returned.

If a change in the instrument state causes interpolation and/or extrapolation of the source power cal, the correction data associated with this command correspond to the new instrument state (interpolated and/or extrapolated data).

If the channel is sweeping the source backwards, then the first data point is the highest frequency value; the last data point is the lowest. Use the SENS:X:VALues? command to return the X-axis values in the displayed order.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.
- `<data>` Correction Data
**SOURce<cnum>:POWer<port>:CORRection:DATA**

(Read-Write) Specifies the power level that is expected at the desired reference plane (DUT input or output). This is not used for segment sweep with independent power levels or power sweeps.

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1
- `<port>` Port number to correct for source power. If unspecified, value is set to 1.
- `<num>` Cal power level in dBm. Because this could potentially be at the output of a device-under-test, no limits are placed on this value here. It is realistically limited by the specifications of the device (power sensor) that will be used for measuring the power. The power delivered to the PNA receiver must never exceed PNA specifications for the receiver!

**Note:** Although this command still works, it is recommended that you specify cal power by setting the test port power and offset value.

**Examples**

SOUR:POW:CORR:LEV 10
source1:power2:correction:level 0 dbm

**Query Syntax**

SOURce:POWer:CORRection:DATA? [src]
SOURce<cnum>:POWer<port>:CORRection:OFFSet[:MAGNitude] <num>[,src]

(Read-Write) Sets or returns a power level offset from the PNA test port power. This can be a gain or loss value (in dB) to account for components you connect between the source and the reference plane of your measurement. For example, specify 10 dB to account for a 10 dB amplifier at the input of your DUT.

Cal power is the sum of the test port power setting and this offset value. Following the calibration, the PNA power readouts are adjusted to the cal power.

Parameters

- **<cnum>**  Any existing channel number. If unspecified, value is set to 1
- **<port>**  Port number to correct for source power. If unspecified, value is set to 1.
- **<num>**  Gain or loss value in dB. Choose a value between -200 and 200

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

Examples

```
SOUR:POW:CORR:OFFS 10
source1:power2:correction:offset:magnitude -3
```

Query Syntax

SOURce:POWer:CORRection:OFFSet[:MAGNitude]? [src]

Return Type  Numeric

Default  0 dB

SOURce<cnum>:POWer<port>:CORRection[:STATe] <bool>[,src]
(Read-Write) Enables and disables source power correction for the specified port on the specified channel.

**Parameters**

- <cnum>: Any existing channel number. If unspecified, value is set to 1
- <port>: Port number to correct for source power. If unspecified, value is set to 1.
- <bool>: ON (or 1) turns source power correction ON.
  
  OFF (or 0) - turns source power correction OFF.

While this argument can be used to make settings for ALL ports, it is designed to access ports that are not simple numbers, such as "Port 1 Src2". Otherwise, the <port> argument performs the same function. If both arguments are specified, [src] takes priority.

**Examples**

```plaintext
SOUR:POW:CORR ON
source1:power2:correction:state off
```

**Query Syntax**

```
SOURce:POWer:CORRection[::STATe]? [src]
```

**Return Type**

Boolean (1 = ON, 0 = OFF)

**Default**

OFF (0)

---

Last modified:

- April 17, 2007 Removed ch and port arguments for 4 settling and accuracy commands.
- 9/12/06 MQ Modified for receiver only SPC
Status Register Commands

The status registers enable you to query the state of selected events that occur in the analyzer.

Note: This documentation requires familiarity with the "Standard Status Data Structure - Register Model" as defined in IEEE Std 488.2-1992.

Click on a blue keyword to view the command details.

See Also

- Example Programs
- Learn about Status Registers
- Synchronizing the PNA and Controller
- SCPI Command Tree

Note: Any bit not shown in the registers is not used but may be reserved for future use.
Status Byte Register

Summarizes the states of the other registers and monitors the PNA output queue. It also generates service requests. The Enable register is called the Service Request Enable Register.

<table>
<thead>
<tr>
<th>Commands</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*CLS</td>
<td>Clears ALL &quot;event&quot; registers and the SCPI Error / Event queue. The corresponding ENABLE registers are unaffected.</td>
</tr>
<tr>
<td>*STB?</td>
<td>Reads the value of the analyzer's status byte. The byte remains after being read.</td>
</tr>
<tr>
<td>*SRE?</td>
<td>Reads the current state of the Service Request Enable Register.</td>
</tr>
<tr>
<td>*SRE &lt;num&gt;</td>
<td>Sets bits in the Service Request Enable register. The current setting of the SRE register is stored in non-volatile memory. Use *SRE 0 to clear the enable.</td>
</tr>
</tbody>
</table>

<num> Combined value of the weights for bits to be set.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>Error / Event queue Summary</td>
<td>the Error / Event queue is not empty. To read the error message, use SYST:ERR?</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>Questionable Register Summary</td>
<td>any enabled bit in the questionable event status register is set to 1</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>Message Available</td>
<td>the output queue is not empty</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>Standard Event Register Summary</td>
<td>any enabled bit in the standard event status register is set to 1</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>Request Service</td>
<td>any of the other bits in the status byte register is set to 1 (used to alert the controller of a service request within the analyzer). This bit cannot be disabled.</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>Operation Register Summary</td>
<td>any enabled bit in the standard operation event status register is set to 1</td>
</tr>
</tbody>
</table>
STATus:QUESTionable:<keyword>

Summarizes conditions that monitor the quality of measurement data.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONDition?</td>
<td>STAT:QUES:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:QUES:ENAB 1024</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:QUES?</td>
</tr>
<tr>
<td>:NTRansition</td>
<td>STAT:QUES:NTR 1024</td>
</tr>
<tr>
<td>:PTRansition</td>
<td>STAT:QUES:PTR 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>512</td>
<td>Integrity Reg summary</td>
<td>any enabled bit in the Integrity event register is set to 1</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>Limit Registers summary</td>
<td>any enabled bit in the Limit event registers is set to 1</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>Define Registers summary</td>
<td>any enabled bit in the Define event registers is set to 1</td>
</tr>
</tbody>
</table>

STATus:QUESTionable:INTegrity <keyword>

Summarizes conditions in the Measurement Integrity register.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONDition?</td>
<td>STAT:QUES:INT:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:QUES:INT:ENAB 1024</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:QUES:INT?</td>
</tr>
<tr>
<td>:NTRansition</td>
<td>STAT:QUES:INT:NTR 1024</td>
</tr>
<tr>
<td>:PTRansition</td>
<td>STAT:QUES:INT:PTR 0</td>
</tr>
</tbody>
</table>

2312
<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>Measurement Summary</td>
<td>any bit in the <strong>Measurement Integrity</strong> event register is set to 1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>Hardware Summary</td>
<td>any bit in the <strong>Hardware</strong> event register is set to 1</td>
</tr>
</tbody>
</table>

**STATus:QUEStionable:INTegrity:HARDware<keyword>**

Monitors the status of hardware failures.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>[:EVENt]?</td>
<td>STAT:QUES:INT:HARD?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:QUES:INT:HARD:NTR 1024</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
<td>STAT:QUES:INT:HARD:PTR 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>Phase Unlock</td>
<td>the source has lost phaselock, possibly caused by a reference channel open or a hardware failure.</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>Unleveled</td>
<td>the source power is unleveled. This could be caused by a source set for more power than it can deliver at the tuned frequency. Or it could be caused by a hardware failure.</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>Not used</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>EE Write Failed</td>
<td>an attempted write to the EEPROM has failed, possibly caused by a hardware failure.</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>Not used</td>
<td>N/A</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>Ramp Cal Failed</td>
<td>the analyzer was unable to calibrate the analog ramp generator due to a possible hardware failure.</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>Not used</td>
<td>N/A</td>
</tr>
</tbody>
</table>
STATus:QUESTionable:INTegrity:MEASurement<n> <keyword>

Monitors the lag between changing a channel setting and when the data is ready to query.

When you change the channel state (start/stop freq, bandwidth, and so forth), then the questionable bit for that channel is set. This indicates that your desired channel state does not yet match the data you would get if querying a data trace. When the next sweep is complete (without aborting in the middle), and the data trace matches the channel state that produced it, the bit is cleared for that channel.

<n> Measurement register number. Choose from 1 to 3

Example

:CONDition?
STAT:QUES:INT:MEAS1:COND?

:ENABle <bits>
STAT:QUES:INT:MEAS2:ENAB 1024

[:EVENt]?
STAT:QUES:INT:MEAS3?

:NTRansition <bits>
STAT:QUES:INT:MEAS2:NTR 1024

:PTRansition <bits>
STAT:QUES:INT:MEAS1:PTR 0

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td>Summary from Meas Reg 3 a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>15</td>
<td>29</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
<td>16</td>
<td>30</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>4</td>
<td>17</td>
<td>31</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>5</td>
<td>18</td>
<td>32</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>6</td>
<td>19</td>
<td></td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>7</td>
<td>20</td>
<td></td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>8</td>
<td>21</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>-----</td>
<td>---</td>
<td>----</td>
<td>-----------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>9</td>
<td>22</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>10</td>
<td>23</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>11</td>
<td>24</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>12</td>
<td>25</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>13</td>
<td>26</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>14</td>
<td>27</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>Summary from Meas Reg 2</td>
<td>28</td>
<td>a setting change on this channel has occurred and the data does not yet reflect that change.</td>
<td></td>
</tr>
</tbody>
</table>

**STATus:QUEStionable:LIMit<n> <keyword>**

Monitors and summarizes the status of limit line failures. When a trace fails, the representative bit is set to 1.

Bit 0 is used to summarize failures in the registers that follow. For example, Limit Register 3, bit 0, summarizes the failures from registers 4 through 10.

All enable bits are set to 1 by default.

To find the measurement number, use **Calc:Par:Mnum**

<n> Limit register: Choose from 1 to 10.

<keyword> Example

:CONDition? STAT:QUES:LI1M4:COND?

:ENABle <bits> STAT:QUES:LI1M1:ENAB 1024

[:EVENt]? STAT:QUES:LI1M3?

:NTRansition <bits> STAT:QUES:LI1M2:NTR 1024

:NTRansition? STAT:QUES:LI1M1:NTR?
### Limit Register <n>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>...</th>
<th>41</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2-42</td>
<td>3-42</td>
<td>4-42</td>
<td>5-42</td>
<td>6-42</td>
<td>7-42</td>
<td>8-42</td>
<td>...</td>
<td>42</td>
<td></td>
</tr>
</tbody>
</table>

**Bit is set to 1 when the following conditions exist:**

Summary Bit - If any bit from that register fails, it propagates to the previous register, bit 0.

### Trace Numbers

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>...</th>
<th>575</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>2</th>
<th>4</th>
<th>2</th>
<th>16</th>
<th>30</th>
<th>44</th>
<th>58</th>
<th>72</th>
<th>86</th>
<th>100</th>
<th>...</th>
<th>576</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>3</th>
<th>8</th>
<th>3</th>
<th>17</th>
<th>31</th>
<th>45</th>
<th>59</th>
<th>73</th>
<th>87</th>
<th>101</th>
<th>...</th>
<th>577</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>4</th>
<th>16</th>
<th>4</th>
<th>18</th>
<th>32</th>
<th>46</th>
<th>60</th>
<th>74</th>
<th>88</th>
<th>102</th>
<th>...</th>
<th>578</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>5</th>
<th>32</th>
<th>5</th>
<th>19</th>
<th>33</th>
<th>47</th>
<th>61</th>
<th>75</th>
<th>89</th>
<th>103</th>
<th>...</th>
<th>579</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>6</th>
<th>64</th>
<th>6</th>
<th>20</th>
<th>34</th>
<th>48</th>
<th>62</th>
<th>76</th>
<th>90</th>
<th>104</th>
<th>...</th>
<th>580</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>7</th>
<th>128</th>
<th>7</th>
<th>21</th>
<th>35</th>
<th>49</th>
<th>63</th>
<th>77</th>
<th>91</th>
<th>105</th>
<th>...</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>8</th>
<th>256</th>
<th>8</th>
<th>22</th>
<th>36</th>
<th>50</th>
<th>64</th>
<th>78</th>
<th>92</th>
<th>106</th>
<th>...</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>9</th>
<th>512</th>
<th>9</th>
<th>23</th>
<th>37</th>
<th>51</th>
<th>65</th>
<th>79</th>
<th>93</th>
<th>107</th>
<th>...</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>10</th>
<th>1024</th>
<th>10</th>
<th>24</th>
<th>38</th>
<th>52</th>
<th>66</th>
<th>80</th>
<th>94</th>
<th>108</th>
<th>...</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th>11</th>
<th>2048</th>
<th>11</th>
<th>25</th>
<th>39</th>
<th>53</th>
<th>67</th>
<th>81</th>
<th>95</th>
<th>109</th>
<th>...</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any point on trace fails the limit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
To determine Register, Bit number, and Weight for trace numbers between 113 and 574 (not shown in the above table) use the following calculations.

The limit status for trace numbers higher than 580 can NOT be tracked.

The following example calculates the Register, Bit number, and Bit Weight for trace # 400:

- To determine Limit **Register** number, use \(((\text{Trace } \# - 1) / 14) + 1\).
- To determine Limit **Bit Number**, use the remainder +1 of the above calculation.
- \(((400-1)/14) + 1 = \text{Register} # r+1\text{Bit} \)
  - \(399/14 = 28 \, r7\)
  - \(28+1 = \text{Register} 29\)
  - \(7+1 = \text{Bit number} 8\)
- To determine Limit **Bit Weight**: Use above table. For example: Bit 8 = 256

**STATus:**QUESTIONable:DEFINE<keyword>

Summarizes conditions in the Questionable:Define:User<1|2|3> event registers.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONDition?</td>
<td>STAT:QUES:DEF:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:QUES:DEF:ENAB 1024</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:QUES:DEF?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:QUES:DEF:NTR 1024</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
<td>STAT:QUES:DEF:PTR 0</td>
</tr>
</tbody>
</table>

2317
Bit   Weight | Description | Bit is set to 1 when the following conditions exist:
--- | --- | ---
1 | 2 | USER1 | any bit in the USER1 event register is set to 1
2 | 4 | USER2 | any bit in the USER2 event register is set to 1
3 | 8 | USER3 | any bit in the USER3 event register is set to 1

**STATus:QUESTionable:DEFine:USER<1|2|3><keyword>**

Monitors conditions that you define and map in any of the three QUES:DEF:USER event registers.

- **Example**
  - **:ENABle <bits>**
    - STAT:QUES:DEF:USER1:ENABle 1024
  - **[:EVENT]?**
    - STAT:QUES:DEF:USER1?
  - **:MAP <bit>,<error>**
    - STAT:QUES:DEF:USER1:MAP 0,-113 'when error -113 occurs, bit 0 in USER1 will set to 1.'
Standard Event Status Register

Monitors "standard" events that occur in the analyzer. This register can only be cleared by:

- a Clear Command (*CLS).
- reading the Standard Enable Status Register (*ESE?).
- a power-on transition. The analyzer clears the register and then records any transitions that occur, including setting the Power On bit (7).

<table>
<thead>
<tr>
<th>Commands</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*ESE?</td>
<td>Reads the settings of the standard event ENABLE register.</td>
</tr>
<tr>
<td>*ESE &lt;bits&gt;</td>
<td>Sets bits in the standard event ENABLE register. The current setting is saved in non-volatile memory. &lt;bits&gt; The sum of weighted bits in the register. Use *ESE 0 to clear the enable register.</td>
</tr>
<tr>
<td>*ESR?</td>
<td>Reads and clears the EVENT settings in the Standard Event Status register.</td>
</tr>
<tr>
<td>*OPC</td>
<td>Sets bit 0 when the overlapped command is complete. (see Understanding Command Synchronization / OPC).</td>
</tr>
<tr>
<td>*OPC?</td>
<td>Operation complete query - read the Operation Complete bit (0).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
</table>
| 0   | 1      | Operation Complete | the two following events occur in order:  
|     |        |             | 1. the *OPC command is sent to the analyzer  
|     |        |             | 2. the analyzer completes all pending overlapped commands |
| 1   | NA     | Request Control | Not Supported - the analyzer application is not configured to control GPIB operation |
| 2   | 4      | Query Error   | a query error is detected indicating:  
|     |        |             | - an attempt to read data from the output queue when no data was present OR  
|     |        |             | - data in the output queue was lost, as in an overflow |
| 4   | 16     | Execution Error | an execution error is detected indicating:  
|     |        |             | - a <PROGRAM DATA> element was outside the legal range or inconsistent with the operation of the analyzer OR |
- the analyzer could not execute a valid command due to some internal condition

| Command Error | a command error is detected indicating that the analyzer received a command that: |
| 5 32 | - did not follow proper syntax  
| 7 128 | - was misspelled  
|  | - was an optional command it does not implement |

Power to the analyzer has been turned OFF and then ON since the last time this register was read.

**STATus:OPERation<keyword>**

Summarizes conditions in the Averaging and Operation::Define:User<1|2|3> event registers.

- **:CONDition?**
  - **STAT:OPER:COND?**
- **:ENABle <bits>**
  - **STAT:OPER:ENAB 1024**
  - **STAT:OPER?**
- **[:EVENt]?**
  - **STAT:OPER:NTR 1024**
- **:NTRansition <bits>**
  - **STAT:OPER:PTR 0**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>256</td>
<td>Averaging summary</td>
<td>either enabled bit in the <strong>Averaging summary</strong> event register is set to 1</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>User Defined summary</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>Device summary</td>
<td>either enabled bit in the <strong>Device summary</strong> event register is set to 1</td>
</tr>
</tbody>
</table>
**STATus:OPERation:AVERaging<n> <keyword>**

Monitors and summarizes the status of Averaging on traces 1 to 128. When averaging for a trace is complete, the representative bit is set to 1.

Bit 0 is used to summarize the status in the registers that follow. For example, Average Register, bit 0, summarizes the status from registers 4 through 10.

All enable bits are set to 1 by default.

To find the measurement number, use `Calc:Par:Mnum`.

<table>
<thead>
<tr>
<th>&lt;n&gt; Averaging Register. Choose from 1 to 10</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;keyword&gt; Example</td>
<td></td>
</tr>
<tr>
<td>:CONDition? STAT:OPER:AVER1:COND?</td>
<td></td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt; STAT:OPER:AVER1:ENAB 1024</td>
<td></td>
</tr>
<tr>
<td>[:EVENt]? STAT:OPER:AVER1?</td>
<td></td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt; STAT:OPER:AVER1:NTR 1024</td>
<td></td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt; STAT:OPER:AVER1:PTR 0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Averaging Register &lt;n&gt;</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Weight</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
| Summary Bit - If any bit from that register fails, it propagates to the previous register, bit 0.

<table>
<thead>
<tr>
<th>Trace Numbers</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
</tr>
</tbody>
</table>
To determine Register, Bit number, and Weight for trace numbers between 113 and 574 (not shown in the above table) use the following calculations.

The averaging status for trace numbers higher than 580 can NOT be tracked.

The following example calculates the Register, Bit number, and Bit Weight for trace # 400:

- To determine **Register** number, use \(((\text{Trace } \# - 1) / 14) + 1\).
- To determine **Bit Number**, use the **remainder** +1 of the above calculation.
- \(((400-1)/14) + 1 = \text{Register}\# \text{ r+1 Bit}\)
  - \(399/14 = 28 \text{ r7}\)
  - \(28+1= \text{Register 29}\)
  - \(7+1= \text{Bit number 8}\)

<table>
<thead>
<tr>
<th>Trace #</th>
<th>Register</th>
<th>Bit Number</th>
<th>Bit Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>32</td>
<td>5</td>
<td>33</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>6</td>
<td>34</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>7</td>
<td>35</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>8</td>
<td>36</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>9</td>
<td>37</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>10</td>
<td>38</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>11</td>
<td>39</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>12</td>
<td>40</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>13</td>
<td>41</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>14</td>
<td>42</td>
</tr>
</tbody>
</table>

Averaging on this trace is complete.
To determine **Bit Weight**: Use above table. For example: Bit 8 = 256

### STATus:OPERation:DEFine<keyword>
Summarizes conditions in the OPERation:Define:User<1|2|3> event registers.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONDition?</td>
<td>STAT:OPER:DEF:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:OPER:DEF:ENAB &lt;bits&gt;</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:OPER:DEF?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:OPER:DEF:NTR &lt;bits&gt;</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
<td>STAT:OPER:DEF:PTR &lt;bits&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Weight</th>
<th>Description</th>
<th>Bit is set to 1 when the following conditions exist:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>USER1</td>
<td>any bit in the <strong>USER1</strong> event register is set to 1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>USER2</td>
<td>any bit in the <strong>USER2</strong> event register is set to 1</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>USER3</td>
<td>any bit in the <strong>USER3</strong> event register is set to 1</td>
</tr>
</tbody>
</table>

### STATus:OPERation:DEFine:USER<1|2|3><keyword>
Monitors conditions that you define and map in any of the three OPER:DEF:USER event registers.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:OPER:DEF:USER1:ENAB &lt;bits&gt;</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:OPER:DEF:USER1?</td>
</tr>
<tr>
<td>:MAP &lt;bit&gt;,&lt;error&gt;</td>
<td>STAT:OPER:DEF:USER1:MAP &lt;bit&gt;,&lt;error&gt;</td>
</tr>
<tr>
<td>Bit</td>
<td>Weight</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
</tr>
</tbody>
</table>

**STATus:OPERation:DEVice<keyword>**

Summarizes conditions in the OPERation:DEVice event registers.

<table>
<thead>
<tr>
<th>&lt;keyword&gt;</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONDition?</td>
<td>STAT:OPER:DEV:COND?</td>
</tr>
<tr>
<td>:ENABle &lt;bits&gt;</td>
<td>STAT:OPER:DEV:ENAB 16</td>
</tr>
<tr>
<td>[:EVENt]?</td>
<td>STAT:OPER:DEV?</td>
</tr>
<tr>
<td>:NTRansition &lt;bits&gt;</td>
<td>STAT:OPER:DEV:NTR 16</td>
</tr>
<tr>
<td>:PTRansition &lt;bits&gt;</td>
<td>STAT:OPER:DEV:PTR 0</td>
</tr>
<tr>
<td>Bit</td>
<td>Weight</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
</tr>
</tbody>
</table>

Last modified:

March 10, 2008   Fixed typo in STAT:QUES:INT register
9/19/06   MQ Modified for unlimited windows.
System Commands

SYSTem:
  ACTive
      | CHANnel
      | MEASurement
CHANnels
  | HOLD
  | RESume
COMMunicate
  | GPIB
      | PMETER
          | ADDRess
      | RDEVice
          | CLOSE
          | OPEN
          | READ?
          | RESet
          | WBNary
          | WBLock
          | WRITE
      | PSENsensor
      | USB:PMETER:CAT?

CONFigure
CORRection
  | WIZard
ERRor?
  | COUNt?
  | REPort
      | SUNLeveled
FPReset
MACRO:COPY
  | CHANnel
      | SOURce
PRESet
PREFerences:ITEM
  | OFFSet
      | RCV
      | SRC
      | PSRTrace
      | RETRace
Click on a blue keyword to view the command details.

See Also

- Example Programs
- Synchronizing the PNA and Controller
- SCPI Command Tree

**SYSTem:ACTive:CHANnel?**

*(Read-only)* Returns the number of the active channel or an error message if there is no active channel. The active channel is the channel number that contains the active measurement.

**Examples**

- SYST:PRES
- SYST:ACT:CHAN?

'Returns 1

**Return Type** Integer

**Default** Not Applicable

**SYSTem:ACTive:MEASurement?**
(Read-only) Returns the name of the active measurement or an error message if there is no active measurement. While looking at the PNA display, the active measurement is the trace that has an indented Trace Status button and a label in the upper-left corner of the display. Only displayed measurements can be active.

| Examples      | SYST:PRES  
|              | SYST:ACT:MEAS?  
|              | 'Returns "CH1_S11_1"  

Return Type String

Default Not Applicable

**SYSTem:CHANnels:HOLD**

(Write-only) Places all channels in hold mode. To place a single channel in hold mode, use SENS:SWE:MODE.

| Examples      | SYST:CHAN:HOLD  

Query Syntax Not Applicable

Default Not Applicable

**SYSTem:CHANnels:RESume**

(Write-only) Resumes the trigger mode of all channels that was in effect before sending SYSTem:CHANnels:HOLD (must be sent before SYST:CHAN:RESume).

| Examples      | SYST:CHAN:RES  

Query Syntax Not Applicable

Default Not Applicable

**SYSTem:COMMunicate:GPIB:PMETer:ADDRess <num>** Superseded
**Note:** This command is replaced with `SYST:COMM:PSENsor`

(Read-Write) Specifies the GPIB address of the power meter to be used in a source power calibration. When performing a source power cal, the PNA will search VISA interfaces that are configured in the Agilent IO Libraries on the PNA.

Use

**Parameters**

- `<num>`  
  GPIB address of the power meter. Choose any integer between 0 and 30.

**Examples**

```
SYST:COMM:GPIB:PMET:ADDR 13
```

```
system:communicate:gpib:pmeter:address 14
```

**Query Syntax**

`SYSTem:COMMunicate:GPIB:PMETer:ADDress?`

**Return Type**

Numeric

**Default**

13

---

`SYSTem:COMMunicate:GPIB:RDEVice:CLOSE <ID>`

(Write only) Closes the remote GPIB session. This command should be sent when ending every successful OPEN session.

**Parameters**

- `<ID>`  
  Session identification number that was returned with the `OPEN?` command.

**Examples**

See an example program

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

`SYSTem:COMMunicate:GPIB:RDEVice:OPEN <bus>, <addr>, <timeout>`
(Read-Write) Initiates a GPIB pass-through session. First send this OPEN command, then send the OPEN query to read the session ID number. An existing GPIB pass-through session remains open after an instrument preset.

To learn more about GPIB pass-through capability, see the example program.

**Parameters**

- `<bus>` Bus ID number.
  You can find the USB-GPIB adapter bus number by looking at the dialog that appears when the USB-GPIB device is connected. Error 1073 indicates the bus or address number is incorrect.
  Use 0 (zero) when connected using a GPIB cable to the PNA controller port.

- `<addr>` GPIB Address of the device to be controlled

- `<timeout>` The amount of time (in milliseconds) to wait for a response from the remote device after sending a command. A "timeout" error is displayed after this time has passed without a response.

**Examples** See an example program

**Query Syntax** SYSTem:COMMunicate:GPIB:RDEVice:OPEN?

Returns the session identification number that is used when communicating with this device.

**Return Type** Numeric

**Default** Not Applicable

---

**SYSTem:COMMunicate:GPIB:RDEVice:READ? <ID>**

(Read-only) Returns data from the GPIB pass-through device.

**Parameters**

- `<ID>` Session identification number that was returned with the OPEN? command.

**Examples** See an example program

**Return Type** String

**Default** Not Applicable

---

**SYSTem:COMMunicate:GPIB:RDEVice:RESet**
(Write-only) Performs the same function as `SYST:COMM:GPIB:RDEV:CLOS` except that ALL pass-through sessions are closed.

**Examples**

```
SYST:COMM:GPIB:RDEV:RES
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SYStem:COMMunicate:GPIB:RDEVice:WBINary <ID>,<data>**

(Write-only) Sends data to a GPIB pass-through device. This command requires a header that specifies the size of the data to be written. The header (described below) is not passed along to the device.

Use this command if too many embedded quotes prevent you from using `SYST:COMM:GPIB:RDEV:WRIT`. Use `SYST:COMM:GPIB:RDEV:OPEN` to open the pass through session.

**Parameters**

- `<ID>` Session identification number that was returned with the `OPEN?` command.
- `<data>` Data to be sent to the GPIB pass-through device. Use the following syntax:

```
#<num_digits><byte_count><data_bytes><NL><END>
```

- `<num_digits>` specifies how many digits are contained in `<byte_count>`
- `<byte_count>` specifies how many data bytes will follow in `<data_bytes>`

**Examples**

```
SYStem:COMMunicate:GPIB:RDEVice:WBINary 101,#17ABC+XYZ<nl><end>
```

# - always sent before data.

1 - specifies that the byte count is one digit (7).

7 - specifies the number of data bytes that will follow, not counting `<NL><END>`.

ABC+XYZ - Data block

`<nl><end>` - always sent at the end of block data.

The following example sends a line feed at the end.
SYSTem:COMMunicate:GPIB:RDEVice:WBLock <ID>,<data>

(Write-only) Same as SYSTem:COMM:GPIB:RDEV:WBIN (above) but the header IS passed along to the device.

Use this command if too many embedded quotes prevent you from using SYST:COMM:GPIB:RDEV:WRIT.

Parameters

- **<ID>**  Session identification number that was returned with the OPEN? command.
- **<data>** Data to be sent to the GPIB pass-through device. See previous command.

Examples

See previous example.

Query Syntax Not Applicable

**Default** Not Applicable

SYSTem:COMMunicate:GPIB:RDEVice:WRITe <ID>,<string>

(Write-only) Sends ASCII string data to the GPIB pass-through device.

A line feed is NOT appended to the string data. To send a line feed, see the example in SYST:COMM:GPIB:RDEV:WBIN.

Parameters

- **<ID>**  Session identification number that was returned with the OPEN? command.
- **<string>** Commands to be sent to the GPIB pass-through device.

Examples

See an example program

Query Syntax Not Applicable

**Default** Not Applicable

SYSTem:COMMunicate:PSENsor <char>, <string>
This command replaces \texttt{SYST:COMM:GPIB:PMET:ADDR}.

\textbf{(Read-Write)} Specifies the type and location of the power meter to be used in a source power calibration.

\begin{itemize}
  \item \textbf{Parameters}
  \begin{itemize}
    \item \texttt{char} Type of power meter/sensor. Choose from:
      \begin{itemize}
        \item \texttt{GPIB} GPIB power meter
        \item \texttt{USB} USB power sensor or USB power sensor
        \item \texttt{LAN} LAN enabled power meter
      \end{itemize}
    \item \texttt{string} For \texttt{GPIB}, address of the power meter. Choose any integer between 0 and 30.
    For \texttt{USB}, the ID string of the power meter or power sensor. Use \texttt{SYST:COMM:USB:PMET:CAT?} to see a list of ID strings of connected power meters and sensors.
    For \texttt{LAN}, the hostname or IP address of the power meter.
  \end{itemize}

  \textbf{Examples}
  \begin{itemize}
    \item \texttt{SYST:COMM:PSEN gpib, "14"}
    \item \texttt{system:communicate:psensor usb, "Agilent Technologies,U2000A,MY12345678,X.01.16"}
    \item \texttt{syst:comm:psen lan, "mymeter.agilent.com"}
  \end{itemize}

  \textbf{Query Syntax} \texttt{SYSTem:COMMunicate:PSENsor?}

  \textbf{Return Type} Character/\texttt{String} \textbf{Default} GPIB
\end{itemize}

\textbf{SYSTem:COMMunicate:USB:PMETer:CATalog?}

\textbf{(Read-only)} Returns the ID string of power meters/sensors that are connected to the PNA USB. Use the list to select a power sensor for a source power cal.

\begin{itemize}
  \item \textbf{Parameters}
  \item \textbf{Examples}
  \begin{itemize}
    \item \texttt{SYST:COMM:USB:PMET:CAT?}
    \item \texttt{system:communicate:usb:pmeter:catalog?}
  \end{itemize}

  \textbf{Return Type} Comma-delimited strings \textbf{Default} Not applicable
**SYSTem:CONFigure <model>,<address>**

(Write-only) Restarts as an "N-port" PNA using the specified multiport test set.

**Learn more about PNA Multiport capability.**

**See other commands to configure multiport test sets.**

**Parameters**

- `<model>` String - Model of the test set with which to restart.
  
  Use "Native" to restart without a test set.

  To see a list of supported test sets, use **SENS:MULT:CAT?**

- `<address>` Numeric - GPIB Address of the test set. Ignored when model = "Native".

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:CONF &quot;NATIVE&quot;,0</td>
<td>System:configure &quot;N44xx&quot;,18</td>
</tr>
</tbody>
</table>

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SYSTem:CORRection:WIZard <char>**

(Write-only) Launches either the Calibration Wizard or the Version 2 Calibration Kit File Manager dialog box.

**Parameters**

- `<char>` Choose from:

  **MAIN** - Launches the Calibration Wizard

  **CKIT** - Launches the Version 2 Calibration Kit File Manager dialog box.

  Both display on the PNA screen.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:CORR:WIZ MAIN</td>
<td>System:correction:wizard ckit</td>
</tr>
</tbody>
</table>

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SYSTem:ERRor?**
(Read-only) Returns the next error in the error queue. Each time the analyzer detects an error, it places a message in the error queue. When the SYSTEM:ERROR? query is sent, one message is moved from the error queue to the output queue so it can be read by the controller. Error messages are delivered to the output queue in the order they were received. The error queue is cleared when any of the following conditions occur:

- When the analyzer is switched ON.
- When the *CLS command is sent to the analyzer.
- When all of the errors are read.

If the error queue overflows, the last error is replaced with a "Queue Overflow" error. The oldest errors remain in the queue and the most recent error is discarded.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:ERR?</td>
<td>System:Error?</td>
</tr>
<tr>
<td>system:error?</td>
<td></td>
</tr>
</tbody>
</table>

**Default** Not Applicable

**SYSTem:ERRor:COUNt?**

(Read-only) Returns the number of errors in the error queue. Use SYST:ERR? to read an error.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:ERR:COUN?</td>
<td>System:Error:Count?</td>
</tr>
</tbody>
</table>

**Default** Not Applicable

**SYSTem:ERRor:REPort:SUNLeveled <bool>**

(Read-Write) Specifies whether or not to report Source Unleveled errors to the SCPI system error buffer.

**Parameters**

- **<bool>**
  - **True** (or 1) Report Source Unleveled Errors. Read errors from the system error buffer using SYST:ERR?
  - **False** (or 0) Do NOT report Source Unleveled Errors.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>

**Query Syntax** SYSTem:ERRor:REPort:UNLeveled?

**Default** False
**SYSTem:FPReset**

*(Write-only)* Performs a standard Preset, then deletes the default trace, measurement, and window. The PNA screen becomes blank.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:FP</td>
</tr>
<tr>
<td>system:fpreset</td>
</tr>
</tbody>
</table>

**Default** Not applicable

**SYSTem:MACRo:COPY:CHANnel<cnum>[:TO] <num>**

*(Write-only)* Sets up channel `<num>` as a copy of channel `<cnum>`. Learn more about [copy channels](#).

**Parameters**

- `<cnum>` Any existing channel number. If unspecified, value is set to 1.
- `<num>` Number of the channel which is to become a copy of channel `<cnum>`.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:MACR:COPY:CHAN1 2</td>
</tr>
<tr>
<td>system:macro:copy:channel2:to 3</td>
</tr>
</tbody>
</table>

**Query Syntax** Not applicable

**Default** Not applicable

**SYSTem:MACRo:COPY:CHANnel<fromChan>:SOURce <fromPort>,<toChan>,<toPort>**

*(Write-only)* Copies and applies an existing Source Power Calibration to another channel. Learn more about [source power calibration](#).

**Parameters**

- `<fromChan>` Channel number of the existing source power correction.
- `<fromPort>` Port number of the existing source power correction.
- `<toChan>` Channel number to which the source power correction will be copied.
- `<toPort>` Port number to which the source power correction will be applied.

**Examples**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:MACR:COPY:CHAN1:SOUR 1,2,1</td>
</tr>
<tr>
<td>system:macro:copy:channel2:sourse 2,1,2</td>
</tr>
</tbody>
</table>

**Query Syntax** Not applicable

**Default** Not applicable
SYSTem:PRESet
(Write-only) Deletes all traces, measurements, and windows. In addition, resets the analyzer to factory
defined default settings and creates a S11 measurement named "CH1_S11_1". For a list of default
settings, see Preset.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always
preset the PNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

If the PNA display is disabled with DISP:ENAB OFF then SYST:PRES will NOT enable the display.

This command performs the same function as *RST.

Examples

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:PRES</td>
<td>system:preset</td>
</tr>
</tbody>
</table>

Default Not applicable

SYSTem:PREFerences:ITEM:OFFSet:RCV <bool>
(Read-Write) Set and return whether to offset the test port receivers by the amount of receiver
attenuation. Learn more.

To send this command using the PNA front panel, open the GPIB Command Processor Console, then
type either of the following examples at the command prompt. Then type the Query Syntax and press
enter to be sure the PNA took the command.

This setting remains until changed again using this command, or until the hard drive is changed or
reformatted.

Parameters

<bool> Choose from:

- **True (1)** Offset the test port receivers
- **False (0)** Do NOT offset the test port receivers

Examples

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>

Query Syntax

SYSTem:PREFerences:ITEM:OFFSet:RCV?

Return Type

Boolean

Default

PNA-L and E836xB: **False** (does NOT offset the display).

PNA-X: **True** (offsets the display).
**SYSTem:PREFerences:ITEM:OFFSet:SRC <bool>**

*(Read-Write)* Set and return whether to offset the reference receiver by the amount of source attenuation. [Learn more.](#)

To send this command using the PNA front panel, open the [GPIB Command Processor Console](#), then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

**Parameters**

<bool> Choose from:

- **True (1)** Offset the reference receivers.
- **False (0)** Do NOT Offset the reference receivers.

**Examples**

<table>
<thead>
<tr>
<th>Command String</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYST:PREF:ITEM:OFFS:SRC 1</td>
</tr>
<tr>
<td>system:preferences:item:offset:src false</td>
</tr>
</tbody>
</table>

**Query Syntax**

SYSTem:PREFerences:ITEM:OFFSet:SRC?

**Return Type**

Boolean

**Default**

All models: **True** (offset the display).

---

**SYSTem:PREFerences:ITEM:PSRTrace <char>**

*(Read-Write)* At the end of a power sweep, while waiting to trigger the next sweep, maintain source power at either the start power level or at the stop power level.

To send this command using the PNA front panel, open the [GPIB Command Processor Console](#), then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

**Parameters**

<char> Choose from:

- **START** - Maintain source power at the start power level.
- **STOP** - Maintain source power at the stop power level.
SYST:PREF:ITEM:PSRT STOP
system:preferences:item:psrtrace start

Query Syntax
SYSTem:PREFerences:ITEM:PSRTrace?

Return Type
Character

Default
START

SYSTem:PREFerences:ITEM:RETRace:POWer <char>

(Read-Write) For single-band frequency or segment sweeps ONLY, specify whether to turn RF power ON or OFF during a retrace. Learn more about RF power during sweep retrace.

To send this command using the PNA front panel, open the GPIB Command Processor Console, then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

This setting remains until changed using this command, or until the hard drive is changed or reformatted.

Parameters

<char> Choose from:

AUTO: Power is left ON during retrace of single-band frequency or segment sweeps ONLY.

OFF: Power is turned OFF during retrace of single-band frequency or segment sweeps ONLY.

Examples
SYST:PREF:ITEM:RETR:POW OFF
system:preferences:item:retrace:power auto

Query Syntax
SYSTem:PREFerences:ITEM:RETRace:POWer?

Return Type
Character

Default
AUTO

SYSTem:PREFerences:ITEM:SWITch:DEF <string>, <int>
(Read-Write) Sets the default setting for the Noise Tuner switch. This is the setting that occurs when a new channel is created. Learn more.

To send this command using the PNA front panel, open the GPIB Command Processor Console, then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

This setting remains until changed using this command, or until the hard drive is changed or reformatted.

**Parameters**

- **<string>** Name of the switch to set. Choose from:
  - "Port1NoiseTuner"

- **<int>** Value to set. Choose from:
  - 0 Sets the default (preset) to INTERNAL
  - 1 Sets the default (preset) to EXTERNAL

**Examples**

```plaintext
SYST:PREF:ITEM:SWIT:DEF "Port1NoiseTuner" 1 'Write
```

```plaintext
system:preferences:item:switch:DEF? "Port1NoiseTuner" 'Read
```

**Query Syntax**

SYSTem:PREFerences:ITEM:SWITch:DEF? <switch>

**Return Type**

Integer

**Default**

1 (External)

**SYSTem:SECurity[:LEVel] <char>**
**SYSTem:SECurity**

**Parameters**

<char> Choose from:

- **NONE** - ALL frequency information is displayed.
- **LOW** - NO frequency information is displayed. Frequency information can be redisplayed using the Security Setting dialog box or this command.
- **HIGH** - LOW setting plus GPIB console is disabled. Frequency information can be redisplayed ONLY by performing a Preset, recalling an instrument state with None or Low security settings, or using this command.
- **EXTRA** - HIGH setting plus ASCII data saving is disabled. Same method to redisplay frequency information as HIGH setting.

**Examples**

SYST:SEC LOW
system:security:level high

**Query Syntax**

SYSTem:SECurity[:LEVel]?  
**Return Type** Character  
**Default** None

---

**SYSTem:UPReset**

(Write-only) Performs a User Preset. There must be an active User Preset state file (see **Load** and **Save**) or an error will be returned. Learn more about User Preset.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the PNA to the factory preset settings, and **SYST:UPReset** will always perform a User Preset.

**Examples**

SYST:UPReset
system:upreset

**Query Syntax** Not Applicable  
**Default** Not Applicable

---

**SYSTem:UPReset:FPANel[:STATe]** <bool>
‘Checks' and 'clears' the enable box on the User Preset dialog box. This only affects subsequent Presets from the front panel user interface.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the PNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;bool&gt;</td>
<td>Front Panel User Preset State. Choose from:</td>
</tr>
<tr>
<td>0</td>
<td>User Preset OFF</td>
</tr>
<tr>
<td>1</td>
<td>User Preset ON</td>
</tr>
</tbody>
</table>

**Examples**

```plaintext
SYST:UPR:FPAN 1
system:upreset:fpanel:state 0
```

**Query Syntax**

SYSTem:UPREset:FPANel[:STATe]?

**Return Type**

Boolean

**Default**

0

**SYSTem:UPReset:LOAD <file>**

(Read-Write) Loads an existing instrument state file (.sta or .cst) to be used for User Preset. Subsequent execution of SYSTem:UPReSet will cause the PNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the SYST:PRESet command will always preset the PNA to the factory preset settings, and SYST:UPReset will always perform a User Preset.

Learn more about User Preset.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;file&gt;</td>
<td>String - Name of the file to be loaded. The default folder &quot;C:\Program Files\Agilent\Network Analyzer\Documents&quot; is used if unspecified. Change the default folder name using MMEMory:CDIRectory.</td>
</tr>
</tbody>
</table>

**Examples**

```plaintext
SYST:UPR:LOAD '1MHzto20GHzUserPreset.cst'

system:upreset:load 'C:\Documents and Settings\Administrator\My Documents\NewUserPreset.cst'
```

**Query Syntax**

Not Applicable

**Default**

Not Applicable
**SYSTem:UPReset:SAVE[:STATE]**

(Write-only) Saves the current instrument settings as UserPreset.sta. Subsequent execution of **SYSTem:UPReset** will cause the PNA to assume this instrument state.

Regardless of the state of the User Preset Enable checkbox, the **SYST:PRESet** command will always preset the PNA to the factory preset settings, and **SYST:UPReset** will always perform a User Preset.

Learn more about User Preset.

| Examples | SYST:UPR:SAVE  
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>system:upreset:save:state</code></td>
</tr>
</tbody>
</table>

**Query Syntax** Not Applicable

**Default** Not Applicable

---

**SYSTem:SHORtcut<n>:ARGuments<string>**

(Read-Write) Reads and writes the arguments for the specified macro. On the Edit Macro Dialog, this is called the "Macro run string parameters".

**Parameters**

- `<n>` Numeric. Number of the macro that is stored in the PNA.
  - To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all 12 macros for the desired macro title.
- `<string>` Arguments for the specified macro.

**Examples**

- **SYST:SHOR1:ARG**

**Query Syntax** SYSTem:SHORtcut<n>:ARGuments?

**Default** Not Applicable

---

**SYSTem:SHORtcut<n>:DELeTe**
(Write-only) Removes the specified macro from the list of macros in the PNA. Does not delete the macro executable file.

**Parameters**

<n> Numeric. Number of the macro that is stored in the PNA.

To find the number of a macro, either open the [Macro Setup dialog](#) and count the line number of the desired macro, or query the titles of all 12 macros for the desired macro title.

**Examples**

SYST:SHOR1:DEL

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SYSTem:SHORtcut<n>:EXECute**

(Write-only) Executes (runs) the specified Macro (shortcut) that is stored in the PNA.

**Parameters**

<n> Numeric. Number of the macro that is stored in the PNA.

To find the number of a macro, either open the [Macro Setup dialog](#) and count the line number of the desired macro, or query the titles of all 12 macros for the desired macro title.

**Examples**

SYST:SHOR1:EXEC

**Query Syntax**

Not Applicable

**Default**

Not Applicable

---

**SYSTem:SHORtcut<n>:PATH <string>**
(Read-Write) Defines a Macro (shortcut) by linking a path and file name to the Macro number. To be executed, the executable file must be put in the PNA at the location indicated by this command.

**Parameters**

- `<n>` Numeric. Number of the macro to be stored in the analyzer. If the index number already exists, the existing macro is replaced with the new macro.
- `<string>` Full path, file name, and extension, of the existing macro "executable" file.

To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all 12 macros for the desired macro title.

**Examples**

```
SYST:SHOR1:PATH "C:\Program Files\Agilent\Network Analyzer\Documents\unguideMultiple.vbs"
```

**Query Syntax**

```
SYSTem:SHORcut<n>:PATH?
```

**Default** Not Applicable

---

**SYSTem:SHORcut<n>:TITLe<string>**

(Read-Write) Reads and writes the name of the specified macro.

**Parameters**

- `<n>` Numeric. Number of the macro that is stored in the PNA.

To find the number of a macro, either open the Macro Setup dialog and count the line number of the desired macro, or query the titles of all 12 macros for the desired macro title.

- `<string>` The name to be assigned to the macro.

**Examples**

```
SYST:SHOR1:TITL "Guided 4-Port Cal"
```

**Query Syntax**

```
SYSTem:SHORcut<n>:TITLe?
```

**Default** Not Applicable

---

**SYSTem:TOUChscreen[:STATe] <bool>**
(Read-Write) Enables and disables the **PNA-X** touchscreen.

This setting remains until changed again from the front-panel or remotely, or until the hard drive is changed or reformatted.

**Parameters**

<bool> Choose from:

- **True (1)** Enables the touchscreen.
- **False (0)** Disables the touchscreen.

**Examples**

```bash
SYST:TOUC 1
system:touchscreen:state false
```

**Query Syntax**

SYSTem:TOUCHscreen[:STATe]?

**Return Type**

Boolean

**Default**

TRUE when shipped from factory.

---

Last modified:

- 811-Feb-2008   Added Noise switch preference
- 5-Feb-2007     Added Extra security and USB power meter commands
- February 23, 2007   MX Added touchscreen command
- 15-Nov-2006     MX Added Unleveled Error reporting
- 31-Oct-2006     Added PSRTTrace command
Trigger Commands

Controls the starts or ends of data acquisition.

<table>
<thead>
<tr>
<th>TRIGger:</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUX</td>
</tr>
<tr>
<td>COUNt</td>
</tr>
<tr>
<td>CHANnel:AUX</td>
</tr>
<tr>
<td>DELay</td>
</tr>
<tr>
<td>DURation</td>
</tr>
<tr>
<td>ENABLE</td>
</tr>
<tr>
<td>HANDshake</td>
</tr>
<tr>
<td>INTerval</td>
</tr>
<tr>
<td>IPOLarity</td>
</tr>
<tr>
<td>OPOLarity</td>
</tr>
<tr>
<td>POSition</td>
</tr>
<tr>
<td>TYPE</td>
</tr>
<tr>
<td>DELay</td>
</tr>
<tr>
<td>PREFerence</td>
</tr>
<tr>
<td>AIGHLobal</td>
</tr>
<tr>
<td>ROUTe</td>
</tr>
<tr>
<td>INPut</td>
</tr>
<tr>
<td>READY</td>
</tr>
<tr>
<td>[SEQuence]</td>
</tr>
<tr>
<td>LEVel</td>
</tr>
<tr>
<td>SCOPe</td>
</tr>
<tr>
<td>SOURce</td>
</tr>
</tbody>
</table>

Click on a blue keyword to view the command details.
Red commands are superseded.

See Also

- **Example program** [Triggering the PNA](#)
- **See other SCPI Triggering commands**
- **Learn about External / Aux Triggering**
- **Synchronizing the PNA and Controller**
- **SCPI Command Tree**

---

**TRIGger:AUX:COUNt?**

*(Read-only)* Returns the number of AUX trigger input / output connector pairs in the instrument.

- **PNA-X =** 2
- **E836xB and PNA-L models =** 1

**Parameters**

<table>
<thead>
<tr>
<th>Examples</th>
<th>TRIG:CHAN:AUX:COUN?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>trigger:channel2:aux:count?</td>
</tr>
</tbody>
</table>

**Return Type** Numeric

**Default** Not Applicable

---

**TRIGger:CHANnel<ch>:AUX<n>:DELay <num>**
(Read-Write) Specifies the delay that should be applied by the PNA after the Aux trigger input is received and before the acquisition is made.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Rear panel connectors used to send or receive signals.

- PNA-X - choose from 1 ([AUX TRIG 1 IN]) or 2 ([AUX TRIG 2 IN])
- All other models: choose 1.

  If unspecified, value is set to 1.

- `<num>` Delay value in seconds. Choose a value between 0 and 3.0 seconds.

**Examples**

<table>
<thead>
<tr>
<th>TRIGger:CHANnel&lt;ch&gt;:AUX&lt;n&gt;:DELay</th>
<th>.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>trigger:channel2:aux2:delay</td>
<td>1.5</td>
</tr>
</tbody>
</table>

**Query Syntax**

TRIGger:CHANnel<ch>:AUX<n>:DELay?

**Return Type** Numeric

**Default** 0

---

**TRIGger:CHANnel<ch>:AUX<n>:DURation <num>**

(Read-Write) Specifies the width of the output pulse, which is the time that the Aux trigger output will be asserted.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Rear panel connector used to send or receive signals.

- PNA-X - choose from 1 ([AUX TRIG 1]) or 2 ([AUX TRIG 2])
- All other models: choose 1. ([BNC Trig I/O]

  If unspecified, value is set to 1.

- `<num>` Duration value in seconds. Choose a value between 1us (1E-6) and 1

**Examples**

<table>
<thead>
<tr>
<th>TRIGger:CHAN: AUX: DUR</th>
<th>.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>trigger:channel2:aux2:duration</td>
<td>.01</td>
</tr>
</tbody>
</table>

**Query Syntax**

TRIGger:CHANnel<ch>:AUX<n>:DURation?
**Return Type**  Numeric

**Default**  1E-6

---

**TRIGger:CHANnel<ch>:AUX<n>[:ENABle] <bool>**

(Read-Write) Turns ON / OFF the trigger output.

**Parameters**
- `<ch>`  Any existing channel number. If unspecified, value is set to 1.
- `<n>`  Rear panel connector used to send or receive signals.
  - PNA-X - choose from 1 ([AUX TRIG 1]) or 2 ([AUX TRIG 2])
  - All other models: choose 1. ([BNC Trig I/O])

If unspecified, value is set to 1.

- `<bool>`  ON (or 1) - turns trigger output ON.

**Examples**

```
TRIG:CHAN:AUX 1
trigger:channel2:aux2:enable off
```

**Query Syntax**  TRIGger:CHANnel<ch>:AUX<n>[:ENABle]?

**Return Type**  Boolean

**Default**  OFF

---

**TRIGger:CHANnel<ch>:AUX<n>:HANDshake <bool>**
(Read-Write) Turns handshake ON / OFF.

To enable handshake, the main trigger enable must also be set using TRIG:CHAN:AUX:ENAB.

When ON, PNA waits indefinitely for the input line to be asserted before continuing with the acquisition. When OFF, the PNA acquires data without waiting.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Rear panel connector used to send or receive signals.
  - PNA-X - choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)
  - All other models: choose 1. (BNC Trig I/O) If unspecified, value is set to 1.
- `<bool>` ON (or 1) - turns handshaking ON.
  - OFF (or 0) - turns handshaking OFF.

Examples

TRIG:CHAN:AUX:HAND 1

trigger:channel2:aux2:handshake off

Query Syntax

TRIGger:CHANnel<ch>:AUX<n>:HANDshake?

Return Type

Boolean

Default

OFF

TRIGger:CHANnel<ch>:AUX<n>:INTerval <char>
(Read-Write) Specifies how often a trigger output signal is sent.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Rear panel connector used to send or receive signals.

  - PNA-X - choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)
  - All other models: choose 1. (BNC Trig I/O)

If unspecified, value is set to 1.

- `<char>` Choose from:
  - **POINT** Trigger signal is sent every data point. (effectively the same as Point sweep)
  - **SWEep** Trigger signal is sent once every sweep.

**Examples**

```
TRIG:CHAN:AUX:INT POI
trigger:channel2:aux2:interval sweep
```

**Query Syntax**

```
TRIGger:CHANnel<ch>:AUX<n>:INTerval?
```

**Return Type**

Character

**Default**

SWEep

---

**TRIGger:CHANnel<ch>:AUX<n>:IPOLarity  <char>**

(Read-Write) Specifies the polarity of the trigger IN signal to which the PNA will respond.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Rear panel connector used to send or receive signals.

  - PNA-X - choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)
  - All other models: choose 1. (BNC Trig I/O)

If unspecified, value is set to 1.

- `<char>` Choose from:
  - **POSitive** PNA responds to leading edge or HIGH level
- **NEGative**  PNA responds to trailing edge or LOW level.

Set Edge or Level triggering using **TRIG:CHAN:AUX:TYPE**

**Examples**

```
TRIG:CHAN:AUX:IPOL POS
```

```
trigger:channel2:aux2:ipolarity negative
```

**Query Syntax**  TRIGger:CHANnel<ch>:AUX<n>:IPOLarity?

**Return Type**  Character

**Default**  NEGative

---

**TRIGger:CHANnel<ch>:AUX<n>:OPOLarity <char>**

(Read-Write)  Specifies the polarity of the Aux Output signal being supplied by the PNA.

**Parameters**

- `<ch>`  Any existing channel number.  If unspecified, value is set to 1.
- `<n>`  Rear panel connector used to send or receive signals.

  - PNA-X - choose from 1 ([AUX TRIG 1]) or 2 ([AUX TRIG 2])
  - All other models: choose 1. ([BNC Trig I/O]

If unspecified, value is set to 1.

- `<char>`  Choose from:

  - **POSitive**  PNA sends positive going pulse.
  - **NEGative**  PNA sends negative going pulse.

**Examples**

```
TRIG:CHAN:AUX:OPOL NEG
```

```
trigger:channel2:aux2:opolarity positive
```

**Query Syntax**  TRIGger:CHANnel<ch>:AUX<n>:OPOLarity?

**Return Type**  Character

**Default**  NEGative

---

**TRIGger:CHANnel<ch>:AUX<n>:POSition <char>**
(Read-Write) Specifies whether the aux trigger out signal is sent BEFore or AFTer the acquisition.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Rear panel connector used to send or receive signals.

- **PNA-X** - choose from 1 ([AUX TRIG 1]) or 2 ([AUX TRIG 2])
- **All other models:** choose 1. ([BNC Trig I/O])

If unspecified, value is set to 1.

- `<char>` Choose from:
  - **BEFore** Use if the external device needs to be triggered before the data is acquired, such as a power meter.
  - **AFTer** Use if the external device needs to be triggered just after data has been acquired, such as an external source. This could be more efficient since it allows the external device to get ready for the next acquisition at the same time as the PNA.

**Examples**

```
TRIG:CHAN:AUX:POS BEF
```

```
trigger:channel2:aux2:position after
```

**Query Syntax**

TRIGger:CHANnel<ch>:AUX<n>:POSition?

**Return Type**

Character

**Default**

AFTer
(Read-Write) Specifies the type of Aux input detection that the PNA will employ.

Parameters

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<n>` Rear panel connector used to send or receive signals.

- **PNA-X** - choose from 1 (AUX TRIG 1) or 2 (AUX TRIG 2)
- **All other models** - choose 1. (BNC Trig I/O)

If unspecified, value is set to 1.

- `<char>` Choose from:
  - **EDGE** PNA responds to the leading edge of a signal
  - **LEVel** PNA responds to the level (HIGH or LOW) of a signal

**Examples**

```
TRIG:CHAN:AUX:TYPE EDGE
trigger:channel2:aux2:type level
```

**Query Syntax**

TRIGger:CHANnel<ch>:AUX<n>:TYPE?

**Return Type** Character

Default EDGE

---

**TRIGger:DELay <num>**

(Read-Write) Sets and reads the trigger delay. This delay is only applied while in External Trigger mode. After an external trigger is applied, the start of the sweep is held off for an amount of time equal to the delay setting plus any inherent latency.

Parameters

- `<num>` Delay value in seconds. Choose from 0 to 107.

**Examples**

```
TRIG:DEL .0003
```

Sets the trigger delay to 300 microseconds. The sweep will not start until approximately 300 microseconds after an external trigger is applied.

**Query Syntax**

TRIGger:DELas

**Return Type** Numeric

Default 0
**TRIGger:CHANnel<ch>:ROUTe:INPut <char>**

(Read-Write) Specifies the connector to use for the external trigger input. (Similar to CONTROL:SIGNal <conn> )

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Choose from:
  - **MAIN** - meas trig ready
  - **MATH** - material handler
  - **AUXT** - auxio (PNA-L and E836xB ONLY)

**Examples**

```
TRIG:CHAN:ROUT:INP MAIN
trigger:channel2:route:input main
```

**Query Syntax**

```
TRIGger:CHANnel<ch>:ROUTe:INPut?
```

**Return Type** Character

**Default**

---

**TRIGger:CHANnel<ch>:ROUTe:READy <char>**

(Read-Write) Specifies the connector to use for the external trigger OUT ready line.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Choose from:
  - **MAIN** - Meas trig ready
  - **MATH** - Material handler pin 21
  - **AUXT** - AUX I/O pin 18 (PNA-L and E836xB ONLY)

**Examples**

```
TRIG:CHAN:ROUT:READ main
trigger:channel2:route:ready auxt
```

**Query Syntax**

```
TRIGger:CHANnel<ch>:ROUTe:READy?
```

**Return Type** Character

**Default**
TRIGger:PREFerence:AIGLobal <bool>
(Read-Write) Sets the Trigger OUT behavior to either Global or Channel. Learn more about this setting.

This command will cause the PNA to Preset.

This setting remains until changed again using this command, or until the hard drive is changed or reformatted.

To send this command using the PNA front panel, open the GPIB Command Processor Console, then type either of the following examples at the command prompt. Then type the Query Syntax and press enter to be sure the PNA took the command.

**Parameters**

<bool> Choose from:

- **ON** (or 1) - Trigger properties apply to ALL channels (Global).
  - Allows use of CONT:SIGNal command to configure the external trigger properties.
  - "Per Point" trigger property is not settable. Use the channel's Point trigger setting.
  - Default setting for E836xB and PNA-L models.
- **OFF** (or 0) - External Trigger properties apply to each channel independently.
  - Must use TRIG:CHAN:AUX commands to configure the external trigger properties. CONT:SIGNal will NOT work.
  - "Per Point" trigger output property is set using the channel's Point trigger setting AND TRIG:CHAN:AUX:INTerval.
  - Default setting for PNA-X models.

**Examples**

<table>
<thead>
<tr>
<th>TRIGger:PREFerence:AIGLobal 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>trigger:preference:aiglobal 0</td>
</tr>
</tbody>
</table>

**Query Syntax**  TRIGger:PREFerence:AIGLobal?

**Return Type**  Boolean

**Default**  1 - E836xB and PNA-L models

0 - PNA-X models
**TRIgger[:SEQuence]:LEVel <char> - Superseded**

This command is replaced with **CONTrol:SIGNal**

(Read-Write) Triggers either on a **High** or **Low** level trigger signal. This setting only has an effect when **TRIg:SOURce EXTernal** is selected.

### Parameters

<char> Choose from:

- **HIGH** - analyzer triggers on TTL High
- **LOW** - analyzer triggers on TTL Low

### Examples

```
TRIg:LEV HIGH
trigger:sequence:level low
```

### Query Syntax

**TRIgger[:SEQuence]:LEVel?**

**Return Type** Character

**Default** LOW

---

**TRIgger[:SEQuence]:SCOPe <char>**

(Read-Write) Specifies whether a trigger signal is sent to all channels or only the current channel.

See **Triggering the PNA using SCPI**.

### Parameters

<char> Choose from:

- **ALL** - trigger signal is sent to all channels. Also sets **SENS:SWEep:TRIg:POINT OFF** on ALL channels.
- **CURRENT** - trigger signal is sent to only one channel at a time. With each trigger signal, the channel is incremented to the next triggerable channel.

### Examples

```
TRIg:SCOPE ALL
trigger:sequence:SCOPE current
```

### Query Syntax

**TRIgger[:SEQuence]:SCOPe?**

**Return Type** Character

**Default** ALL
**TRIGger[:SEQuence]:SOURce <char>**

(Read-Write) Sets the source of the sweep trigger signal. This command is a super-set of `INITiate:CONTinuous`, which can NOT set the source to External. To configure external triggering, see `CONTrol:SIGNal`.

See [Triggering the PNA using SCPI](#).

**Parameters**

- `<char>` Choose from:
  - **EXTERNAL** - external (rear panel) source.
  - **IMMediate** - internal source sends continuous trigger signals
  - **MANual** - sends one trigger signal when manually triggered from the front panel or `INIT:IMM` is sent.

**Examples**

```
TRIG:SOUR EXT
trigger:sequence:source immediate
```

**Query Syntax**

`TRIGger[:SEQuence]:SOURce?`

**Return Type**

Character

**Default**

IMMediate

---

**TRIGger:TYPE <char>**

(Read-Write) Specifies the type of EXTERNAL trigger input detection used to listen for signals on the Ext I/O Trig and Meas Trig IN connectors. Edge triggers are most commonly used.

**Parameters**

- `<ch>` Any existing channel number. If unspecified, value is set to 1.
- `<char>` Choose from:
  - **EDGE** PNA responds to the rising and falling edge of a signal
  - **LEVEL** PNA responds to a level (HIGH or LOW).

**Examples**

```
TRIG:TYPE EDGE
trigger:type level
```

**Query Syntax**

`TRIGger:TYPE?`

**Return Type**

Character

**Default**

EDGE
Last modified:

22-Feb-2008  Clarified AIGL command
24-Apr-2007  Clarified trigger source and scope
15-Feb-2007  MX Updated for AUX triggering
Catalog Measurements using SCPI

This Visual Basic Program does the following:

- Catalogs the currently defined measurements, windows, and traces
- Selects a measurement for further definition
- Adds a Title to the window

To run this program, you need:

- An established GPIB interface connection

```vbnet
Dim Meas as String
Dim Win as String
Dim Trace as String

'Read the current measurements in Channel 1
GPIB.Write "CALCulate1:PARameter:CATalog?"
Meas = GPIB.Read
MsgBox ("Ch1 Measurements: " & Meas)

'Read the current windows
GPIB.Write "DISPlay:CATalog?"
Win = GPIB.Read
MsgBox ("Windows: " & Win)

'Read current traces in window 1
GPIB.Write "DISPlay:WINDow1:CATalog?"
Trace = GPIB.Read
MsgBox("Traces in Window1: " & Win)
```
Create an FOM Measurement

All three VBScript examples in this topic create a FOM measurement with the following attributes:

- Sweep the Source (input) from 1 GHz to 2 GHz
- Sweep the Receivers (output) from 2 GHz to 3 GHz
- You provide an LO at 1 GHz

Learn more about Frequency Offset Mode

These programs can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as FOM.vbs. Learn how to setup and run the macro.

See Other SCPI Example Programs

The following example will run on any PNA model with FOM (opt 080). However, these commands have no provisions for internal second source. It uses Sens:Offset commands introduced before 'enhanced FOM' was released for the A.07.10 release.

```vbscript
' This section gets the PNA application
' starts the scpi parser, and presets the PNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:FPRESET")

' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")

'Define a measurement name, parameter
scpi.Execute ("CALCulate:PARameter:DEFine 'MyMeas',S21")

'Associate ("FEED") the measurement name ("MyMeas") to WINDow (1)
'and give the new TRACe a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")

scpi.Execute ("SENS:FREQ:START 1e9")
scpi.Execute ("SENS:FREQ:STOP 2e9")

' set the receivers to be 2e9 -> 3e9
scpi.Execute ("SENS:OFFS:OFFS 1e9")
scpi.Execute ("SENS:OFFS ON")
```
The following example can be run ONLY on a PNA with revision A.07.10 or later and has FOM (opt 080). It uses new Sens:FOM commands.

' This section gets the PNA application
' starts the scpi parser, and presets the PNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:FPRESET")
' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
' Define a measurement name, parameter
scpi.Execute ("CALCulate:PARameter:DEFine 'MyMeas',S21")
' Associate ("FEED") the measurement name ("MyMeas") to WINDow (1), and give the new TRACe a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")
scpi.Execute("SENS:FREQ:START 1e9")
scpi.Execute("SENS:FREQ:STOP 2e9")
'set the receivers to be 2e9 -> 3e9
scpi.Execute("SENS:FOM:RANG3:FREQ:OFFS 1e9")
scpi.Execute("SENS:OFFS ON")

The following example can be run ONLY on a PNA with a second internal source, has revision A.07.10 or later, and has FOM (opt 080). It uses the internal 2nd source for the fixed LO frequency.

' This section gets the PNA application
' starts the scpi parser, and presets the PNA
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
scpi.Execute("SYST:FPRESET")
' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
' Define a measurement name, parameter
scpi.Execute ("CALCulate:PARameter:DEFine 'MyMeas',S21")
' Associate ("FEED") the measurement name ("MyMeas") to WINDow (1)
' and give the new TRACe a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")
scpi.Execute("SENS:FREQ:START 1e9")
scpi.Execute("SENS:FREQ:STOP 2e9")
'set the receivers to be 2e9 -> 3e9
scpi.Execute("SENS:FOM:RANG3:FREQ:OFFS 1e9")
'setup the 2nd source frequencies
scpi.Execute("SENS:FOM:RANG4:COUP 0")
scpi.Execute("SENS:FOM:RANG4:FREQ:START 1e9")
scpi.Execute("SENS:FOM:RANG4:FREQ:STOP 1e9")
'turn off coupling
scpi.Execute("SOUR:POW:COUP 0")
' set LO power to 10dBm
scpi.Execute("SOUR:POW3 10")
'turn ON port 3, our LO signal
scpi.Execute("SOUR:POW3:MODE ON")
scpi.Execute("SENS:FOM:STAT ON")
Create a Measurement using SCPI

This VBScript program creates a new S21 measurement and displays it on the PNA screen. The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as NewMeas.vbs. Learn how to setup and run the macro.

See Other SCPI Example Programs

```
Dim app
Dim scpi

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' A comment
' Preset the analyzer
scpi.Execute ("SYST:FPReset")
' Create and turn on window 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
' Define a measurement name, parameter
scpi.Execute ("CALCulate:PARameter:DEFine 'MyMeas',S21")
' Associate ("FEED") the measurement name ("MyMeas") to WINDow (1), and give the new TRACe a number (1).
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas'")
```
Create a Balanced Measurement using SCPI

This example program does the following:

- creates several Balanced measurements in separate windows
- generates markers
- calculates statistics
- sets limit lines and queries results
- queries a measurement to determine if we have a balanced parameter and what type it is.

**Note:** By their nature, balanced measurements are extremely sensitive to phase differences between the two RF paths that make up the balanced port, especially at higher frequencies. A good calibration (not performed in this example) is critical to achieving good balanced measurement results.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Balanced.vbs. Learn how to setup and run the macro.

See Other SCPI Example Programs

```vbs
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
' A comment
scpi.Parse("SYST:FPRESET")
' Put in "Single" trigger mode.
scpi.Parse("SENS:SWE:GROups:COUNt 1")
scpi.Parse("SENS:SWE:MODE GROUPS")

' This example uses DUT topology Bal-Bal -
' a DUT with a balanced input and balanced output.

' Port mapping for our DUT:
' logical port 1 = physical ports 1 and 4
' logical port 2 = physical ports 2 and 3
' The default is:
' logical port 1 = physical ports 1 and 2
' logical port 2 = physical ports 3 and 4

' logical 1               logical 2
' 1 --------[___________]-------- 2 +
'          |         DUT         |
' 4 --------[___________]-------- 3 -
```
scpi.Parse("CALC:FSIM:BAL:DEVi ce BBALanced")
scpi.Parse("CALC:FSIM:BAL:TOPology:BBAL:PORts 1,4,2,3")

' Set up stimulus
scpi.Parse("SENS:SWE:POINts 801")
scpi.Parse("SENS:FREQ:STARt 10e6")
scpi.Parse("SENS:FREQ:STOP  1e9")

' Turn on Four windows
scpi.Parse("DISP:WIND1:STATe ON")
scpi.Parse("DISP:WIND2:STATe ON")
scpi.Parse("DISP:WIND3:STATe ON")
scpi.Parse("DISP:WIND4:STATe ON")

' Create a trace called "sdd21", and for that trace turn on the balanced
' transformation and set the balanced transformation to BBAL SDD21.
scpi.Parse("CALC:PAR:DEF ""sdd21"",S11")
scpi.Parse("CALC:PAR:SEL ""sdd21""")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")

' Feed the sdd21 trace to window 1, trace 1
scpi.Parse("DISP:WIND1:TRAC1:FEED ""sdd21"")

' Similarly create 3 more balanced transmission/conversion parameters
' Create Scd21
scpi.Parse("CALC:PAR:DEF ""scd21"",S11")
scpi.Parse("CALC:PAR:SEL ""scd21"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND1:TRAC2:FEED ""scd21"")

' Create Sdc21
scpi.Parse("CALC:PAR:DEF ""sdc21"",S11")
scpi.Parse("CALC:PAR:SEL ""sdc21"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND1:TRAC3:FEED ""sdc21"")

' Create Scc21
scpi.Parse("CALC:PAR:DEF ""scc21"",S11")
scpi.Parse("CALC:PAR:SEL ""scc21"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND1:TRAC4:FEED ""scc21"")

' Now create logical port 1 reflection parameters, and place them in window 2
scpi.Parse("CALC:PAR:DEF ""sdd11"",S11")
scpi.Parse("CALC:PAR:SEL ""sdd11"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")

' Feed the sdd11 trace to window 2, trace 1
scpi.Parse("DISP:WIND2:TRAC1:FEED ""sdd11"")

' Similarly create 3 more balanced reflection/conversion parameters
scpi.Parse("CALC:PAR:DEF ""sdc11"",S11")
scpi.Parse("CALC:PAR:SEL ""sdc11"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND2:TRAC2:FEED ""sdc11"")
scpi.Parse("CALC:PAR:DEF ""sdc11"",S11")

2367
scpi.Parse("CALC:PAR:SEL ""sdc11"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND2:TRAC3:FEED ""sdc11"")
scpi.Parse("CALC:PAR:DEF ""scc11", S11")
scpi.Parse("CALC:PAR:SEL ""scc11"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND2:TRAC4:FEED ""scc11"")
'
Now create reverse transmission parameters, and place them in window 3
scpi.Parse("CALC:PAR:DEF ""sdd12", S11")
scpi.Parse("CALC:PAR:SEL ""sdd12"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND3:TRAC1:FEED ""sdd12"")
'
Feed the sdd11 trace to window 3, trace 1
scpi.Parse("DISP:WIND3:TRAC1:FEED ""sdd12"")
'
Similarly create 3 more balanced reverse transmission/conversion parameters
scpi.Parse("CALC:PAR:DEF ""scd12", S11")
scpi.Parse("CALC:PAR:SEL ""scd12"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("CALC:FSIM:BAL:PAR:BBAL:DEF SCD12")
scpi.Parse("DISP:WIND3:TRAC2:FEED ""scd12"")
scpi.Parse("CALC:PAR:DEF ""sdc12", S11")
scpi.Parse("CALC:PAR:SEL ""sdc12"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND3:TRAC3:FEED ""sdc12"")
scpi.Parse("CALC:PAR:DEF ""scc12", S11")
scpi.Parse("CALC:PAR:SEL ""scc12"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND3:TRAC4:FEED ""scc12"")
'
Now create reverse reflection parameters, and place them in window 4
scpi.Parse("CALC:PAR:DEF ""sdd22", S11")
scpi.Parse("CALC:PAR:SEL ""sdd22"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND4:TRAC1:FEED ""sdd22"")
'
Feed the sdd11 trace to window 3, trace 1
scpi.Parse("DISP:WIND4:TRAC1:FEED ""sdd22"")
'
Similarly create 3 more balanced reverse reflection parameters
scpi.Parse("CALC:PAR:DEF ""scd22", S11")
scpi.Parse("CALC:PAR:SEL ""scd22"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND4:TRAC2:FEED ""scd22"")
scpi.Parse("CALC:PAR:DEF ""sdc22", S11")
scpi.Parse("CALC:PAR:SEL ""sdc22"")
scpi.Parse("CALC:FSIM:BAL:PAR:STATe ON")
scpi.Parse("DISP:WIND4:TRAC3:FEED ""sdc22"")
scpi.Parse("CALC:PAR:DEF ""scc22", S11")
scpi.Parse("CALC:PAR:SEL ""scc22"")

Show statistics for forward differential transmission.

Set up test range for markers. Set up range 1 to be between 100MHz and 900MHz.

Set up a marker on Sdd21, find minimum differential transmission.

Set this marker to use user-range 1 (previously defined).

Set up limit lines.

'MIN' Limit on Sdd21 between 100MHz and 900MHz at -2dB.

'MAX' Limit on Sdc21 between 100MHz and 900MHz at -20dB.

'Single' trigger to take data.

Read the limit status byte to determine pass/fail.

If bit 1 is set (+2), then our first measurement failed (Sdd21).
If bit 3 is set (+8), then our third measurement failed (Sdc21).
Both measurements failing would return +10.

Here we demonstrate how to determine if we have
a balanced parameter and what type it is.
Read back one parameter to verify its type
scpi.Parse("CALC:PAR:SEL ""sdd21""")
Is this a balanced parameter?
  isbal = scpi.Parse("CALC:FSIM:BAL:PAR?")
Which topology/device is set?
device = scpi.Parse("CALC:FSIM:BAL:DEV?")
  device = Left( device, Len(device)-1 ) ' strip off newline
Which parameter are we measuring within that topology?
balparam = scpi.Parse("CALC:FSIM:BAL:PAR:" & device & ":DEF?")
  balparam = Left( balparam, Len(balparam)-1 ) ' strip off newline
If isbal Then
  WScript.Echo "Balanced Parameter: " & balparam & " in topology: " & device & "."
Else
  WScript.Echo "Parameter not balanced."
End If
Channels, Windows, and Measurements using SCPI

SOURce and most SENSe commands act on the channel that is specified in the command. Channel 1 is default if not specified.

Most DISPlay commands act on the window and trace specified in the command. Window1 and Trace1 are default if not specified.

CALCulate commands act on the selected measurement in the specified channel. Select the measurement for each channel using CALCulate<channel number>:PARameter:SELect <meas name>. You can select one measurement in each channel.

The following Visual Basic program does the following:

- Presets the analyzer
- Create 2 windows
- Create 2 Measurements
- Feed the measurements to windows / traces
- Change frequency ranges for channels
- Select both measurements
- Turn marker 1 ON for each measurement

To run this program, you need:

- An established GPIB interface connection

See Other SCPI Example Programs

GPIB.Write "SYSTem:PRESet"

'Create Measurements
GPIB.Write "CALCulate1:PARameter:DEFine 'Meas1',S11"
GPIB.Write "CALCulate2:PARameter:DEFine 'Meas2',S21"

'Turn on windows - creates if new
GPIB.Write "DISPlay:WINDow1:STATE ON"
GPIB.Write "DISPlay:WINDow2:STATE ON"

'Associate ("FEED") the measurement name('Meas1') to WINDow(1), and give the new TRACe a number(1).
GPIB.Write "DISPlay:WINDow1:TRACe1:FEED 'Meas1'"
GPIB.Write "DISPlay:WINDow2:TRACe2:FEED 'Meas2'"

'Change each channel's frequency range
GPIB.Write "SENSe1:FREQuency:SPAN 1e9"
GPIB.Write "SENSe2:FREQuency:SPAN 2e9"
Select both measurements
GPIB.Write "CALCulate1:PARameter:SELect 'Meas1'"
GPIB.Write "CALCulate2:PARameter:SELect 'Meas2'"

Turn marker 1 ON for each measurement
GPIB.Write "CALCulate1:MARKer:STATE ON"
GPIB.Write "CALCulate2:MARKer:STATE ON"
Setup Sweep Parameters using SCPI

This Visual Basic program sets up sweep parameters on the Channel 1 measurement. To run this program, you need:

- An established [GPIB interface connection](#)

---

See Other SCPI Example Programs

```plaintext
GPIB.Write "SYSTem:PRESet"
'Select the measurement
GPIB.Write "CALCulate:PARameter:SELect 'CH1_S11_1'"
'Set sweep type to linear
GPIB.Write "SENSe1:SWEep:TYPE LIN"

'Set IF Bandwidth to 700 Hz
GPIB.Write "SENSe1:BANDwidth 700"

'Set Center and Span Freq's to 4 GHz
GPIB.Write "SENSe1:FREQuency:CENTer 4ghz"
GPIB.Write "SENSe1:FREQuency:SPAN 4ghz"

'Set number of points to 801
GPIB.Write "SENSe1:SWEep:POINts 801"

'Set sweep generation mode to Analog
GPIB.Write "SENSe1:SWEep:GENeration ANAL"

'Set sweep time to Automatic
GPIB.Write "SENSe1:SWEep:TIME:AUTO ON"

'Query the sweep time
GPIB.Write "SENSe1:SWEep:TIME?"
SweepTime = GPIB.Read
```
Setup the Display using SCPI

This Visual Basic program:

- Sets data formatting
- Turns ON the Trace, Title, and Frequency Annotation
- Autoscales the Trace
- Queries Per Division, Reference Level, and Reference Position
- Turn ON and set averaging
- Turn ON and set smoothing

To run this program, you need:

- An established GPIB interface connection

See Other SCPI Example Programs

GPIB.Write "SYSTem:PRESet"

'Select the measurement
GPIB.Write "CALCulate:PARameter:SELect 'CH1_S11_1'"

'Set the Data Format to Log Mag
GPIB.Write ":CALCulate1:FORMat MLOG"

'Turn ON the Trace, Title, and Frequency Annotation
GPIB.Write "DISPLAY:WINDow1:TRACe1:STATe ON"
GPIB.Write "DISPLAY:WINDow1:TITle:STATe ON"
GPIB.Write "DISPLAY:ANNotation:FREQuency ON"

'Autoscale the Trace
GPIB.Write "DISPLAY:WINDow1:TRACe1:Y:Scale:AUTO"

'Query back the Per Division, Reference Level, and Reference Position
GPIB.Write "DISPLAY:WINDow1:TRACe1:Y:SCALE:PDIVision?"
Pdiv = GPIB.Read
GPIB.Write "DISPLAY:WINDow1:TRACe1:Y:SCALE:RLEVel?"
Rlev = GPIB.Read
GPIB.Write "DISPLAY:WINDow1:TRACe1:Y:SCALE:RPOSition?"
Ppos = GPIB.Read

'Turn ON, and average five sweeps
GPIB.Write "SENSe1:AVERage:STATe ON"
GPIB.Write "SENSe1:AVERage:Count 5"

'Turn ON, and set 20% smoothing aperture
GPIB.Write "CALCulate1:SMOothing:STATe ON"
GPIB.Write "CALCulate1:SMOothing:APERture 20"
Triggering the PNA using SCPI

To understand how to trigger the PNA using SCPI, it is very important to understand the PNA trigger model. Here is a very simple explanation. These three separate functions control PNA triggering:

1. **Trigger:Source** - Where the trigger signals originate:
   - Internal Continuous
   - Internal Manual (Single)
   - External - a trigger source that is connected to the PNA rear panel.

2. **Trigger:Scope** - what gets triggered:
   - Global - each signal triggers all channels in turn.
   - Channel - each signal triggers ONE channel.

3. Channel settings (Sense<ch>:Sweep:Mode) How many triggers will each channel accept before going into hold.
   - HOLD - channel will not trigger.
   - CONTinuous - channel triggers indefinitely.
   - GROups - channel accepts the number of triggers specified with the last SENS:SWE:GRO:COUN <num>.
   - **SINGle** - channel accepts ONE trigger, then goes to HOLD.
   - Point trigger SENS1:SWE:TRIG:POIN

When controlling the PNA using SCPI, a SINGLE trigger is used to ensure that a complete sweep is taken. This example demonstrates how to Single trigger the PNA using two methods.

- **Simplest Triggering** This method sets the Trigger Source to Internal Continuous - a stream of trigger signals. Each channel is configured to ACCEPT only a single trigger signal, then HOLD. This method can also be used when an External trigger source sends a continuous stream of trigger signals.

- **Advanced Triggering** This method SENDS a single trigger from the Source, either Internal (using INIT:IMM) or External triggering. Each channel is configured to accept an unlimited number of triggers. This method is the only way to perform point triggering. See INIT:IMM Advanced for details when requiring multiple channels to accept continuous and single triggers.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Trigger.vbs. Learn how to setup and run the
**Macro.**

**Measurement setup example:** sets up S11 traces on two channels, 10 points, sweep time of 2 seconds, which allows us to verify that the trace is being triggered.

```vba
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

'============================================
' Setup the PNA
' Preset the analyzer
scpi.Execute ("SYST:FPReset")
' Create and turn on window/channel 1
scpi.Execute ("DISPlay:WINDow1:STATE ON")
'Define a measurement name, parameter
scpi.Execute ("CALCulate1:PARameter:DEFine 'MyMeas1',S11")
' Associate ("FEED") the measurement name ('MyMeas') to WINDow (1)
scpi.Execute ("DISPlay:WINDow1:TRACe1:FEED 'MyMeas1'")
' Create and turn on window/channel 2
scpi.Execute ("DISPlay:WINDow2:STATE ON")
'Define a measurement name, parameter
scpi.Execute ("CALCulate2:PARameter:DEFine 'MyMeas2',S11")
' Associate ("FEED") the measurement name ('MyMeas') to WINDow (2)
scpi.Execute ("DISPlay:WINDow2:TRACe2:FEED 'MyMeas2'")
'Set slow sweep so we can see
scpi.Execute ("SENS1:SWE:TIME 2")
scpi.Execute ("SENS2:SWE:TIME 2")
'set number of points to 10
scpi.Execute ("SENS1:SWE:POIN 10")
scpi.Execute ("SENS2:SWE:POIN 10")

'============================================
' Put both channels in Hold
scpi.Execute ("SENS1:SWE:MODE HOLD")
scpi.Execute ("SENS2:SWE:MODE HOLD")

'============================================
```

2377
'Pick Single Send or Single Accept

resp=Msgbox ("Single Send? - Click No for Single Accept", 4, "PNA Trigger Demo")
If resp=6 Then
SingleSend()
Else
SingleAccept()
End If

Simple Triggering  The following example sends a continuous stream of trigger signals and each PNA channel is set to ACCEPT only a signal trigger signal, then HOLD.

- This example can be used to configure External triggering where the trigger source sends a continuous stream of trigger signals. Configure the type of trigger signal that the PNA responds to using the CONTrol:SIGNal command. The command in this example sets the PNA to respond to HIGH TTL signals at the rear-panel BNC1 trigger IN connector. This command also automatically sets Trigger Source to External Trigger.
- The TRIG SCOPE (Global or Channel) setting is NOT necessary with a continuous stream of trigger signals. The example program directly controls when each channel is triggered.
- Point triggering can NOT be used with a continuous stream of trigger signals because in point triggering the channel will accept as many triggers as necessary to complete ONE full sweep. Use the single SEND example for point triggering.

Sub SingleAccept()
  'PNA sends continuous trigger signals
  scpi.Execute ("TRIG:SOUR IMMEDIATE")
  'Uncomment the following to set External triggering
  'scpi.Execute ("CONT:SIGN BNC1,TILHIGH")
  AcceptOne()
End Sub

Sub AcceptOne()
  'The following command makes the channel immediately sweep
  '*OPC? allows the measurement to complete before the controller sends another command
  scpi.Execute ("SENS1:SWE:MODE SINGLe;*OPC?")
  ' You could do something to ch2 here before sweeping it
  scpi.Execute ("SENS2:SWE:MODE SINGLe;*OPC?")
Advanced Trigger This example section performs Single Send triggering. Here, single triggering is accomplished by sending one trigger signal from the Trigger source and each channel is setup to accept unlimited trigger signals. See the INIT:IMM command for more details.

- Using this method, it is possible to change Trigger:Scope to Global or Channel. Set trigger scope to channel if there is some code to execute between channel measurements. Similarly, this method can be used to set Point triggering. Use this method if there is some code to execute between channel measurements.

- In addition, this method can also be used to perform External triggering if the external trigger source is capable of sending single triggers. See the CONT:SIGNal command to set the type of signal to which the PNA will respond.

- If the external source can only send a continuous stream of trigger signals, then the Single Accept section must be used.

Sub SingleSend()
' Set Source Internal - Manual Triggering
scpi.Execute ("TRIG:SOUR MANual")
' If using an External trigger source that is capable of
' sending SINGLE trigger signals, then uncomment the following.
' This command automatically sets trigger source to External
' scpi.Execute ("CONT:SIGN BNC1,TILHIGH")

' Setup Trigger Scope
' WHAT gets triggered
' Pick one using comments
' Set Channel triggering
' scpi.Execute ("TRIG:SCOpe CURRent")
' Set Global triggering (Default)
scpi.Execute ("TRIG:SCOpe ALL")

' Set Channel Settings
' The channels respond to UNLIMITED trigger signals (Default)
scki.Execute ("SENS:SW:MODE CONTinuous")
scpi.Execute("SENS2:SWE:MODE CONTinuous")

'To do Point trigger on one or more channels, uncomment the following
'Point trigger automatically sets Trig:Scope to Current/Channel
'scki.Execute("SENS1:SWE:TRIG:POINT ON")
'scki.Execute("SENS2:SWE:TRIG:POINT ON")
IntTrig()
End Sub

Sub IntTrig()
'If External triggering, replace this Sub with code
'to single trigger the External Trig Source
Dim resp
'*OPC? allows the measurement to complete before the controller sends another command
scpi.Execute("INITiate:IMMediate;*OPC?")
resp=Msgbox("Another trigger?",1,"PNA Trigger Demo")
If resp=1 Then
IntTrig()
End If
End Sub

Last modified:

18-Jun-2007 Updated with Sens:Swe:Mode Single
June 6, 2007 Changed order and wording
April 24, 2007 Updated with links
Oct. 5, 2006 New topic
GPIB using Visual C++

See Other SCPI Example Programs

/*
 * This example assumes the user's PC has a National Instruments GPIB board. The example is comprised of three basic parts:
 * 
 * 1. Initialization
 * 2. Main Body
 * 3. Cleanup
 *
 * The Initialization portion consists of getting a handle to the PNA and then doing a GPIB clear of the PNA.
 *
 * The Main Body consists of the PNA SCPI example.
 *
 * The last step, Cleanup, releases the PNA for front panel control.
 */

#include <stdio.h>
#include <stdlib.h>

#include <windows.h>
#include "decl-32.h"

#define ERRMSGSIZE 1024 // Maximum size of SCPI command string
#define ARRAYSIZE 1024 // Size of read buffer
#define BDINDEX 0 // Board Index of GPIB board
#define PRIMARY_ADDR_OF_PNA 16 // GPIB address of PNA
#define NO_SECONDARY_ADDR 0 // PNA has no Secondary address
#define TIMEOUT T10s // Timeout value = 10 seconds
#define EOTMODE 1 // Enable the END message
#define EOSMODE 0 // Disable the EOS mode

int pna;
char ValueStr[ARRAYSIZE + 1];
char ErrorMnemonic[21][5] = {
    "EDVR", "ECIC", "ENOL", "EADR", "EARG",
    "ESAC", "EABO", "ENEB", "EDMA", ",",
    "EOIP", "ECAP", "EFSO", ",", "EBUS",
    "ESTB", "ESRQ", ",", ",", ",", "ETAB"};

void GPIBWrite(char* SCPIcmd);
char *GPIBRead(void);
void GPIBCleanup(int Dev, char* ErrorMsg);

int main()
{
    char *opc;
    char *result;
    char *value;

    /*
     * INITIALIZATION SECTION
     *=================================================================
     */

    /*
     * The application brings the PNA online using ibdev. A device handle, pna, is returned and is used in all subsequent calls to the PNA.
     */
    pna = ibdev(BDINDEX, PRIMARY_ADDR_OF_PNA, NOSECONDARY_ADDR, TIMEOUT, EOTMODE, EOSMODE);
    if (ibsta & ERR)
    {
        printf("Unable to open handle to PNA\nibsta = 0x%x iberr = %d\n", ibsta, iberr);
        return 1;
    }

    /*
     * Do a GPIB Clear of the PNA. If the error bit ERR is set in ibsta, call GPIBCleanup with an error message.
     */
    ibclr (pna);
    if (ibsta & ERR)
    {
        GPIBCleanup(pna, "Unable to perform GPIB clear of the PNA");
        return 1;
    }

    /*
     * MAIN BODY SECTION
     *=================================================================
     */

    // Reset the analyzer to instrument preset
    GPIBWrite("SYST:FPRESET");

    // Create S11 measurement
    GPIBWrite("CALC1:PAR:DEF 'My_S11',S11");
// Turn on Window #1
GPIBWrite("DISPlay:WINDow1:STATe ON");

// Put a trace (Trace #1) into Window #1 and 'feed' it from the measurement
GPIBWrite("DISPlay:WINDow1:TRACe1:FEED 'My_S11'");

// Setup the channel for single sweep trigger
GPIBWrite("INITiate1:CONTinuous OFF;*OPC?");
opc = GPIBRead();
GPIBWrite("SENSe1:SWEep:TRIGger:POINT OFF");

// Set channel parameters
GPIBWrite("SENSe1:SWEep:POINts 11");
GPIBWrite("SENSe1:FREQuency:STARt 1000000000");
GPIBWrite("SENSe1:FREQuency:STOP 2000000000");

// Send a trigger to initiate a single sweep
GPIBWrite("INITiate1;*OPC?");
opc = GPIBRead();

// Must select the measurement before we can read the data
GPIBWrite("CALCulate1:PARameter:SELection 'My_S11'");

// Read the measurement data into the "result" string variable
GPIBWrite("FORMat ASCII");
GPIBWrite("CALCulate1:DATA? FDATA");
result = GPIBRead();

// Print the data to the display console window
printf("S11(dB) - Visual C++ SCPI Example for PNA\n\n");
value = strtok(result, ",");
while (value != NULL)
{
    printf("%s\n", value);
    value = strtok(NULL, ",");
}

/*
* ===================================================================
* CLEANUP SECTION
* ===================================================================
*/

/* The PNA is returned to front panel control. */
ibonl(pna, 0);
return 0;

/*
* Write to the PNA
*/
void GPIBWrite(char* SCPIcmd)
{
    int length;
    char ErrorMsg[ERRMSGSIZE + 1];
    length = strlen(SCPIcmd);

    ibwrt(pna, SCPIcmd, length);
    if (ibsta & ERR)
    {
        strcpy(ErrorMsg, "Unable to write this command to PNA:\n");
        strcat(ErrorMsg, SCPIcmd);
        GPIBCleanup(pna, ErrorMsg);
        exit(1);
    }
}

/*
* Read from the PNA
*/
char* GPIBRead(void)
{
    ibrd(pna, ValueStr, ARRAYSIZE);
    if (ibsta & ERR)
    {
        GPIBCleanup(pna, "Unable to read from the PNA");
        exit(1);
    }
    else
        return ValueStr;
}

/*
* After each GPIB call, the application checks whether the call succeeded. If an
* NI-488.2 call fails, the GPIB driver sets the corresponding bit in the global status
* variable. If the call failed, this procedure prints an error message, takes the PNA
* offline and exits.
*/
void GPIBCleanup(int Dev, char* ErrorMsg)
{
    printf("Error : %s\nibsta = 0x%x iberr = %d (%s)\n", ErrorMsg, ibsta, iberr, ErrorMnemonic[iberr]);
    if (Dev != -1)
    {
        printf("Cleanup: Returning PNA to front panel control\n");
        ibonl(Dev, 0);
    }
}
Perform a Guided 2-Port or 4-Port Cal using SCPI

This example performs a Guided 2-Port or 4-port Calibration using ONE set of calibration standards or an ECAL module.

A measurement must first be set up with desired frequency range, power, and so forth, ready to be calibrated. The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as *.vbs.

Learn how to setup and run the macro.

See Guided Cal SCPI commands

See Other SCPI Example Programs

Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' To perform 2-port cal, Uncomment TwoPortGuidedCal()
' Then comment FourPortGuidedCal()

'Do 2-port Cal
'TwoPortGuidedCal()

'Do 4-port Cal
FourPortGuidedCal

Sub TwoPortGuidedCal()
' Select the connectors
scpi.Execute("sens:corr:coll:guid:conn:port1 ""APC 3.5 female"")
scpi.Execute("sens:corr:coll:guid:conn:port2 ""APC 3.5 male"")
scpi.Execute("sens:corr:coll:guid:conn:port3 ""Not used"")
MsgBox("Connectors defined for Ports 1 and 2")
' Select the Cal Kit for each port being calibrated.
' To use an ECal module instead, comment out the above two lines
' and uncomment these two lines and use the part number printed
' on your module (which in our case was N4691-60004), followed
' by the word 'Ecal'. Your ECal module must already be connected
' via USB to the PNA.
MsgBox("Cal kits defined for Ports 1 and 2")
' Initiate the calibration and query the number of steps
numSteps = GenerateSteps()
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Sub FourPortGuidedCal()
' Select the connectors
scpi.Execute("sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" "")
MsgBox("Connectors defined for Ports 1 to 4")
' Select the Cal Kit for each port being calibrated.
scpi.Execute("sens:corr:coll:guid:ckit:port1 ""85052D"" "")
' To use an ECal module instead, comment out the above four lines
' and uncomment these four lines and use the part number printed
' on your module (which in our case was N4431-60003), followed
' by the word 'ECal'. Your ECal module must already be connected
' via USB to the PNA.
MsgBox("Cal kits defined for Ports 1 to 4")
' Initiate the calibration and query the number of steps
numSteps = GenerateSteps()
' If your selected cal kit is not a 4-port ECal module which can
' mate to all 4 ports at once, then you may want to choose which
' thru connections to measure for the cal. You must measure at
' least 3 different thru paths for a 4-port cal (for greatest
' accuracy you can choose to measure a thru connection for all 6
' pairings of the 4 ports). If you omit this command, the default
' is to measure from port 1 to port 2, port 1 to port 3, and
' port 1 to port 4. For this example we select to measure
' from port 1 to port 2, port 2 to port 3, and port 2 to port 4.
'sci.Execute("sens:corr:coll:guid:thru:ports 1,2,2,3,2,4")
' Re-generate the connection steps to account for the thru changes
numSteps = GenerateSteps()
' Measure the standards, compute and apply the cal
MeasureAndComplete(numSteps)
End Sub

Function GenerateSteps()
' Initiate the calibration and query the number of steps
scpi.Execute("sens:corr:coll:guid:init")
GenerateSteps =  scpi.Execute("sens:corr:coll:guid:steps?")
End Function
Sub MeasureAndComplete(numSteps)
MsgBox("Number of steps is " + CStr(numSteps))
' Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
MsgBox strPrompt, vbOKOnly, step
scpi.Execute("sens:corr:coll:guid:acq STAN" + CStr(i))
Next
' Conclude the calibration
MsgBox("Cal is done!")
End Sub
Perform a Guided Calibration using SCPI

This VBScript program performs a Guided Calibration using ECal or Mechanical standards. This example includes optional ECal orientation features.

- This example has been updated to include the setting of Unknown Thru or Adapter Removal adapter delay. (March 2006).
- This example has been updated to show the activation of a channel to be calibrated. (Aug. 2006).

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Guided.vbs. [Learn how to setup and run the macro.]

```
' Performing a Guided 2-port cal (Ports 1 and 2)
TwoPortGuidedCal
Sub TwoPortGuidedCal
Dim app
Dim scpi
Dim connList
Dim selectedConn1, selectedConn2
Dim kitList
Dim selectedKit
Dim message
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' The following demonstrates that the Active Channel is cal'd
' Preset the PNA
scpi.Execute "SYST:PRES"
' Create a new measurement on Chan 2
' Now there are two windows, channels and measurements
' This becomes the Active Measurement
scpi.Execute ("DISPlay:WINDow2:STATE ON")
' Define a measurement name, parameter
scpi.Execute ("CALCulate2:PARameter:DEFine 'MyMeas',S21")
"FEED" the measurement
scpi.Execute ("DISPlay:WINDow2:TRACe1:FEED 'MyMeas'")
```
' This is the Active Measurement
' Activate the 'Preset' measurement to cal chan 1
scpi.Execute ("CALC1:PAR:SEL 'CH1_S11_1'")

' Query the list of connectors that the PNA system recognizes
' Format the list with linefeed characters in place of the commas
connList = FormatList(connList)
message = "Enter your DUT connector for Port 1. Choose from this list:"
message = message & Chr(10) & Chr(10) & connList
' Select the connector for Port 1
selectedConn1 = InputBox(message)
If selectedConn1 = "" Then Exit Sub
scpi.Execute "sens:corr:coll:guid:conn:port1 " & selectedConn1 & ""
message = "Enter your DUT connector for Port 2. Again, choose from this list:"
message = message & Chr(10) & Chr(10) & connList
' Select the connector for Port 2
selectedConn2 = InputBox(message)
If selectedConn2 = "" Then Exit Sub
scpi.Execute "sens:corr:coll:guid:conn:port2 " & selectedConn2 & ""
' Note: If your PNA has more than 2 ports, then uncomment
' one or both of these next two lines.
'scpi.Execute "sens:corr:coll:guid:conn:port3 "'Not used" "
'scpi.Execute "sens:corr:coll:guid:conn:port4 "'Not used" "

' This next block of commented code demonstrates how to specify an adapter
' and its electrical delay, in situations where you are performing an
' Unknown Thru or Adapter Removal calibration. In most situations, the
' PNA is able to correctly determine an adapter's electrical length
' at the end of the calibration. However, there are scenarios where
' the PNA cannot correctly calculate the length -- such as when the channel
' has a relatively small number of measurement points (for example, 201 or less)
' and the adapter is significantly long (for example, a cable that is several feet).
' In these cases, the ADAP commands (below) enable you to explicitly specify
' the adapter you are using.
' Send these commands prior to the "sens:corr:coll:guid:init" command.
Create adapter and return the adapter number

adapterNum = scpi.Execute("sens:corr:coll:guid:adap:cre?" & selectedConn1 & "," & selectedConn2 & ")

The adapterNum string contains a '+' character.

Here we convert to integer to remove that.

adapterNum = CStr(CInt(adapterNum))

Specify that this adapter has 10 nanoseconds electrical delay (coaxial).

scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":del 10E-9"

Text description of adapter

scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":desc 'My adapter'"

Select to use this adapter specifically between ports 1 and 2

scpi.Execute "sens:corr:coll:guid:adap" & adapterNum & ":path 1,2"

End of adapter block

Query the list of acceptable cal kits and ECal module characterizations for Port 1.


Format the list with linefeed characters in place of the commas

kitList = FormatList(kitList)

message = "Enter your cal kit or ECal module characterization for Port 1.  "
message = message & "Choose from this list:"
message = message & Chr(10) & Chr(10) & kitList

Select the Cal Kit or ECal module characterization to use for Port 1.

selectedKit = InputBox(message)

If selectedKit = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:ckit:port1" & selectedKit & ";"

Query the list of acceptable cal kits and ECal module characterizations for Port 2.


Format the list with linefeed characters in place of the commas

kitList = FormatList(kitList)

message = "Enter your cal kit or ECal module characterization for Port 2.  "
message = message & "Choose from this list:"
message = message & Chr(10) & Chr(10) & kitList

Select the Cal Kit or ECal module
' characterization to use for Port 2.

selectedKit = InputBox(message)
If selectedKit = "" Then Exit Sub

scpi.Execute "sens:corr:coll:guid:ckit:port2 " & selectedKit & ""

' This next block of commented code
' shows optional functions when using ECal.
' Send these "sens:corr:pref" commands prior to the
' Read ECAL information from ECal module #1 on the USB bus
' about the Agilent factory characterization data
'module1Info = scpi.Execute("sens:corr:coll:ckit:inf? ECAL1,CHAR0")
'MsgBox "Description of ECal Module #1:" & Chr(10) & Chr(10) & module1Info

' The following command enables auto orientation of
' the ECal module (The PNA senses which port of the
' module is connected to which port of the PNA).
'scpi.Execute "sens:corr:pref:ecal:ori ON"
' However, if you are measuring at very low power levels where
' the PNA may fail to sense the module's orientation, then turn auto
' orientation OFF and specify how the module is connected.
' "A1,B2" indicates Port A of the module is connected
' to PNA Port 1 and Port B is connected to PNA Port 2.
'scpi.Execute "sens:corr:pref:ecal:ori OFF"
' End of optional ECal setup

' Select the thru method of "Default". This instructs the PNA to
' determine which thru standard measurement technique to use
' based upon the selected connectors and
' calibration kit(s) and the PNA model number.
' with new CMET and TMET 'default' is set by not sending the commands
'
' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numSteps)
' Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next
' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"
End Sub

Function FormatList(list)
Dim tokens
' Strip the leading and trailing quotation
' marks from the list string
list = Mid(list, 2, Len(list) - 3)
' Tokenize the comma-delimited list string
' into an array of the individual substrings
tokens = Split(list, ",")
' Rebuild the list string, placing linefeed
' characters where the commas were,
' using Trim to remove leading and trailing spaces.
list = ""
For i = 0 To UBound(tokens)
tokens(i) = Trim(tokens(i))
list = list & tokens(i) & Chr(9)
If i < UBound(tokens) Then
    i = i + 1
    tokens(i) = Trim(tokens(i))
    list = list & tokens(i) & Chr(10)
End If
Next
FormatList = list
End Function
Last Modified:

14-May-2007    MX Updated for new CMET and TMET commands
Perform Guided ECal using SCPI

This VBScript program performs a Guided ECal Calibration. While this example is good to use as a starting point for Guided ECal, the [Guided comprehensive cal example](#) has some advanced features that are not in this program.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Guided.vbs. [Learn how to setup and run the macro](#).

```vbscript
' Performing a 2-port cal (Ports 1 and 2)
Dim app
Dim scpi

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Specify the DUT connectors
' (for each connector of your DUT, one of the ECal module's ports must have
' that same connector, or else you cannot achieve the cal using that module).
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female""
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male""

' Note: If your PNA has more than 2 ports, you would need to uncomment
' one or both of these next two lines, to explicitly specify this is
' just a 2-port cal.
'scpi.Execute "sens:corr:coll:guid:conn:port3 ""Not used""
'scpi.Execute "sens:corr:coll:guid:conn:port4 ""Not used"
MsgBox "Connectors defined for Ports 1 and 2"

' Select the ECal module for each port being calibrated.
' Replace N4691-60004 with the part number printed on your module
' followed by User x if using a User characterization, then the word 'ECal'
' Your ECal module must already be connected via USB to the PNA.
scpi.Execute "sens:corr:coll:guid:ckit:port1 ""N4691-60004 ECal"
MsgBox "Cal kits defined for Ports 1 and 2"

' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards
For i = 1 To numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
```
' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"
Perform Guided Mechanical Cal using SCPI

This VBScript program performs a Guided Calibration using Mechanical standards. While this example is good to use as a starting point for guided mechanical cal, the Guided comprehensive cal example has some advanced features that are not in this program.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Guided.vbs. Learn how to setup and run the macro.

' Performing a 2-port cal (Ports 1 and 2)

Dim app
Dim scpi

' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser

' Specify the DUT connectors
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" "
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male"" "

' Note: If your PNA has more than 2 ports, you would need to uncomment
' one or both of these next two lines, to explicitly specify this is
' just a 2-port cal.
scpi.Execute "sens:corr:coll:guid:conn:port3 ""Not used"" "
scpi.Execute "sens:corr:coll:guid:conn:port4 ""Not used"" "
MsgBox "Connectors defined for Ports 1 and 2"

' Select the Cal Kit for each port being calibrated.
scpi.Execute "sens:corr:coll:guid:ckit:port1 ""85052D"" "
scpi.Execute "sens:corr:coll:guid:ckit:port2 ""85052D"" "
MsgBox "Cal kits defined for Ports 1 and 2"

' Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numSteps)

' Measure the standards
' The following series of commands shows that standards
can be measured in any order. These steps acquire
'measurement of standards in reverse order.
' It is easiest to iterate through standards using
'a For-Next Loop.
For i = numSteps To 1
    step = "Step " + CStr(i) + " of " + CStr(numSteps)
    MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next

' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"

Last Modified:

20-Jan-2007  Added note about any order for steps.
Perform a Guided 1-Port Cal on Port 2

This VBScript program does the following:

1. Clear measurements from the PNA
2. Create a new S22 measurement
3. Set an instrument state
4. Select the connector types
5. Select a cal kit
6. Initiate a Guided calibration
7. Display a prompt to connect each standard
8. Save the calibration to a newly created cal set

**Note:** This example illustrates an important step when calibrating a reflection measurement in the reverse direction. You MUST create a reverse (S22) measurement and have it be the active (selected) measurement on the channel that is being calibrated. This is not necessary for any calibrating any other measurement parameter.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Guided.vbs. **Learn how to setup and run the macro.**

```vbs
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
App.Preset

Dim step
Dim Parser
Dim prompt
Dim txtDat
Dim Chan

' Rem Clear old measurements
App.Reset

' Rem Create a new Measurement
Set Parser = App.SCPIStringParser
Parser.Parse "DISPlay:WINDow1:STATE ON"
Parser.Parse "CALCulate:PARameter:DEFine 'MyMeas',S22"
Parser.Parse "DISPlay:WINDow1:TRACe1:FEED 'MyMeas'"

' Rem Initialize state
Set Chan = App.ActiveChannel
Chan.StartFrequency = 200e6
```
Chan.StopFrequency = 1.5e9
Chan.IFBandwidth = 1000
step = 3

Rem Begin a guided calibration
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'Not used''"
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'Type N (50) male''"
Parser.Parse "SENS:CORR:COLL:GUID:INIT"

Rem Query the number of steps

Rem Display the number of steps
MsgBox("Number of steps is " + txtDat)

Rem Set the loop counter limit
step = txtDat

Rem Measure the standards
For i = 1 To step
  If i = 1 Then
    MsgBox(prompt)
    Parser.Parse ("sens:corr:coll:guid:acq STAN1")
  ElseIf i = 2 then
    MsgBox(prompt)
    Parser.Parse ("sens:corr:coll:guid:acq STAN2")
  ElseIf i = 3 then
    MsgBox(prompt)
    Parser.Parse ("sens:corr:coll:guid:acq STAN3")
  End If
Next

Rem All standards have been measured. Save the result
Parser.Parse "SENS:CORR:COLL:GUID:SAVE"
MsgBox("The calibration has been completed")
Perform Guided TRL Calibration

This VBScript file performs a 2-Port Guided TRL calibration on 2-port PNA analyzers. (See example of TRL cal on a 4-port PNA.) This program does the following:

- Clear old measurements from the PNA
- Create a new S22 measurement
- Set an instrument state
- Select the connectors and cal kit
- Initiate a Guided calibration
- Display a prompt as each new standard must be connected
- Save the calibration to a newly created cal set.

Note: This program runs without error on all PNA code revisions 7.21 and higher.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as TRL.vbs. Learn how to setup and run the macro.

```vbs
Dim App
Dim Parser
Dim Chan
Dim txtDat
Dim step
Dim parserTxt
Dim prompt
Set App = CreateObject("AgilentPNA835x.Application")
' Clear old measurements
App.Reset
' Create a new Measurement
Set Parser = App.SCPIStringParser
Parser.Parse "DISPlay:WINDow1:STATE ON"
Parser.Parse "CALCulate:PARameter:DEFine 'MyMeas',S12"
Parser.Parse "DISPlay:WINDow1:TRACe1:FEED 'MyMeas'"
' Initialize state
Set Chan = App.ActiveChannel
```
Chan.StartFrequency = 18.0e9
Chan.StopFrequency = 20.0e9
Chan.IFBandwidth = 1000
' Begin a guided calibrations
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 male'"
Parser.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 female'"
' Select TRL cal method.
Parser.Parse "SENS:CORR:COLL:GUID:PATH:CMET 1,2,'TRL'"

txtDat = Parser.Parse("SENS:CORR:COLL:GUID:PATH:CMET? 1,2")
MsgBox("Method " + txtDat)
Parser.Parse "SENS:CORR:COLL:GUID:INIT"
' Query the number of steps
' Display the number of steps
MsgBox("Number of steps is " + txtDat)
' Set the loop counter limit
step = CInt(txtDat)
' Measure the standards
For i = 1 To step
  parserTxt = "sens:corr:coll:guid:desc? " + CStr(i)
  prompt = Parser.Parse(parserTxt)
  MsgBox(prompt)
  parserTxt = "sens:corr:coll:guid:acq STAN" + CStr(i)
  Parser.Parse (parserTxt)
Next
' All standards have been measured.  Save the result
Parser.Parse "SENS:CORR:COLL:GUID:SAVE"
MsgBox("The TRL calibration has been completed")
Perform Unknown Thru or TRL Cal

The following program performs either a 2-port SOLT Unknown Thru Cal or a 2-port TRL Cal. The 85052C Cal Kit used in this program contains both types of standards. This program can be run on 2-port or 4-port PNAs. When run on a multiport (4 or more ports) PNA, which does not have a reference receiver per port, a Delta Match Cal is required.

See example of Delta Match Cal.

See the Guided Cal commands

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unknown.vbs. Learn how to setup and run the macro.

```
Sub PerformUnknownThruOrTRLCal()
    Set pna = CreateObject("AgilentPNA835x.Application")
    Set scpi = pna.ScpiStringParser
    ' Specify connectors for Ports 1 and 2
    scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 female''
    scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male''
    'If your PNA has 3 or 4 ports, uncomment one or both of
    'these next two lines, to explicitly specify this is a 2-port cal.
    'scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT3 'Not used''
    'scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT4 'Not used''
    ' Specify cal kit for Ports 1 and 2
    scpi.Parse "SENS:CORR:COLL:GUID:CKIT:PORT1 '85052C''
    ' Since the 85052C cal kit contains SOLT standards and also TRL
    ' standards, these next two lines set cal and thru method
    scpi.Execute "SENS:CORR:COLL:GUID:PATH:CMEThod 1,2,"SOLT"
    scpi.Execute "SENS:CORR:COLL:GUID:PATH:TMEThod 1,2,"UNKN"
    ' To set up the cal as TRL, comment the previous line and uncomment
    ' this next line. The TMEThod is set by default
    'scpi.Execute "SENS:CORR:COLL:GUID:PATH:CMEThod 1,2,"TRL"
    ' Initiate the calibration
    scpi.Parse "SENS:CORR:COLL:GUID:INIT"
    ' Query the list of ports that need delta match
    portList = Split(retStr, ",")
    ' If portList contains just one element and it's value is 0, then that indicates
    ' none of the ports being calibrated require delta match data.
    ' Note: if each testport on the PNA has it's own reference receiver (R channel),
    ' then delta match is never needed, so portList will always be just 0.
    lowerBound = LBound(portList)
    If (UBound(portList) <> lowerBound) Or (CInt( portList(lowerBound) ) <> 0) Then
```
' Delta match data is required for at least one port.
' For this example, we assume a Global Delta Match Cal has previously been
' performed so the Global Delta Match CalSet exists.
' The Global Delta Match CalSet is used when the APPL command is invoked
' without a specific calset ID (GUID).
    scpi.Parse "SENS:CORR:COLL:GUID:DMAT:APPL"
End If

' Query the number of calibration steps
retStr = scpi.Parse("SENS:CORR:COLL:GUID:STEP?")
numSteps = CInt(retStr)

' Measure the cal standards
For i = 1 To numSteps
    retVal = MsgBox(prompt, vbOKCancel)
    If retVal = vbCancel Then Exit Sub
    retStr = scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i) & ";*OPC?")
Next

' Compute the error coefficients and save the cal to CalSet, and turn it on
scpi.Parse "SENS:CORR:COLL:GUID:SAVE"
MsgBox "Cal is done!"
End Sub

Last Modified:

14-May-2007   MX Updated for new CMET and TMET commands
Perform Global Delta Match Cal

The following program performs a Global Delta Match Calibration. This is required when performing an Unknown Thru Cal or TRL Cal on PNAs without a reference receiver for each test port. See example of Unknown Thru or TRL Cal.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Delta.vbs. Learn how to setup and run the macro.

```
Sub PerformGlobalDeltaMatchCal()
    Set pna = CreateObject("AgilentPNA835x.Application")
    Set scpi = pna.ScpiStringParser

    ' Initiate a Global Delta Match calibration, choosing connector and cal kit
    scpi.Parse "SENS:CORR:COLL:GUID:DMAT 'APC 3.5 female', '85033D/E'"

    ' Query the number of calibration steps
    retStr = scpi.Parse("SENS:CORR:COLL:GUID:STEP?")
    numSteps = CInt(retStr)

    ' Measure the cal standards
    For i = 1 To numSteps
        retVal = MsgBox(prompt, vbOKCancel)
        If retVal = vbCancel Then Exit Sub
        retStr = scpi.Parse("SENS:CORR:COLL:GUID:ACQ STAN" & CStr(i) & ";*OPC?")
    Next

    ' Compute the error coefficients and save the cal to Global Delta Match CalSet
    scpi.Parse "SENS:CORR:COLL:GUID:SAVE"
    MsgBox "Cal is done!"
End Sub
```
Perform an Unguided ECal

This VBScript program performs an Unguided Full 2-Port ECal.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unguided.vbs. Learn how to setup and run the macro.

```vbs
Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser
' Preset the analyzer
scpi.Execute "SYSTem:PRESet"

' Start frequency of 10 MHz
scpi.Execute "SENSe:FREQuency:STARt 10E6"

' Stop frequency of 9 GHz
scpi.Execute "SENSe:FREQuency:STOP 9E9"

' Select the preset S11 measurement
scpi.Execute "CALCulate:PARameter:SELect 'CH1_S11_1'"
' Read the information about the Agilent factory characterization data of ECal module #1 on the USB bus
module1Info = scpi.Execute("SENSe:CORRection:COLLect:CKIT:INFormation? ECAL1,CHAR0")

' Prompt for the ECal module
MsgBox "Description of ECal Module #1:" & Chr(10) & Chr(10) & module1Info & _Chr(10)
' ECal full 1 port and 2 port
scpi.Execute "SENSe:CORRection:COLLect:METHod refl3"
scpi.Execute "SENSe:CORRection:COLLect:METHod SPARSOLT"
' Specify to have the PNA automatically determine which port of the ECal module is connected to which port of the PNA.
scpi.Execute "SENSe:CORRection:PREFerence:ECAL:ORIentation ON"
' Alternatively, if you are measuring at very low power levels where the PNA fails to sense the module's orientation, you may need to turn off the auto orientation and specify how the module is connected (as in these next two commented lines of code -- "A1,B2" would indicate Port A of the module is connected to Port 1 and Port B is connected to Port 2).
'scpi.Execute "SENSe:CORRection:PREFerence:ECAL:ORIentation OFF"
'scpi.Execute "SENSe:CORRection:PREFerence:ECAL:PMAP ECAL1,'A1,B2'"
' Acquire and store the calibration terms. *OPC? causes a "+1" to be returned when finished. CHAR0 indicates to use the Agilent factory characterized data within the ECal module (as opposed to a user characterization).
x = scpi.Execute("SENSe:CORRection:COLLect:ACQuire ECAL1,CHAR0;*OPC?"
MsgBox "Done with calibration."
```
Perform an Unguided 2-Port Mechanical Cal

This VBScript program performs an Unguided, Full 2-Port, calibration using ONE set of mechanical calibration standards.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unguided.vbs. Learn how to setup and run the macro.

Set App = CreateObject("AgilentPNA835x.Application")
Set Scpi = App.SCPIStringParser

'Start state
Scpi.Execute ("SYSTem:PRESet")

'Select the Preset measurement
Scpi.Execute ("CALCulate:PARameter:SELect \"CH1_S11_1\"")

'Set the calibration method
Scpi.Execute ("SENSe:CORRection:COLLect:METHod SPARSOLT")

'Select a cal kit
Scpi.Execute ("SENSe:CORRection:COLLect:CKIT:SELection 1")

'Set one set of standards
Scpi.Execute ("SENSe:CORRection:TSTandards OFF")

'Set acquisition to FORWARD
Scpi.Execute ("SENSe:CORRection:SFORward ON")

'Measure the standards in forward direction
MsgBox "Connect OPEN to Port 1; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan1")

MsgBox "Connect SHORT to Port 1; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan2")

MsgBox "Connect LOAD to Port 1; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan3")

'Set acquisition to REVERSE
Scpi.Execute ("SENSe:CORRection:SFORward OFF")

'Measure the standards in reverse direction
MsgBox "Connect OPEN to Port 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan1")

MsgBox "Connect SHORT to Port 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan2")
MsgBox "Connect LOAD to Port 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan3")

'Measure the thru standard
MsgBox "Connect THRU between Ports 1 and 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan4")

'OPTIONAL Measure Isolation
MsgBox "Connect LOADS to Port 1 AND Port 2; then press OK"
Scpi.Execute ("SENSe:CORRection:COLLect:ACQuire stan5")

'All standards have been measured. Save the result
Scpi.Execute ("SENS:CORR:COLL:SAVE")
MsgBox "The calibration has been completed"
Perform an Unguided 1-Port Cal on Port 2

This VBScript program does the following:

1. Clear measurements from the PNA
2. Create a new S22 measurement
3. Set an instrument state
4. Select a cal kit
5. Initiate an Unguided calibration
6. Display a prompt to connect each standard
7. Save the calibration to a newly created cal set

**Note:** This example illustrates an important step when calibrating a reflection measurement in the reverse direction. You MUST create a reverse (S22) measurement and have it be the active (selected) measurement on the channel that is being calibrated. This is not necessary for any calibrating any other measurement parameter.

The SCPI commands in this example are sent over a COM interface using the `SCPIStringParser` object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unguided.vbs. [Learn how to setup and run the macro.](#)

```vbnet
Dim App
Set App = CreateObject("AgilentPNA835x.Application")
App.Preset

Dim Parser
Dim Chan

' Rem Clear old measurements
App.Reset

' Rem Create a new Measurement
Set Parser = App.SCPIStringParser
Parser.Parse "DISPLAY:WINDOW1:STATE ON"
Parser.Parse "CALCulate:PARameter:DEFine 'MyMeas',S22"
Parser.Parse "DISPLAY:WINDOW1:TRACe1:FEED 'MyMeas'"

' Rem Initialize state
Set Chan = App.ActiveChannel
Chan.StartFrequency = 200e6
Chan.StopFrequency = 1.5e9
Chan.IFBandwidth = 1000

' Rem Begin an unguided calibration
```
Rem Set the calibration method
Parser.Parse "SENSe:CORRection:COLLect:METHod REFL3"

Rem Turn off continuous sweep
Parser.Parse "INITiate:CONTinuous OFF"

Rem Select a cal kit
Parser.Parse "SENSe:CORRection:COLLect:CKIT:SELect 1"

Rem Measure the standards
MsgBox("Connect OPEN to port 2. Then press OK")
Parser.Parse ("sens:corr:coll:acq STAN1")

MsgBox("Connect SHORT to port 2. Then press OK")
Parser.Parse ("sens:corr:coll:acq STAN2")

MsgBox("Connect LOAD to port 2. Then press OK")
Parser.Parse ("sens:corr:coll:acq STAN3")

Rem All standards have been measured. Save the result
Parser.Parse "SENS:CORR:COLL:SAVE"

Rem Turn ON continuous sweep
Parser.Parse "INITiate:CONTinuous ON"
MsgBox("The calibration has been completed")
Perform an Unguided Cal on a 4-Port PNA

This topic describes how to perform an unguided calibration on a multiport network analyzer using SCPI. The objective here is to make clear the relationship between the physical port on which a standard is being measured, the actual device in the cal kit, and the SCPI command used to acquire the device.

There are two sets of SCPI commands that acquire calibrations. One set is used for guided cal, the other for unguided. The SCPI commands that provide remote access to unguided cal are in the SENS:CORR:COLL block:

- SENS:CORR:COLL:METHod
- SENS:CORR:COLL:ACQuire
- SENS:CORR:COLL:SAVE

On a four port network analyzer, the remote programmer needs to be aware of the relationship between the physical port and the calibration kit class assignments. The example program (below) illustrates the usage by performing three unique 2 port cals, taking care to acquire the appropriate standards.

Calibration standards classes are ‘categories’ of standard types. To perform a 2 port calibration, the cal wizard requires the user to measure:

3 reflection standards on the forward port:

- Class S11A typically an open
- Class S11B typically a short
- Class S11C typically a load

Likewise, 3 reflection standards are required for the reverse port:

- Class S22A typically an open
- Class S22B typically a short
- Class S22C typically a load

There is also a transmission standard that is measured in both directions:

- Class S21T typically a thru

The following illustrates the relationship between cal kit physical standards and calibration classes.

Here is a list of the physical devices in my calibration kit.
Standard #1 = "3.5 mm male short"
Standard #2 = "3.5 mm male open"
Standard #3 = "3.5 mm male broadband load"
Standard #4 = "Insertable thru standard"
When you perform a calibration remotely using SCPI, you don’t specify the device number directly. Rather, you specify the class you want to measure. Each device in the calibration kit is assigned to a class. And since more than one device can be assigned to the same class, each class contains an ordered list of devices. The class assignments are user-settable using the Advanced Modify Cal Kit dialog or the SCPI command:

**SENS:CORR:COLL:CKIT:ORDER**<class>, <std>, <std>, <std>, <std>,<std>,<std>,<std>

The 85052B kit used in the example program had the following standard list for each class: The list was obtained by issuing the corresponding SCPI query:

**SENS:CORR:COLL:CKIT:OLIST1?**  
S11A = +2,+15,+0,+0,+0,+0,+0  
S11B = +1,+7,+0,+0,+0,+0,+0  
S11C = +6,+5,+3,+12,+13,+14,+0  
S21T = +4,+8,+0,+0,+0,+0,+0  
S22A = +2,+15,+0,+0,+0,+0,+0  
S22B = +1,+7,+0,+0,+0,+0,+0  
S22C = +6,+5,+3,+12,+13,+14,+0  
S12T = +4,+8,+0,+0,+0,+0,+0  

When you perform the calibration, you acquire data by issuing the ACQuire command:

**SENS:CORR:COLL:ACQ**<class>, <<substd> ]

For example:

**SENS:CORR:COLL:SFOR** 1  
SENS:CORR:COLL:ACQ STANA, SST2

The SFOR command tells the wizard to make the next acquisition in the forward direction. The ACQuire command specifies that we are measuring the 2nd device in the list for STANA. And since we are measuring SFORward, STANA refers to class #1 or S11A. The list of devices for this class are specified in the OLIST1 query above. The associations are shown in red.

Alternately, you could modify the device order for the S11A class to move device #15 into the first position (SENS:CORR:COLL:CKIT:ORDER1). When the desired device is in the first position, you needn’t specify the order number in the ACQuire command. The default is the first device in the OLIST. This worked well for two port network analyzers where the order for S11A,B,C classes were setup for port 1 and the order for S22A,B,C was set up for port 2. With the kit setup in the proper order, you could eliminate the specification of the substandard
When performing 2 port calibrations on 4 Port Network Analyzers (e.g.: PNA Model N5230A), the wizard applies S11A,B,C standards to the lower numbered port, S22A,B,C standards to the higher numbered port. Since the two classes (S11A,B,C and S22A,B,C) are applied to multiple ports, the programmer must take into account the ports being measured and take greater care when specifying the ACQuire command to ensure that the correct device is being measured.

<table>
<thead>
<tr>
<th>Ports</th>
<th>S11A Port</th>
<th>S22A Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1,3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1,4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>2,3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2,4</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3,4</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

The following example program shows one method of handling two port cals on a multiport network analyzer. The connectors at the measurement plane are assumed to be (1) male, (2) female, (3) male, and (4) male. In the example, three cals are performed: 1-2 (insertable male to female), 2-3 (insertable female to male), and 3-4 (noninsertable using an characterized adapter).

```vba
option explicit
public scpi
public pna
' assume a 4 port PNA with the following connectors:
' the standard measured on these ports will be the opposite gender
' PORT 1 = 3.5 male
' PORT 2 = 3.5 female
' PORT 3 = 3.5 male
' PORT 4 = 3.5 male
To perform 2 port calibrations between 1-2, 2-3, and 3-4 you need to do the following

call main

sub main
set pna = CreateObject("AgilentPnA835x.Application")
set scpi = pna.ScpiStringParser
pna.Preset
' select a kit to use for this demonstration
' kit #1 for the N5230A is the 85052B 3.5mm kit with sliding load
scpi.execute("SENS:CORR:COLL:CKIT:SELECT 1")
```
' ---------------------------------------------
'   CALIBRATE PORTS 1 and 2, insertable cal
' ---------------------------------------------

wscript.echo
wscript.echo "Calibrating ports 1 and 2"
scpi.execute("SYST:PRES;")
scpi.execute("calc:par:sel CH1_S11_1")
scpi.execute("SENS:CORR:TST:STATE 0")
scpi.execute("SENS:CORR:COLL:METHod SPARSOLT")
scpi.execute("SENS:CORR:SFOR 1")
MeasureFemaleStandards 1
scpi.execute("SENS:CORR:SFOR 0")
MeasureMaleStandards 2
MeasureTransmissionStandards 1,2
scpi.execute("SENS:CORR:COLL:SAVE")

' ---------------------------------------------
'   CALIBRATE PORTS 2 and 3, insertable cal
' ---------------------------------------------

wscript.echo
wscript.echo "Calibrating ports 2 and 3"
scpi.execute("SYST:PRES;")
scpi.execute("calc:par:sel CH1_S11_1")
scpi.execute("calc:par:mod S23")
scpi.execute("SENS:CORR:TST:STATE 0")
scpi.execute("SENS:CORR:COLL:METHod SPARSOLT")
scpi.execute("SENS:CORR:SFOR 1")
MeasureMaleStandards 2
scpi.execute("SENS:CORR:SFOR 0")
MeasureFemaleStandards 3
MeasureTransmissionStandards 2,3
scpi.execute("SENS:CORR:COLL:SAVE")

' ---------------------------------------------
'   CALIBRATE PORTS 3 and 4, non-insertable cal
' ---------------------------------------------

wscript.echo
wscript.echo "Calibrating ports 3 and 4"
scpi.execute("SYST:PRES;")
scpi.execute("calc:par:sel CH1_S11_1")
scpi.execute("calc:par:mod S43")
scpi.execute("SENS:CORR:COLL:METHod SPARSOLT")
scpi.execute("SENS:CORR:SFOR 1")
MeasureFemaleStandards 3
scpi.execute("SENS:CORR:SFOR 0")
MeasureFemaleStandards 4
MeasureAdapter 3, 4
scpi.execute("SENS:CORR:COLL:SAVE")
end sub
sub MeasureMaleStandards ( portNumber )
dim portstr
portstr = formatnumber(portNumber, 0)
Promptconnect1 1, 1, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN1;*OPC?")

Promptconnect1 2, 1, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN2;*OPC?")
Promptconnect1 3, 3, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN3,SST3;*OPC?")
end sub

sub MeasureFemaleStandards (portNumber)
dim portstr
portstr = formatnumber(portNumber, 0)
Promptconnect1 1, 2, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN1,SST2;*OPC?")
Promptconnect1 2, 2, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN2,SST2;*OPC?")
Promptconnect1 3, 6, portNumber
scpi.execute("SENS:CORR:COLL:ACQ STAN3,SST6;*OPC?")
end sub

sub MeasureTransmissionStandards( port1, port2)
dim p1str
dim p2str
p1str = formatnumber( port1, 0)
p2str = formatnumber( port2, 0)

Promptconnect2 4, 1, port1, port2
scpi.execute("SENS:CORR:COLL:ACQ STAN4;*OPC?")
end sub

sub MeasureAdapter( port1, port2)
dim p1str
dim p2str
p1str = formatnumber( port1, 0)
p2str = formatnumber( port2, 0)

Promptconnect2 4, 2, port1, port2
scpi.execute("SENS:CORR:COLL:ACQ STAN4,SST2;*OPC?")
end sub

' return the nth item in the comma separated list
Function GetItemNumber( list, n)
dim strVector
strVector = split(list,"",-1,1)
GetItemNumber = strVector(n-1)
end function

' remove the trailing newline from str
function chop( str )
```vba
' remove the appended newline
dim pos
pos = InStrRev(tmp,vblf)
if (pos >0) then
tmp = mid(tmp,1,pos-1)
end if
chop = tmp
end function

' return the label for the nth standard assigned to the class described by
' class_index.
' if class_index = 1, class is S11A (STAN1)
' if class_index = 2, class is S11B (STAN2), etc
function GetStandardLabel( class_index, nth)
dim olist
dim stdnum
dim resp
olist = scpi.execute("SENS:CORR:COLL:CKIT:OLIST" + formatnumber(class_index,0)+"?")
stdnum = GetItemNumber( olist, nth)
scpi.execute("SENS:CORR:COLL:CKIT:STAN " + formatnumber(stdnum,0))
resp = scpi.execute("SENS:CORR:COLL:CKIT:STAN:LABel?")
GetStandardLabel = chop(resp)
end function

sub PromptConnect1( class_index, nth, port)
wscript.echo "CONNECT " + GetStandardLabel( class_index, nth) + " to port " + formatnumber(port,0)
end sub

sub PromptConnect2( class_index, nth, port1, port2)
wscript.echo "CONNECT " + GetStandardLabel( class_index, nth) + " between ports " + formatnumber(port1,0) + " and " + formatnumber(port2,0)
end sub

' Print the order of standards per class for this kit
sub PrintKitOlist( kit )
dim i
dim cmd
dim resp
wscript.echo
wscript.echo olistcmd
olistcmd = "SENS:CORR:COLL:CKIT:OLIST"
' list the sub standards for each of the following classes
' S11A, S11B, S11C, FWD TRANS, FWD ISOL, S22A, S22B, S22C, REV TRANS, REV ISOL
for i = 1 to 8
    cmd = olistcmd + formatNumber(i,0) + "?"
    resp = scpi.execute(cmd)
wscript.echo cmd + "= " + chop(resp)
next
end sub
```
sub PrintKitStandardInfo( kit )
  wscript.echo scpi.execute("SENS:CORR:COLL:CKIT:NAME?")
  dim i
  for i = 1 to 30
    dim slabel
    dim snum
    snum = formatNumber(i,0)
    scpi.execute("SENS:CORR:COLL:CKIT:STAN " + snum)
    slabel=sctpi.execute("SENS:CORR:COLL:CKIT:STAN:LABel?")
    wscript.echo "Standard #"+snum+ " = " + chop(slabel)
  next
end sub

The output from this program is as follows:

Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

"85052B 3.5 mm with sliding load"
Standard #1 = "3.5 mm male short"
Standard #2 = "3.5 mm male open"
Standard #3 = "3.5 mm male broadband load"
Standard #4 = "Insertable thru standard"
Standard #5 = "3.5 mm male sliding load"
Standard #6 = "3.5 mm male lowband load"
Standard #7 = "3.5 mm female short"
Standard #8 = "female to female characterized thru adapter"
Standard #9 = "0-2 Load"
Standard #10 = "Open"
Standard #11 = "Non-insertable thru"
Standard #12 = "3.5 mm female lowband load"
Standard #13 = "3.5 mm female sliding load"
Standard #14 = "3.5 mm female broadband load"
Standard #15 = "3.5 mm female open"
Standard #16 = "Open"
Standard #17 = "Open"
Standard #18 = "Open"
Standard #19 = "Open"
Standard #20 = "Open"
Standard #21 = "Open"
Standard #22 = "Open"
Standard #23 = "Open"
Standard #24 = "Open"
Standard #25 = "Open"
Standard #26 = "Open"
Standard #27 = "Open"
Standard #28 = "Open"
Standard #29 = "Open"
Standard #30 = "Open"

SENS:CORR:COLL:CKIT:OLIST1?= +2,+15,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST2?= +1,+7,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST3?= +6,+5,+3,+12,+13,+14,+0
SENS:CORR:COLL:CKIT:OLIST4?= +4,+8,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST5?= +2,+15,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST6?= +1,+7,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST7?= +6,+5,+3,+12,+13,+14,+0
SENS:CORR:COLL:CKIT:OLIST8?= +4,+8,+0,+0,+0,+0,+0

Calibrating ports 1 and 2
CONNECT "3.5 mm female open" to port 1
CONNECT "3.5 mm female short" to port 1
CONNECT "3.5 mm female broadband load" to port 1
CONNECT "3.5 mm male open" to port 2
CONNECT "3.5 mm male short" to port 2
CONNECT "3.5 mm male broadband load" to port 2
CONNECT "Insertable thru standard" between ports 1 and 2

Calibrating ports 2 and 3
CONNECT "3.5 mm male open" to port 2
CONNECT "3.5 mm male short" to port 2
CONNECT "3.5 mm male broadband load" to port 2
CONNECT "3.5 mm female open" to port 3
CONNECT "3.5 mm female short" to port 3
CONNECT "3.5 mm female broadband load" to port 3
CONNECT "Insertable thru standard" between ports 2 and 3

Calibrating ports 3 and 4
CONNECT "3.5 mm female open" to port 3
CONNECT "3.5 mm female short" to port 3

2417
CONNECT "3.5 mm female broadband load" to port 3
CONNECT "3.5 mm female open" to port 4
CONNECT "3.5 mm female short" to port 4
CONNECT "3.5 mm female broadband load" to port 4
CONNECT "female to female characterized thru adapter" between ports 3 and 4
Perform an Unguided Cal on Multiple Channels

This VBScript program performs an Unguided Calibration simultaneously on two channels.

This could be used in the following cases:

- If you need more than the current number of data points per trace, so the additional points must be added to a different channel.
- If you need several channels with independent settings, but you want to calibrate all channels with a minimal number of standard connections. This would be especially critical for on wafer calibration.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as Unguided.vbs. Learn how to setup and run the macro.

```vbs
Dim app
Dim scpi
Dim NumberOfActiveChannels
NumberOfActiveChannels = 2
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
' Query the list of connectors that the PNA system recognizes
scpi.Execute("SYST:PRES")
'Wait for successful preset before continuing
done=scpi.Execute("*OPC?")
'The following section sets up 2 channels with different frequency ranges
scpi.Execute("DISP:WIND1:STATE OFF")
'Reset Windows
scpi.Execute("DISP:WIND1:STATE ON")
scpi.Execute("DISP:WIND2:STATE ON")
'
' Assign a measurement to the first window
scpi.Execute("CALC1:PAR:DEF 'Meas1', S21")
scpi.Execute("DISP:WIND1:TRAC1:FEED 'Meas1' ")
'Assign a measurement to the second window
scpi.Execute("CALC2:PAR:DEF 'Meas2', S21")
scpi.Execute("DISP:WIND2:TRAC1:FEED 'Meas2' ")
```
'Set up two channels with independent parameters
scpi.Execute("SENS1:FREQ:SPAN 1e9")
scpi.Execute("SENS2:FREQ:SPAN 1e6")
'Wait for changes before continuing
done=scpi.Execute("*OPC?")
'
'This section sets the calibration kits for channel 1 and channel 2
'Select a trace from channel 1 and set calibration type and cal kit
scpi.Execute("CALC1:PAR:SEL 'Meas1'")
scpi.Execute("SENS1:CORR:COLL:METH SPARSOLT")
scpi.Execute("SENS1:CORR:COLL:CKIT 2") '85056D for default settings
'Same standards for forward and reverse direction
scpi.Execute("SENS1:CORR:TST OFF")
'Select a trace from channel 2 and set calibration type and cal kit
scpi.Execute("CALC2:PAR:SEL 'Meas2'")
scpi.Execute("SENS2:CORR:COLL:METH SPARSOLT")
scpi.Execute("SENS2:CORR:COLL:CKIT 2") '85056D for default settings
'Same standards for forward and reverse direction
scpi.Execute("SENS2:CORR:TST OFF")

'Set both channels to manual triggering
scpi.Execute("INIT1:CONT OFF")
scpi.Execute("INIT2:CONT OFF")
'
'The following assumes female port connector on port 1
' and male port connector on port 1
'Step through all active channels and calibrate and measure all standards.
scpi.Execute("SENS1:CORR:SFOR ON") 'Set acquisition to forward
scpi.Execute("SENS2:CORR:SFOR ON") 'Set acquisition to forward
MsgBox("Connect OPEN standard to port 1")
For CurrentChannel  = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":'"")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan1")
done= scpi.Execute("*OPC?")
Next
MsgBox("Connect SHORT standard to port 1")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan2")
done=scpi.Execute("*OPC?")
Next

MsgBox("Connect LOAD standard to port 1")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan3")
done=scpi.Execute("*OPC?")
Next
scpi.Execute("SENS1:CORR:SFOR OFF") 'Set acquisition to reverse
scpi.Execute("SENS2:CORR:SFOR OFF") 'Set acquisition to forward

MsgBox("Connect OPEN standard to port 2")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan1")
done=scpi.Execute("*OPC?")
Next

MsgBox("Connect SHORT standard to port 2")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan2")
done=scpi.Execute("*OPC?")
Next

MsgBox("Connect LOAD standard to port 2")
For CurrentChannel = 1 To NumberOfActiveChannels
scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":'")
scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan3")
done=scpi.Execute("*OPC?")
Next
'Measure thru standard for all channels in both forward and reverse direction
MsgBox("Connect THRU between ports 1 and 2")
scpi.Execute("SENS1:CORR:SFOR ON") 'Set acquisition to forward
scpi.Execute("SENS2:CORR:SFOR ON") 'Set acquisition to forward
For CurrentChannel = 1 To NumberOfActiveChannels
    scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":'"")
    scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan4")
    done=scpi.Execute("*OPC?")
Next
scpi.Execute("SENS1:CORR:SFOR OFF") 'Set acquisition to reverse
scpi.Execute("SENS2:CORR:SFOR OFF") 'Set acquisition to reverse
For CurrentChannel = 1 To NumberOfActiveChannels
    scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":'"")
    scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL stan4")
    done=scpi.Execute("*OPC?")
Next
For CurrentChannel = 1 To NumberOfActiveChannels
    scpi.Execute("CALC" & CurrentChannel & ":PAR:SEL 'Meas" & CurrentChannel & ":'"")
    scpi.Execute("SENS" & CurrentChannel & ":CORR:COLL:SAVE")
    done=scpi.Execute("*OPC?")
Next
'Set both channels to continuous triggering
scpi.Execute("INIT1:CONT ON")
scpi.Execute("INIT2:CONT ON")
ECALConfidence Check using SCPI

This Visual Basic program performs a complete ECAL confidence check.

To run this program, you need:

- An established GPIB interface connection
- Agilent's VISA or National Instrument's VISA installed on your PC
- The module visa32.bas added to your VB project.
- A form with two buttons: cmdRun and cmdQuit
- A calibrated S11 1-port or N-port measurement active on Channel 1
- Window 1 is visible

See Other SCPI Example Programs

' Session to VISA Default Resource Manager
Private defRM As Long
' Session to PNA
Private viPNA As Long
' VISA function status return code
Private status As Long

Private Sub Form_Load()
    defRM = 0
End Sub

Private Sub cmdRun_Click()
    ' String to receive data from the PNA
    Dim strReply As String * 200

    ' Open the VISA default resource manager
    status = viOpenDefaultRM(defRM)
    If (status < VI_SUCCESS) Then HandleVISAError

    ' Open a VISA session (viPNA) to the PNA at GPIB address 16.
    status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
    If (status < VI_SUCCESS) Then HandleVISAError

    ' Need to set the VISA timeout value to give all our GPIB Reads
    ' sufficient time to complete before a timeout error occurs.
    ' For this example, let's try setting the limit to
    ' 10000 milliseconds (10 seconds).
    status = viSetAttribute(viPNA, VI_ATTR_TMO_VALUE, 10000)
    If (status < VI_SUCCESS) Then HandleVISAError

    ' Get the catalog of all the measurements currently on Channel 1.
    status = myGPIBWrite(viPNA, "CALC1:PAR:CAT?")
    If (status < VI_SUCCESS) Then HandleVISAError
    status = myGPIBRead(viPNA, strReply)
    If (status < VI_SUCCESS) Then HandleVISAError
If an S11 measurement named "MY_S11" doesn't already exist, then create it.

If InStr(strReply, "MY_S11") = 0 Then
    status = myGPIBWrite(viPNA, "CALC1:PAR:DEF MY_S11,S11")
    If (status < VI_SUCCESS) Then HandleVISAError
End If
strReply = ""

Get the catalog of all the trace numbers currently active in Window 1.
status = myGPIBWrite(viPNA, "DISP:WIND1:CAT?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

If a trace number 4 already exists in Window 1, then this will remove it.
If InStr(strReply, "4") > 0 Then
    status = myGPIBWrite(viPNA, "DISP:WIND1:TRAC4:DEL")
    If (status < VI_SUCCESS) Then HandleVISAError
End If

Set trace number 4 to MY_S11.
status = myGPIBWrite(viPNA, "DISP:WIND1:TRAC4:FEED MY_S11")
If (status < VI_SUCCESS) Then HandleVISAError

Set up trace view so we are viewing only the data trace.
status = myGPIBWrite(viPNA, "DISP:WIND1:TRAC4 ON")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBWrite(viPNA, "DISP:WIND1:TRAC4:MEM OFF")
If (status < VI_SUCCESS) Then HandleVISAError

Select MY_S11 as the measurement to be used for the Confidence Check.
status = myGPIBWrite(viPNA, "SENS1:CORR:CCH:PAR MY_S11")
If (status < VI_SUCCESS) Then HandleVISAError

Acquire the S11 confidence check data from ECal Module A into the memory buffer (asking for an OPC reply when it's done).
status = myGPIBWrite(viPNA, "SENS1:CORR:CCH:ACQ ECAL1:*OPC?")
If (status < VI_SUCCESS) Then HandleVISAError

The PNA sends an OPC reply ("+1") when the confidence data acquisition into memory is complete, so this Read is waiting on the reply until it is received.
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

Turn on trace math so the trace shows data divided by memory.
You can be confident the S11 calibration is reasonably good if
the displayed trace varies no more than a few tenths of a dB from 0 dB across the entire span.
status = myGPIBWrite(viPNA, "CALC1:PAR:SEL MY_S11")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBWrite(viPNA, "CALC1:MATH:FUNC DIV")
If (status < VI_SUCCESS) Then HandleVISAError
End Sub

Private Sub cmdQuit_Click()
' Turn off trace math
status = myGPIBWrite(viPNA, "CALC1:MATH:FUNC NORM")
If (status < VI_SUCCESS) Then HandleVISAError

' Conclude the confidence check to set the ECal module back to it's idle state.
status = myGPIBWrite(viPNA, "SENS1:CORR:CCH:DONE")
If (status < VI_SUCCESS) Then HandleVISAError

' Close the resource manager session (which also closes the session to the PNA).
If defRM <> 0 Then Call viClose(defRM)

' End the program
End
End Sub

Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long
' The "+ Chr$(10)" appends an ASCII linefeed character to the output, for terminating the write transaction.
myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)
End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation
End
End Sub
Perform a Source and Receiver Power Cal using SCPI

Programming the PNA using COM or using SICL/VISA over LAN (as in this example) leaves the PNA free to control GPIB devices as needed.

The first example, using Visual Basic, demonstrates the following:

- Performing a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

The second example performs a Receiver Power Cal using VBScript.

Learn more about Power Calibrations.
See an example that Uploads a Source Power Cal.

---

Other SCPI Example Programs

To run this program, you need:

- One of the following power meters connected to the PNA through GPIB: E4416A, E4417A, E4418A/B, E4419A/B, 437B, 438A, EPM-441A, EPM-442A

**Note:** If your power meter is other than these, you can create your own Power Meter Driver using our template.

- Your PC and PNA both connected to a LAN (for communicating with each other).
- The SICL and VISA components of Agilent’s I/O Libraries software installed on your PC (both are included when you install the software, unless you already have another vendor’s VISA installed. Then specify Full SICL and VISA installation to overwrite the other vendor’s VISA.
- The module visa32.bas added to your VB project.
- A form with one button labeled cmdRun.
- A VISA interface configured on your remote PC to control the PNA. This could be GPIB interface or a VISA LAN Client.
- On the PNA connect a Thru cable from port 1 to port 2.

**Note:** The SOURce:POWer:CORRection:COLLect:ACQuire command, when used with a power meter, cannot be sent over the GPIB unless the power meter is connected to a different GPIB interface. See the alternative methods described in the command details.

'Session to VISA Default Resource Manager

Private defRM As Long
Private viPNA As Long
' VISA function status return code
Private status As Long

Private Sub Form_Load()
defRM = 0
End Sub

Private Sub cmdRun_Click()
' String to receive data from the PNA.
' Dimensioned large enough to receive scalar comma-delimited values
' for 21 frequency points (20 ASCII characters per point)
Dim strReply As String * 420

Dim strStimulus, strCalValue
Dim strResult As String

' Open the VISA default resource manager
status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then HandleVISAError

' Open a session (viPNA) to the PNA at "address 16" on the VISA
' interface configured as "GPIB1" on this PC. This could be a
' VISA LAN Client pointing to the SICL LAN Server on the PNA, or
' an actual GPIB interface on this PC connected to the PNA GPIB
' (in which case the power meter would need to be connected to a
' different GPIB interface on the PNA, such as the Agilent 82357A
' USB-to-GPIB).
status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then HandleVISAError

' Set the number of sweep points to 21 on Channel 1.
status = myGPIBWrite(viPNA, "SENS1:SWE:POIN 21")
If (status < VI_SUCCESS) Then HandleVISAError
' Specify the GPIB address of the power meter
' that will be used in performing the calibration.
status = myGPIBWrite(viPNA, "SYST:COMM:GPIB:PMET:ADDR 13")
If (status < VI_SUCCESS) Then HandleVISAError

' Turn use of the loss table OFF (this assumes there is
' virtually no loss in the RF path to the power sensor
' due to a splitter, coupler or adapter).
status = myGPIBWrite(viPNA, "SOUR:POW:CORR:COLL:TABL:LOSS OFF")
If (status < VI_SUCCESS) Then HandleVISAError

' Turn frequency checking OFF (so one power sensor is used for the entire cal
' acquisition sweep regardless of frequency span).
status = myGPIBWrite(viPNA, "SOUR:POW:CORR:COLL:FCH OFF")
If (status < VI_SUCCESS) Then HandleVISAError

' Specify a nominal power accuracy tolerance (NTOLerance) in dB for the
calibration,
' and the maximum number (COUNT) of iterations to adjust power at each point,
' attempting to achieve within tolerance of the desired power. If at any
stimulus
' point the power fails to reach within the set tolerance of the desired power
' after the maximum number of iterations, the power at that point will be set to
' value determined by the last iteration (the Source Power Cal dialog box will
' indicate the FAIL, but we can still apply the cal if desired when it's
' complete).
' Each iteration is based upon a SETTLED power reading (see comments preceding
the
' next two commands below).
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:ITER:NTOL 0.1")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:ITER:COUN 3")
If (status < VI_SUCCESS) Then HandleVISAError

' The worst-case window of power uncertainty (for a calibration which meets
' tolerance) is the sum of the iteration tolerance and the power meter settling
' tolerance (which is described below).
At each stimulus point, the PNA takes power meter readings and determine when they have settled by comparing the magnitude difference between consecutive readings versus a nominal dB tolerance limit (NTOLerance) on that magnitude difference. When consecutive readings are within tolerance of each other, or if they are not within tolerance but we've taken a maximum number of readings (COUNT), the PNA does a weighted average of the readings taken at that stimulus point and that is considered our settled power reading.

```vbnet
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:AVER:NTOL 0.1")
If (status < VI_SUCCESS) Then HandleVISAError

status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:AVER:COUN 5")
If (status < VI_SUCCESS) Then HandleVISAError
```

Specify if the cal power level is offset (positive value for a gain, negative value for a loss) from the PNA port power setting on the channel when no source power cal is active. This is to account for components between the PNA test port and cal reference plane. In this example, we will calibrate at the PNA test port, so there is no offset (it is zero).

```vbnet
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:OFFS 0 DB")
If (status < VI_SUCCESS) Then HandleVISAError
```

Show the source power cal dialog during the source power cal acquisition. (this is the default, so this command is only necessary if this setting may have been changed beforehand, perhaps by another program).

```vbnet
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:DISP ON")
If (status < VI_SUCCESS) Then HandleVISAError
```

Specify the method (type of device) that will be used to perform the cal. Choose from power meter (PMETer), power meter and receiver (PMReceiver) or just receiver (RECeiver). PMReceiver uses the power meter for the first iteration of each point and the PNA's reference receiver for subsequent iterations, so is much faster than using power meter only. But the power meter accounts for compression when calibrating at the output of an active device, whereas the reference receiver cannot unless it is coupled to the cal reference plane (on a PNA
' which allows direct access to the receivers).
' Perform the source power cal acquisition sweep using the sensor attached to
' Channel A of the power meter (asking for an OPC reply when it’s done). This
' assumes that the power sensor is already connected to Port 2 of the PNA.

' We'll put up an hourglass cursor while waiting for the acquire to complete.
Screen.MousePointer = vbHourglass
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:ACQ PMET,'ASEN';*OPC")
If (status < VI_SUCCESS) Then HandleVISAError
' Other valid selections would be the following:
' This mode uses Power Meter and Reference Receiver
'status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:ACQ PMR,'BSEN';*OPC")
' This mode uses PNA receiver only (no power meter)
'status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:ACQ REC,'b1';*OPC")

' In the process of beginning a source power cal acquisition, the PNA searches
' for the power meter on VISA interfaces configured in the Agilent I/O Libraries
' on the PNA. One of those interfaces is the SICL/VISA LAN server, so if this
' program is using that interface, we need to ensure our program is not pending
' an operation on that interface when the PNA wants to search it. So this
' Wait subroutine suspends execution of our program (for 6000 milliseconds =
' 6 seconds), giving the PNA time to search that interface and discover that the
' power meter is not there (the 6 seconds is just to be safe, the search actually
' takes only a few seconds).
' Note: If instead of using the VISA LAN server interface, you are having this
' program communicate with the PNA via it's GPIB interface (which requires the
' power meter be connected to a different GPIB interface on the PNA, such as the
' Agilent 82357A USB-to-GPIB), then this Wait is not needed.
Wait 6000

' The PNA sends an OPC reply ("+1") when the cal acquisition is complete, so
' our Read operation will wait on the reply until it is received. We need to
' set the VISA timeout value long enough to give our Read sufficient time to
' complete before a timeout error occurs. For this example, let's try setting
' the limit to 60000 milliseconds (60 seconds).
status = viSetAttribute(viPNA, VI_ATTR_TMO_VALUE, 60000)
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Change mouse cursor from hourglass back to normal
Screen.MousePointer = vbDefault

' Conclude the calibration. This applies the cal data to PNA channel memory,
' and turns the correction ON for Port 2 on Channel 1,
' but does NOT save the calibration.
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:COLL:SAVE")
If (status < VI_SUCCESS) Then HandleVISAError

' At this point, if you choose to save the instrument state as a ".CST" file,
' the calibration will be saved with the instrument state in that file.

' Prepare for doing data transfer in ASCII format.
status = myGPIBWrite(viPNA, "FORM:DATA ASCII")
If (status < VI_SUCCESS) Then HandleVISAError

' Read the stimulus values from Channel 1.
status = myGPIBWrite(viPNA, "SENS1:X?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strStimulus = Split(strReply, ",")

' Read the source power correction data.
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:DATA?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strCalValue = Split(strReply, ",")

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).
strResult = "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(strStimulus)
    strResult = strResult & Val(strStimulus(i)) & Chr(9) & Val(strCalValue(i)) & Chr(10)
Next
MsgBox strResult
End Sub

Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long

    ' The "+ Chr$(10)" appends an ASCII linefeed character to the
    ' output, for terminating the write transaction.
    myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)
End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation

    ' Close the resource manager session (which also closes
    ' the session to the PNA).
If defRM <> 0 Then Call viClose(defRM)
End
End Sub

Public Sub Wait(ByVal mS_delay As Long)
Dim t0 As Single  
t0 = Timer  
Do While Timer - t0 < mS_delay / 1000  
Dim dummy As Integer  
dummy = DoEvents() ' if we cross midnight, back up one day  
If Timer < t0 Then t0 = t0 - 86400  
Loop  
End Sub

**Perform a Receiver Power Cal**

The SCPI commands in this example are sent over a COM interface using the **SCPIStringParser** object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file, such as Notepad, and save it on the PNA hard drive as *.vbs.

Learn how to setup and run the macro.

Dim pna  
Dim scpi  
Set pna = CreateObject("AgilentPNA835x.Application")  
Set scpi = pna.ScpiStringParser  
' For simplicity, this example starts from the preset instrument state  
scpi.Execute "SYST:PRESet"  
' Turn off continuous sweep  
scpi.Execute "INITiate:CONTinuous OFF"  
' Select the S11 measurement that was created by the instrument preset  
scpi.Execute "CALCulate:PARameter:SELect 'CH1_S11_1'"  
' Change the measurement parameter to measure the B receiver  
scpi.Execute "CALCulate:PARameter:MODify B,1"  
' Specify the Calibration Type, then Prompt  
' to ensure the receiver is connected to port 1.  
scpi.Execute "SENSe:CORRection:COLLect:METHod RPOWer"  
MsgBox "Connect port 1 to port 2 so power is supplied to the B receiver, then press enter"  
' Acquire the power measurement; returning reply to *OPC? when finished.  
response = scpi.Execute("SENSe:CORRection:COLLect:ACQuire POWER;*OPC?")  
' Compute the error term, store to calset and turn on the calibration.  
response = scpi.Execute("SENSe:CORRection:COLLect:SAVE")  
MsgBox "Done with calibration."

Last modified:

9/22/06  Modified for receiver only feature
Uploading a Source Power Cal using SCPI

Programming the PNA using COM or using SICL/VISA over LAN (as in this example) leaves the PNA free to control GPIB devices as needed. This Visual Basic program demonstrates:

- Uploading a source power calibration of Port 2 for Channel 1.
- Reading the calibration data.

Learn more about Power Calibrations

Other SCPI Example Programs

To run this program, you need:

- Your PC and PNA both connected to a LAN (if using VISA LAN server / client).
- The SICL and VISA components of Agilent's I/O Libraries software installed on your PC (both are included when you install the software, unless you already have another vendor's VISA installed. Then specify Full SICL and VISA installation to overwrite the other vendor's VISA.
- The module visa32.bas added to your VB project.
- A form with two buttons: cmdRun and cmdQuit.
- A VISA interface configured on your remote PC to control the PNA. This could be GPIB interface or a VISA LAN Client.

```vbscript
' Session to VISA Default Resource Manager
Private defRM As Long
' Session to PNA
Private viPNA As Long
' VISA function status return code
Private status As Long
Private Sub Form_Load()
defRM = 0
End Sub
Private Sub cmdRun_Click()
  ' String to receive data from the PNA.
  ' Dimensioned large enough to receive scalar comma-delimited values
  ' for 21 frequency points (20 ASCII characters per point)
Dim strReply As String * 420
Dim strPower As String, strCalPower As String
Dim strStimulus, strCalValue
Dim strResult As String
  ' Open the VISA default resource manager
```
status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then HandleVISAError

' Open a session (viPNA) to the PNA at "address 16" on the VISA
' interface configured as "GPIB0" on this PC.
status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then HandleVISAError

' Set the number of sweep points to 2 on Channel 1.
status = myGPIBWrite(viPNA, "SENS1:SWE:POIN 2")
If (status < VI_SUCCESS) Then HandleVISAError

' Ensure there's currently no source power cal on for this channel and port.
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR OFF")
If (status < VI_SUCCESS) Then HandleVISAError

' Specify if the cal power level is offset (positive value for a gain, negative
' value for a loss) from the PNA port power setting on the channel when no source
' power cal is active. This is to account for components between the PNA test
' port and cal reference plane. In this example, let's set up our calibration
' at the output of an amplifier with 15 dB gain.
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:OFFS 15 DB")
If (status < VI_SUCCESS) Then HandleVISAError

' Prepare for doing data transfer in ASCII format.
status = myGPIBWrite(viPNA, "FORM:DATA ASCII")
If (status < VI_SUCCESS) Then HandleVISAError

' Send our source power correction data to the PNA. For purpose of simplicity
' in this example, we'll set up for no correction (0) at our start stimulus and
' 0.5 dB at our stop stimulus (recall that our sweep currently has just 2 points).
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:DATA 0,0.5")
If (status < VI_SUCCESS) Then HandleVISAError

' Set the number of sweep points to 21 on Channel 1.
status = myGPIBWrite(viPNA, "SENS1:SWE:POIN 21")
If (status < VI_SUCCESS) Then HandleVISAError

' Read the fixed power level for this port on Channel 1.
status = myGPIBWrite(viPNA, "SOUR1:POW2:LEV?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError
strPower = strReply

' Turn the source power cal on.
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR ON")
If (status < VI_SUCCESS) Then HandleVISAError

' Again read the fixed power level for this port on Channel 1
' (with our calibration turned on, this should now include the 15 dB offset
' we indicated our power amplifier provides).
status = myGPIBWrite(viPNA, "SOUR1:POW2:LEV?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError
strCalPower = strReply

' Read the stimulus values from Channel 1.
status = myGPIBWrite(viPNA, "SENS1:X?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strStimulus = Split(strReply, ",")

' Read back the source power correction data, now interpolated for 21 points
status = myGPIBWrite(viPNA, "SOUR1:POW2:CORR:DATA?")
If (status < VI_SUCCESS) Then HandleVISAError
status = myGPIBRead(viPNA, strReply)
If (status < VI_SUCCESS) Then HandleVISAError

' Tokenize the reply string into an array containing the values
strCalValue = Split(strReply, ",")

' Print the data using a message box (here, Chr returns the ASCII characters
' for Tab (9) and Linefeed (10)).
strResult = "PNA port power = " & Val(strPower) & Chr(10)
strResult = strResult & "Power at reference plane = " & Val(strCalPower) & Chr(10)
    Chr(10)
strResult = strResult & "Stimulus" & Chr(9) & Chr(9) & "Cal Value" & Chr(10)
For i = 0 To UBound(strStimulus)
    strResult = strResult & Val(strStimulus(i)) & Chr(9) & Val(strCalValue(i)) & Chr(10)
Next
MsgBox strResult
End Sub
Private Sub cmdQuit_Click()
    ' Close the resource manager session (which also closes
    ' the session to the PNA).
    If defRM <> 0 Then Call viClose(defRM)

    ' End the program
End
End Sub
Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long
    ' The "+ Chr$(10)" appends an ASCII linefeed character to the
    ' output, for terminating the write transaction.
    myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)
End Function
Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)

' Remove trailing linefeed character
If Right(strIn, 1) = Chr(10) Then strIn = Left(strIn, Len(strIn) - 1)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation

' Close the resource manager session (which also closes
' the session to the PNA).
If defRM <> 0 Then Call viClose(defRM)
End
End Sub
Perform a Sliding Load Calibration using GPIB

This Visual Basic program does a only the sliding load portion of a Calibration.
To run this program, you need:

- An established GPIB interface connection
- A measurement and calibration routine to call this sub-program
- STAN3 set up as a sliding load standard

See Other SCPI Example Programs

Sub slide()
'Measure the sliding load for at least 5 and no more than 7 slides
'Note that "SLSET" and "SLDONE" must be executed before the actual acquisition of a slide
MsgBox "Connect Sliding Load; set to Position 1; then press OK"
GPIB.Write "SENS:CORR:COLL SLSET"
GPIB.Write "SENS:CORR:COLL STAN3;"

MsgBox "Set Sliding Load to position 2; then press OK"
GPIB.Write "SENS:CORR:COLL SLSET"
GPIB.Write "SENS:CORR:COLL STAN3;"

MsgBox "Set Sliding Load to position 3; then press OK"
GPIB.Write "SENS:CORR:COLL SLDONE"
GPIB.Write "SENS:CORR:COLL STAN3;"
End Sub
Load Error Terms during a Cal Sequence

This example requires that you already have a Cal Set named "foo" that contains a 1-port cal on port 1 and a 1-port cal on port 2.

This example starts a Guided Calibration specifying an Unknown Thru. It loads the 1-port Cals from the existing "foo" Cal Set, then recalculates the number of steps required to complete the cal. After loading the 1-port cals, only the Unknown Thru standard is left to acquire.

```
SENS:CORR:COLL:GUID:CONN:PORT1 "APC 3.5 female"
SENS:CORR:COLL:GUID:CONN:PORT2 "APC 3.5 female"
SENS:CORR:COLL:GUID:CKIT:PORT1 "85033D/E"
SENS:CORR:COLL:GUID:CKIT:PORT2 "85033D/E"
SENS:CORR:COLL:GUID:METH UNKN
  ' auto-create user calsets for SCPI
SENS:CORR:PREF:CSET:SAVU 1
SENS:CORR:COLL:GUID:INIT
  ' should return the number 7
SENS:CORR:COLL:GUID:STEPS?
  ' to port 1, from port 1 in calset
SENS:CORR:COLL:GUID:ETER:LOAD "foo",1,1
  ' to port 2, from port 2 in calset
SENS:CORR:COLL:GUID:ETER:LOAD "foo",2,2
  ' should now return the number 1
SENS:CORR:COLL:GUID:STEPS?
  ' measure the unknown thru
SENS:CORR:COLL:GUID:ACQ STAN1
  ' save the cal to new user calset
SENS:CORR:COLL:GUID:SAVE
```
Create New Cal Kit using SCPI

When creating new cal kits programmatically, the order in which cal kit commands are sent can be important. For example to create a kit with opens, shorts, loads, and thurs. Be sure to use the following sequence for each newly defined standard.

1. Programatically select the standard number
2. Programatically select the standard type.
3. Program the cal standard's values.
4. Repeat steps 1, 2, 3 for additional new standards being defined.

10 !
20 !
30 ! This example program demonstrates how to create new PNA calibration kits.
40 !
50 !
60 ! 1) Select a kit not previously defined
70 ! 2) Define open, short, load, and thru cal standards
80 ! Note: Each of the newly defined standards is assigned
90 ! a default connector name. These default connector names
100 ! will be replaced in subsequent steps.
110 ! 3) Use the delete connector command to remove default
120 ! connector names.
130 ! 4) Add connectors. Specify:
140 ! Start and Stop Freq
150 ! Z - Impedance
160 ! sex - MALE, FEMALE, NONE
170 ! media - COAX, WAVE
180 ! cutoff - Frequency for waveguide
190 ! 5) Assign the appropriate connector to each standard
200 ! 6) Modify the class assignments for the standards defined
210 ! 7) Verify the kit values
220 !
230 ! Additional Note: After setting each new cal kit value, it is
240 ! recommended that the program periodically perform queries to
250 ! verify the new values.
260 !
270 ! This will prevent program synchronization issues that can affect
280 ! final values stored within new cal kits.
290 !
300 !-------------------------------------------------------------
310 !
320 ! Set up I/O path
330 ASSIGN @Na TO 716
340 DIM Calkname$[80],Conn$[80]
350 INTEGER Calkitnum
! Designate the kit selection to be used for performing cal's

OUTPUT @Na;"sens:corr:ckit:count?"
ENTER @Na;Calkitnum
Calkitnum=Calkitnum+1
OUTPUT @Na;"sens:corr:coll:ckit "&VAL$(Calkitnum)

! Name this kit with your own name
OUTPUT @Na;"sens:corr:coll:ckit:name "Special 2.4 mm Model 85056"

! Now set up standard #1
OUTPUT @Na;"sens:corr:coll:ckit:stan 1"
OUTPUT @Na;"sens:corr:coll:ckit:stan:type SHORT"
Get_std
OUTPUT @Na;"sens:corr:coll:ckit:stan:char coax"
OUTPUT @Na;"sens:corr:coll:ckit:stan:label "My Short"
Get_label

! Now set up standard #2
OUTPUT @Na;"sens:corr:coll:ckit:stan 2"
OUTPUT @Na;"sens:corr:coll:ckit:stan:type OPEN"
Get_std
OUTPUT @Na;"sens:corr:coll:ckit:stan:char coax"
OUTPUT @Na;"sens:corr:coll:ckit:stan:label "My Open"
Get_label

! Now set up standard #3
OUTPUT @Na;"sens:corr:coll:ckit:stan 3"
OUTPUT @Na;"sens:corr:coll:ckit:stan:type LOAD"
Get_std
OUTPUT @Na;"sens:corr:coll:ckit:stan:char coax"
OUTPUT @Na;"sens:corr:coll:ckit:stan:label "My Fixed Load"
Get_label

! Now set up standard #4
OUTPUT @Na;"sens:corr:coll:ckit:stan 4"
OUTPUT @Na;"sens:corr:coll:ckit:stan:type THRU"
Get_std
OUTPUT @Na;"sens:corr:coll:ckit:stan:char coax"
OUTPUT @Na;"sens:corr:coll:ckit:stan:label "My Thru"
Get_label
850 !
860 DISP "Defining kit std 5..."
870 ! Now set up standard #5
880 OUTPUT @Na;":sens:corr:coll:ckit:stan 5"
890 OUTPUT @Na;":sens:corr:coll:ckit:stan:type SLOAD"
900 Get_std
910 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
920 OUTPUT @Na;":sens:corr:coll:ckit:stan:label "Sliding Load"
930 Get_label
940 !
950 DISP "Defining kit std 6..."
960 ! Now set up standard #6
970 !
980 OUTPUT @Na;":sens:corr:coll:ckit:stan 6"
990 OUTPUT @Na;":sens:corr:coll:ckit:stan:type SHORT"
1000 Get_std
1010 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
1020 OUTPUT @Na;":sens:corr:coll:ckit:stan:label "Short"
1030 Get_label
1040 !
1050 DISP "Defining kit std 7..."
1060 ! Now set up standard #7
1070 OUTPUT @Na;":sens:corr:coll:ckit:stan 7"
1080 OUTPUT @Na;":sens:corr:coll:ckit:stan:type SHORT"
1090 Get_std
1100 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
1110 OUTPUT @Na;":sens:corr:coll:ckit:stan:label "Short"
1120 Get_label
1130 !
1140 DISP "Defining kit std 8..."
1150 ! Now set up standard #8
1160 !
1170 OUTPUT @Na;":sens:corr:coll:ckit:stan 8"
1190 OUTPUT @Na;":sens:corr:coll:ckit:stan:type ARBI"
1200 Get_std
1210 OUTPUT @Na;":sens:corr:coll:ckit:stan:char coax"
1220 OUTPUT @Na;":sens:corr:coll:ckit:stan:TZR 15;"
1230 OUTPUT @Na;":sens:corr:coll:ckit:stan:TZI -9;"
1240 OUTPUT @Na;":sens:corr:coll:ckit:stan:label "Z Load"
1250 Get_label
1260 !
1270 !
1280 ! First remove any old connector names
1300 OUTPUT @Na;":sens:corr:coll:ckit:conn:del"
1310 ! Verify that no connectors are currently installed
1320 OUTPUT @Na;":sens:corr:coll:ckit:conn:cat?"
1330 ENTER @Na;Conn$
1340 PRINT "Verify empty list: ";Conn$
Define your new connectors

OUTPUT @Na;"sens:corr:coll:ckit:conn:add "PSC 2.4",0HZ,999GHZ,50.0,MALE,COAX,0.0"
OUTPUT @Na;"sens:corr:coll:ckit:conn:add "PSC 2.4",0HZ,999GHZ,50.0,FEMALE,COAX,0.0"

Verify that the new connectors are installed

OUTPUT @Na;"sens:corr:coll:ckit:conn:cat?"
ENTER @Na;Conn$
PRINT "Verify new connectors: ";Conn$
DISP ""

Disp "Defining conn std 1..."
Now set up standard #1
OUTPUT @Na;"sens:corr:coll:ckit:stan 1"
Verify_std
OUTPUT @Na;"sens:corr:coll:ckit:conn:snam "PSC 2.4",FEMALE,1"
Print_connector

Disp "Defining conn std 2..."
Now set up standard #2
OUTPUT @Na;"sens:corr:coll:ckit:stan 2"
Verify_std
OUTPUT @Na;"sens:corr:coll:ckit:conn:snam "PSC 2.4",FEMALE,1"
Print_connector

Disp "Defining conn std 3..."
Now set up standard #3
OUTPUT @Na;"sens:corr:coll:ckit:stan 3"
Verify_std
OUTPUT @Na;"sens:corr:coll:ckit:conn:snam "PSC 2.4",FEMALE,1"
Print_connector

Disp "Defining conn std 4..."
Now set up standard #4
OUTPUT @Na;"sens:corr:coll:ckit:stan 4"
Verify_std
OUTPUT @Na;"sens:corr:coll:ckit:conn:snam "PSC 2.4",FEMALE,1"
OUTPUT @Na;"sens:corr:coll:ckit:conn:snam "PSC 2.4",MALE,2"
Print_connector

Disp "Defining conn std 5..."
Now set up standard #5
OUTPUT @Na;"sens:corr:coll:ckit:stan 5"
OUTPUT @Na;"sens:corr:coll:ckit:stan:label "Sliding Load"
Verify_std
OUTPUT @Na;"sens:corr:coll:ckit:conn:snam "PSC 2.4",MALE,1"
Print_connector
1820 !
1830 DISP "Defining conn std 6..."
1840 ! Now set up standard #6
1850 !
1860 OUTPUT @Na;":sens:corr:coll:ckit:stan 6"
1870 Verify_std
1880 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC 2.4"",MALE,1"
1890 Print_connector
1900 !
1910 DISP "Defining conn std 7..."
1920 ! Now set up standard #7
1930 OUTPUT @Na;":sens:corr:coll:ckit:stan 7"
1940 Verify_std
1950 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC 2.4"",MALE,1"
1960 Print_connector
1970 !
1980 DISP "Defining conn std 8..."
1990 ! Now set up standard #8
2000 OUTPUT @Na;":sens:corr:coll:ckit:stan 8"
2010 Verify_std
2020 OUTPUT @Na;":sens:corr:coll:ckit:conn:snam ""PSC 2.4"",MALE,1"
2030 Print_connector
2040 !
2050 DISP "Class assignments..."
2060 !
2070 ! Designate the "order" associated with measuring the standards
2080 !
2090 ! Set Port 1, 1st standard measured to be standard #2
2100 OUTPUT @Na;":sens:corr:coll:ckit:order1 2"
2110 ! Set Port 1, 2nd standard measured to be standard #1
2120 OUTPUT @Na;":sens:corr:coll:ckit:order2 1,6,7"
2130 ! Set Port 1, 3rd standard measured to be standard #3 and #5
2140 OUTPUT @Na;":sens:corr:coll:ckit:order3 3,5"
2150 ! Set Port 1, 4th standard measured to be standard #4
2160 OUTPUT @Na;":sens:corr:coll:ckit:order4 4"
2170 !
2180 ! Set Port 2, 1st standard measured to be standard #2
2190 OUTPUT @Na;":sens:corr:coll:ckit:order5 2"
2200 ! Set Port 2, 2nd standard measured to be standard #1
2210 OUTPUT @Na;":sens:corr:coll:ckit:order6 1,6,7"
2220 ! Set Port 2, 3rd standard measured to be standard #3 and #6
2230 OUTPUT @Na;":sens:corr:coll:ckit:order7 3,5"
2240 ! Set Port 2, 4th standard measured to be standard #4
2250 OUTPUT @Na;":sens:corr:coll:ckit:order8 4"
2260 !
2270 ! Set Port 1, 1st standard
2280 OUTPUT @Na;":sens:corr:coll:ckit:olabel1 ""MyOpen1"
2290 ! Set Port 1, 2nd standard
2300 OUTPUT @Na;":sens:corr:coll:ckit:olabel2 ""MyShorts1""
2310 !   Set Port 1, 3rd standard
2320 OUTPUT @Na":sens:corr:coll:ckit:olabel3 ""MyLoads1""
2330 !   Set Port 1, 4th standard measured to be standard #4
2340 OUTPUT @Na":sens:corr:coll:ckit:olabel4 ""MyThru1"
2350 !
2360 !   Set Port 2, 1st standard
2370 OUTPUT @Na":sens:corr:coll:ckit:olabel5 ""MyOpen2"
2380 !   Set Port 2, 2nd standard
2390 OUTPUT @Na":sens:corr:coll:ckit:olabel6 ""MyShorts2"
2400 !   Set Port 2, 3nd standard
2410 OUTPUT @Na":sens:corr:coll:ckit:olabel7 ""MyLoads2"
2420 !   Set Port 2, 4th standard
2430 OUTPUT @Na":sens:corr:coll:ckit:olabel8 ""MyThrus2"
2440 !
2450 BEEP
2460 DISP "Done!"
2470 END
2480 SUB Get_label
2490 OUTPUT 716;":sens:corr:coll:ckit:stan:label?"
2500 ENTER 716;Label$
2510 PRINT Label$
2520 SUBEND
2530 !
2540 SUB Get_std
2550 OUTPUT 716;":sens:corr:coll:ckit:stan:type?"
2560 ENTER 716;Type$
2570 PRINT Type$
2580 SUBEND
2590 !
2600 SUB Print_connector
2610 DIM Nam$[40]
2620 OUTPUT 716;":sens:corr:coll:ckit:conn:sname?"
2630 ENTER 716;Nam$
2640 PRINT Nam$
2650 SUBEND
2660 !
2670 SUB Verify_std
2680 OUTPUT 716;":sens:corr:coll:ckit:stan:label?"
2690 ENTER 716;Label$
2700 SUBEND
2710 !
Modify a Calibration Kit using SCPI

This Visual Basic program:

- Modifies Calibration kit number 3
- Completely defines standard #4 (thru)

To run this program, you need:

- An established GPIB interface connection

See Other SCPI Example Programs

```vbnet
' Modifying cal kit number 3
Calkitnum = 3

'Designate the kit selection to be used for performing cal's
GPIB.Write "SENSe:CORRection:COLLect:CKIT:SELect " & Val(Calkitnum)

'Reset to factory default values.
GPIB.Write "SENSe:CORRection:COLLect:CKIT:RESet " & Val(Calkitnum)

' Name this kit with your own name
GPIB.Write "SENSe:CORRection:COLLect:CKIT:NAME 'My Cal Kit'

' Assign standard numbers to calibration classes
'Set Port 1, class 1 (S11A) to be standard #8
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer1 8"
'Set Port 1, class 2 (S11B) to be standard #7
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer2 7"
'Set Port 1, class 3 (S11C) to be standard #3
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer3 3"
'Set Port 1, class 4 (S21T) to be standard #4
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer4 4"
'Set Port 2, class 1 (S22A) to be standard #8
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer5 8"
'Set Port 2, class 2 (S22B) to be standard #7
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer6 7"
'Set Port 2, class 3 (S22C) to be standard #3
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer7 3"
'Set Port 2, class 4 (S12T) to be standard #4
GPIB.Write "SENSe:CORRection:COLLect:CKIT:ORDer8 4"

'Set up Standard #4 completely
'Select Standard #4; the rest of the commands act on it
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard 4"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:FMIN 300KHz"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:FMAX 9GHz"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:IMPedance 50"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:DE Lay 1.234 ns"
```
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:LOSS 23e6"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:CO 0"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:C1 1"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:C2 2"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:C3 3"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:L0 10"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:L1 11"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:L2 12"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:L3 13"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:LABel 'My Special Thru'
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:TYPE THRU"
GPIB.Write "SENSe:CORRection:COLLect:CKIT:STANdard:CHARacteristic Coax"
Create and Calibrate a VMC Measurement

This VB Script example creates and calibrates a Vector mixer measurement. To run this example **without modification** you need the following:

- A Mixer setup file saved on the PNA: C:\Program Files\Agilent\Network Analyzer\Documents\Mixer\MyMixer.mxr.
- If the mixer file uses an external LO source, it must also be attached and configured.
- An ECal module that covers the frequency range of the measurement.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do **NOT** need a GPIB connection to run this example. However, some modification is necessary to make the program run on a traditional GPIB Interface.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as VMC.vbs. Learn how to setup and run the macro.

```
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'---Create a Vector Mixer Measurement
'First, delete all measurements on the channel
scpi.Parse "CALC:PAR:DEL:ALL"
'Create a forward scalar mixer measurement and configure
'it in channel 1.
'The first parameter is a unique identifying string
'(specified by the user) to allow subsequent
'commands to be directed at this specific measurement.
scpi.Parse "CALC:CUST:DEF 'My VC21', 'Vector Mixer/Converter', 'VC21'
'Setup the new measurement as the 2nd trace in the active window
scpi.Parse "DISP:WIND:TRAC2:FEED 'My VC21'
'Make the new trace the active measurement
scpi.Parse "CALC:PAR:SEL 'My VC21'
'The parameters of the mixer measurement can now be configured.
'This can be done by either using the SENS:MIX commands
'for each of the parameters or by loading a mixer setup file.
'This example loads a mixer setup file. The path name
'for the mixer file may be loaded from other mapped drives.
scpi.Parse "SENS:MIXer:Load 'C:\Program Files\Agilent\Network Analyzer\Documents\Mixer\MyMixer.mxr'

'-------------Perform A Vector Mixer Calibration-------------
'Initialize an VMC guided calibration for session number 6
scpi.Parse "SENS:CORR:COLL:SESS6:INIT ""VMC"
'This sets the VMC operation to full system cal as opposed to
'performing a mixer characterization only.
```
'This example uses ECal for the 2-port cal portion of the procedure.
To use a mechanical kit you will have to use the following command:
scpi.Parse "SENS:CORR:COLL:SESS6:SMC:TWOPort:OPTion" "MECH"

If you select the mechanical method then you also have to
specify the connector types and the cal kits for each of the ports.
The comments below show an example of how that is done:
scpi.Parse "SENS:CORR:COLL:SESS6:CONN:PORT1:SEL" "APC 3.5 male"
scpi.Parse "SENS:CORR:COLL:SESS6:CONN:PORT2:SEL" "APC 3.5 female"

Choose the between ECal or Mechanical calibration for the
Mixer Characterization portion of the VMC cal.

This command sets the port mapping for the ECal to be used
during the Mixer Characterization portion of the VMC cal.
It is a required command if in the previous command the option
was set to 'ECAL'. The only valid port maps are either 'A1'
or 'B1'.

Specify the ECal module and the ECal characterization for the
two port calibration portion of this session. FCA calibrations
currently only support ECal module number 1. In this example
the factory characterization is used by specifying 0 for the
characterization number.

Specify the ECal module and the ECal characterization for the
Mixer Characterization portion of this session. FCA calibrations
currently only support ECal module number 1. In this example
the factory characterization is used by specifying 0 for the
characterization number.

Specify the thru measurement method. This applies to both ECal
and mechanical calibrations. For ECal 'DEFAULT' will use the ECal
thru. Other choices may be used depending on the genders and types
of the connectors on the test interface.

Omit the isolation part of the 2-port cal

'Turn on auto orientation for the ECal

Tell the wizard to generate and report the number of steps in this
cal session

Determine the number of steps required to complete the calibration.
First send the write command, then the query.

```vba
Dim steps
Dim desc
scpi.Parse "SENS:CORR:COLL:SESS6:STEP"
For i = 1 To steps
```
Display the prompt for each step
MsgBox (desc)

Perform the measurement for each step
scpi.Parse "SENS:CORR:COLL:SESS6:ACQ " & CStr(i)
Next

Finish the cal and save the calset
Msgbox ("VMC Cal Complete!")
Create and Cal an SMC Measurement

This Visual Basic example creates and calibrates a scalar mixer measurement.

To run this example **without modification** you need the following:

- A Mixer setup file saved on the PNA: C:\Program Files\Agilent\Network Analyzer\Documents\Mixer\MyMixer.mxr.
- If the mixer file uses an external LO source, it must also be attached and configured.
- An ECAL module that covers the frequency range of the measurement.
- A power meter must be attached to the PNA. If this example is run in the PNA, the power meter does not need to be attached using a GPIB/USB interface.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example. However, some modification is necessary to make the program run on a traditional GPIB Interface. For example, during the power meter portion of this calibration, scpi.Parse will not process a command until the power meter routine has completed. Traditional GPIB would require a serial polling technique to ensure the routine has completed before proceeding.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as SMC.vbs. **Learn how to setup and run the macro.**

```vbs
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
'---Create a Scalar Mixer Forward Measurement
'First, delete all measurements on the channel
scpi.Parse "CALC:PAR:DEL:ALL"
'Create a forward scalar mixer measurement and configure it in channel 1. The first parameter is a unique identifying string (specified by the user) to allow subsequent commands to be directed at this specific measurement.
scpi.Parse "CALC:CUST:DEF 'My SC21', 'Scalar Mixer/Converter', 'SC21'"

' Setup the new measurement as the 2nd trace in the active window
scpi.Parse "DISP:WIND:TRAC2:FEED 'My SC21'"
'Make the new trace the active measurement
scpi.Parse "CALC:PAR:SEL 'My SC21'"
'The parameters of the mixer measurement can now be configured.
'This can be done by either using the individual SENS:MIX commands for each of the parameters or by loading a mixer setup file. This example loads a mixer setup file. The path name for the mixer file may be loaded from other mapped drives
scpi.Parse "SENS:MIXer:Load ""C:\Program Files\Agilent\Network Analyzer\Documents\Mixer\MyMixer.mxr""
'------------------Perform A Scalar Mixer Calibration----------------------
'Initialize an SMC guided calibration for session number 6
```
scpi.Parse "SENS:CORR:COLL:SESS6:INIT ""SMC"
'
'Select to use an ECal for the 2-port cal portion of the procedure
'To use a mechanical kit you will have to use the following command:
'scp.i.Parse "SENS:CORR:COLL:SESS6:SMC:TWOPort:OPTion ""MECH"
'scp.i.Parse "SENS:CORR:COLL:SESS6:SMC:TWOPort:OPTion ""ECAL"
'
'If you select the mechanical method then you also have to
'specify the connector types and the cal kits for each of the ports.
'The comments below show an example of how that is done:
'scp.i.Parse "SENS:CORR:COLL:SESS6:CONN:PORT1:SEL ""APC 3.5 male"
'scp.i.Parse "SENS:CORR:COLL:SESS6:CONN:PORT2:SEL ""APC 3.5 female"
'Specify the ECal module and the ECal characterization for this
'session.  FCA calibrations currently only support ECal module
'number 1.  In this example the factory characterization is used
'by specifying 0 for the characterization number.
'scp.i.Parse "SENS:CORR:COLL:SESS6:SMC:ECAL:CHAR 1,0"
'Specify the thru measurement method.  This applies to both ECal
'and mechanical calibrations.  For ECal 'DEFAULT' will use the ECal
'thru.  Other choices may be used depending on the genders and types
'of the connectors on the test interface.
'Omit the isolation part of the 2-port cal
'scp.i.Parse "SENS:CORR:COLL:SESS6:SMC:TWOP:OMIT 1"
'Turn on auto orientation for the ECal
'Tell the wizard to generate and report the number of steps in this
'cal session
'Dim steps
'Dim desc
'Determine the number of steps required to complete the calibration.
'First send the write command, then the query.
'scp.i.Parse "SENS:CORR:COLL:SESS6:STEP"
'steps = scp.i.Parse ("SENS:CORR:COLL:SESS6:STEP?")
'For i = 1 To steps
'Display the prompt for each step
'desc = scp.i.Parse ("SENS:CORR:COLL:SESS6:DESC? " & CStr(i))
'MsgBox (desc)
'Perform the measurement for each step
'scp.i.Parse "SENS:CORR:COLL:SESS6:ACQ " & CStr(i)
'Next
'Dim calset
'Finish the cal and save the calset
'calset = scp.i.Parse ("SENS:CORR:COLL:SESS6:SAVE?")
'Msgbox ("SMC Cal Complete!")
Create an SMC Fixed Output Measurement with SCPI

This VB Script example creates a calibrated SMC fixed output measurement using a controlled LO. Then a single sweep is taken and data is retrieved.

This example requires that an external LO be previously setup and named '8360'.

Fixed output measurements require that an external LO source be swept and synchronized with the PNA source. FCA performs this synchronization using the external source configuration settings.

The fastest, and recommended, method of controlling the LO source is Hardware List (BNC) triggering mode. However, in this mode, FCA channels will not respond to manual triggers. Therefore, the example uses the following mechanism to trigger a sweep:

Write "SENS:SWE:MODE HOLD" 'place channel 1 in HOLD mode
Write "INIT:CONT ON" 'place PNA in internal trigger mode
Write "SENS:SWE:GRO:COUNT 1" 'specify that the group count is 1 sweep
Write "SENS:SWE:MODE GROUPS" 'execute group count (1 sweep)
Write "*OPC?" 'wait until the sweep is complete
Read

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You can run a VBScript (*.vbs) program from the PNA using Macros. To run this program, copy the following code into a text editor and save it as a *.vbs file.

```vbs
option explicit
' Setup infrastructure to use the SCPI over COM

dim app
set app = createobject("Agilentpna835x.application")
dim p
set p = app.scipistringparser
dim returnStr

sub Write(command)
if len(returnStr) <> 0 then
    err.Raise 55,"Write","Query Unterminated"
end if
returnStr = p.parse(command)
end sub

sub WriteIgnoreError(command)
returnStr = p.Execute(command)
p.Parse("SYST:ERR?") ' clear error queue
end sub

function Read
if len(returnStr) = 0 then
    err.Raise 55,"Read","Bad read"
end if
```

2454
end if
Read = returnStr
returnStr = ""
end function
Write "SYST:PRES"
' When programming in remote mode, hold mode is recommended
Write "SENS:SWE:MODE HOLD"
' Delete the standard measurement
Write "CALC:PAR:DEL:ALL"
' Create an SC21 measurement
Write "CALC:CUST:DEF 'MySMC', 'Scalar Mixer/Converter', 'SC21'"
Write "DISP:WIND:TRACE:FEED 'MySMC'"
Write "CALC:PAR:SEL 'MySMC'"
' Set number of points to 11
Write "SENS:SWE:POIN 11"
' Setup the mixer parameters for a swept LO, fixed output measurement
Write "SENS:MIX:INP:FREQ:START 200e6"
Write "SENS:MIX:INP:FREQ:STOP 700e6"
Write "SENS:MIX:LO:FREQ:MODE Swept"
Write "SENS:MIX:OUTPUT:FREQ:FIX 3.4e9"
Write "SENS:MIX:OUTP:FREQ:SID HIGH"
Write "SENS:MIX:INP:POW -17"
Write "SENS:MIX:LO:POW 10"
' Specify the LO name, for controlled LO.
' This name is setup in the External Source Config Dialog
'Write "SENS:MIX:LO:NAME '8360'"
' The CALC method calculates the LO frequency from the other parameters,
' It also applies the mixer parameters to the channel.
Write "SENS:MIX:CALC LO_1"
' Create an S11 in the same channel
Write "CALC:CUST:DEF 'MyS11', 'Scalar Mixer/Converter', 'S11'"
Write "DISP:WIND:TRACE2:FEED 'MyS11'"
Write "CALC:PAR:SEL 'MyS11'"
' Create an IPwr in the same channel
Write "CALC:CUST:DEF 'MyIPwr', 'Scalar Mixer/Converter', 'IPwr'"
Write "DISP:WIND:TRACE3:FEED 'MyIPwr'"
Write "CALC:PAR:SEL 'MyIPwr'"

' Create an OPwr in the same channel
Write "CALC:CUST:DEF 'MyOPwr', 'Scalar Mixer/Converter', 'OPwr'"
Write "CALC:PAR:SEL 'MyOPwr'"
Write "DISP:WIND:TRACE4:FEED 'MyOPwr'"

' Perform a single sweep, synchronously. When *OPC returns, the sweep is done
Write "SENS:SWE:GRO:COUN 1"
Write "SENS:SWE:MODE GROUPS"
Write "*OPC?"
Read

' Retrieve the SC21 data
Write "CALC:PAR:SEL 'MySMC'"
Write "CALC:DATA? SDATA"
dim data
data = Read()
wscript.echo(data)

'Retrieve the S11 data
Write "CALC:PAR:SEL 'MyS11'"
Write "CALC:DATA? SDATA"
data = Read()
wscript.echo(data)
Create and Cal a GCA Measurement

This VBS program does the following:

- creates and configures GCA to perform a SMART Sweep
- performs a calibration using an ECal with 3.5 mm Female on Port A and 3.5 mm Male connectors on Port B

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as GCA.vbs. Learn how to setup and run the macro.

See the Gain Compression commands

```vbs
option explicit

Dim CompLevel, Tolerance, StartFreq, StopFreq, NumFreqs, Scale, LinearPower
Dim AcqMode, BackOff, StartPower, StopPower, NumPowers, EnableInterp, CompAlg
Dim DwellTime, IFBandwidth, ShowIterations, host, app, parser

' GCA Settings/Values

' Acquisition Mode:
' naSmartSweep = 0
' naSweepPowerAtEachFreq2D = 1
' naSweepFreqAtEachPower2D = 2

' Compression Algorithm
' naCompressionFromLinearGain = 0
' naCompressionFromMaximumGain = 1
' naBackoffCompression = 2
' naXYCompression = 3

' EndOfSweepOperation
' naDefaultPowerSet = 0
' naSetToStartPower = 1
' naSetToStopPower = 2
' naSetRFOff = 3

CompLevel = 1 ' 1 dB compression level
Tolerance = 0.05 ' SMART Sweep tolerance
```
StartFreq = 1E9
StopFreq = 9E9
NumFreqs = 201
Scale = 0.1
LinearPower = -20
BackOff = 10 ' Not used for Deviation from linear gain
StartPower = -20
StopPower = 8
NumPowers = 60 ' Not used for SMART Sweep
DwellTime = 0.0005 ' Allow some time for DUT bias/thermal effects
IFBandwidth = 1000 ' Reasonable trace noise at -20 dBm
EnableInterp = False ' Disable interpolation
AcqMode = 0 ' Smart Sweep
CompAlg = 0 ' Deviation from linear gain
ShowIterations = False ' Configure SMART to not show iteration results

dim objargs
set objargs = wscript . Arguments
if ( objArgs . Count = 1) then host = objargs (0)

'' Create and Configuration GCA Channel:

set app = CreateObject ("Agilentpna835x.application")
set parser = app .ScpiStringParser
call SetupGCA ( parser ,
    StartFreq ,
    StopFreq ,
    NumFreqs ,
    EnableInterp ,
    Scale ,
    CompLevel ,
    LinearPower ,
    AcqMode ,
    BackOff ,
    StartPower ,
    StopPower ,
    NumPowers ,
    )
CompAlg, _
DwellTime, _
IFBAndwidth, _
ShowIterations, _
call CalGCA ( parser )
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'' GCA Setup
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
sub SetupGCA ( parser, StartFreq, StopFreq, NumFreqs, EnableInterp, Scale, CompLevel, LinearPower, _
                               AcqMode, BackOff, StartPower, StopPower, NumPowers, CompAlg, DwellTime, IFBAndwidth, _
                                           ShowIterations, _
) parser . Parse "*RST "
parser . Parse "CALC:PAR:DEL:ALL "
parser . Parse "CALC:CUST:DEF ""S21"",""Gain Compression"",""S21"" "
parser . Parse "DISP:WIND:TRAC1:FEED ""S21"" "
parser . Parse "CALC:PAR:SEL ""S21"" "
parser . Parse "CALC:CUST:DEF ""CompIn21"",""Gain Compression"",""CompIn21"" "
parser . Parse "DISP:WIND:TRAC2:FEED ""CompIn21"" "
parser . Parse "CALC:CUST:DEF ""DeltaGain21"",""Gain Compression"",""DeltaGain21"" "
parser . Parse "DISP:WIND:TRAC3:FEED ""DeltaGain21"" "
parser . Parse "SENS:SWE:MODE HOLD"
parser . Parse "DISP:WIND1:TRAC3:Y:SCAL:PDIV " & Scale
parser . Parse "DISP:WIND1:TRAC3:Y:RLEV " & -CompLevel
select case AcqMode
    case 0 ' SMART Sweep
        parser . Parse "SENS:GCS:AMOD SMAR"
    case 1 ' 2D Power Sweeps
        parser . Parse "SENS:GCS:AMOD PFREQ"
    case 2 ' 2D Freq Sweeps
        parser . Parse "SENS:GCS:AMOD FPow"
end select
select case CompAlg
    case 0 ' Deviation from linear gain
        parser . Parse "SENS:GCS:COMP:ALG CFLG"
2459
case 1 ' Deviation from max gain
  parser . Parse "SENS:GCS:COMP:ALG CFMG"

case 2 ' Back Off
  parser . Parse "SENS:GCS:COMP:ALG BACK"

case 3 ' XY
  parser . Parse "SENS:GCS:COMP:ALG XYCOM"
end select

if EnableInterp then
  parser . Parse "SENS:GCS:COMP:INT ON"
else
  parser . Parse "SENS:GCS:COMP:INT OFF"
end if

if ShowIterations then
  parser . Parse "SENS:GCS:SMAR:SIT ON"
else
  parser . Parse "SENS:GCS:SMAR:SIT OFF"
end if

parser . Parse "SENS:GCS:COMP:LEV " & CompLevel
parser . Parse "SENS:GCS:COMP:BACK:LEV " & BackOff
parser . Parse "SENS:GCS:COMP:DELT:X " & BackOff
parser . Parse "SENS:GCS:SMAR:STIM " & DwellTime
parser . Parse "SENS:Band " & IFBandwidth
parser . Parse "SENS:SWE:DWEL " & DwellTime
parser . Parse "SOUR:POW:STAR " & StartPower
parser . Parse "SOUR:POW:STOP " & StopPower
parser . Parse "SENS:FREQ:STAR " & StartFreq
parser . Parse "SENS:FREQ:STOP " & StopFreq
parser . Parse "SENS:SWE:POIN " & NumFreqs
parser . Parse "SENS:SWE:MODE SING"

dim str
str = parser.Parse("* OPC ?")

end sub

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Create and Cal a Noise Figure Measurement

This program does the following:

- Setup a Noise Figure Measurement
- Calibrate Noise Figure

This program can be run as a macro in the PNA. To do this, copy the code into a text editor file such as Notepad and save on the PNA hard drive as NF.vbs. Learn how to setup and run the macro. See the Noise figure commands.

See Other SCPI Example Programs

```vbs
' This section gets the PNA application
' starts the scpi parser, and presets the PNA
windowNum = 1
channelNum = 1
set pna=CreateObject("AgilentPNA835x.Application")
set scpi = pna.ScpiStringParser
' Create noise figure measurement
scpi.Parse "SYST:FPR"
scpi.Parse "DISP:WIND ON"
scpi.Parse "CALC:CUST:DEF 'noiseFig', 'Noise Figure Cold Source', 'NF'"
scpi.Parse "DISP:WIND:TRAC:FEED 'noiseFig'"
scpi.Parse "CALC:PAR:SEL 'noiseFig'"
' Substitute appropriate Ecal identification strings here
tunerEcal = "N4691-60004 ECal 02821"
pullEcal = "N4691-60004 ECal 02297"
' configure channel
ConfigureChannel
ConfigureNoiseSettings
' perform calibration
SetupNoiseSource
SetupCalAttributes_Insertable
FinishCalibration
' ------ Support subroutines ------
```
' Configure noise channel
sub ConfigureChannel
  scpi.Parse "SENS:FREQ:START 750MHz"
  scpi.Parse "SENS:FREQ:STOP 5.0GHz"
  scpi.Parse "SENS:SWEEP:POINTS 401"
  scpi.Parse "SENS:BWID 1.0E3"
end sub
'
' Configure noise-specific channel settings
sub ConfigureNoiseSettings
  scpi.Parse "SENS:NOIS:AVER:STAT ON"  ' turn averaging ON
  scpi.Parse "SENS:NOIS:AVER 40"  ' noise averaging
  scpi.Parse "SENS:NOIS:BWID 8MHz"  ' noise bandwidth
  scpi.Parse "SENS:NOIS:GAIN 30"  ' gain of noise receiver
  scpi.Parse "SENS:NOIS:TEMP:AMB 301"  ' ambient temperature, in Kelvin
  scpi.Parse "SENS:NOIS:IMP:COUN 5"  ' number of tuner impedance states
  scpi.Parse "SENS:NOIS:TUN:INP 'B''  ' orientation of tuner input port
  scpi.Parse "SENS:NOIS:TUN:OUTP 'A''  ' orientation of tuner output port
  scpi.Parse "SENS:CORR:TCOL:USER:VAL 300"  ' noise source cold temperature
end sub
sub SetupCalAttributes_Insertable
  scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT1 'APC 3.5 female''
  scpi.Parse "SENS:CORR:COLL:GUID:CONN:PORT2 'APC 3.5 male''


  scpi.Parse "SENS:NOIS:SOUR:CONN 'APC 3.5 male''  ' noise source connector type
  scpi.Parse "SENS:NOISE:CAL:METHOD 'Vector''  ' cal method
  scpi.Parse "SENS:CORR:COLL:GUID:INIT"
end sub
sub SetupNoiseSource
  ' specify the ENR file for the noise source
  enrfile = "C:\Program Files\Agilent\Network Analyzer\Noise\346C_MY44420454.enr"

scpi.Parse "SENS:NOIS:ENR:FILENAME " & enrfile & ""
' set noise source cold temperature
scpi.Parse "SENS:CORR:TCOLd:USER:VAL 301.1"
end sub
' Build the connection list and acquire the calibration
sub FinishCalibration
    steps = scpi.Parse("SENS:CORR:COLL:GUID:STEPS?")
    for i = 1 to steps
        messagebox str
        scpi.Parse "SENS:CORR:COLL:GUID:ACQ STAN" & i
    next
    scpi.Parse "SENS:CORR:COLL:GUID:SAVE 0"
    wscript.echo "Calibration complete"
end sub
Show Custom Cal Windows during a Guided Calibration

This VBScript program shows how to send commands that allow you to view specific 'custom' windows, and sweep specific channels, during a UI (Cal Wizard) or remote calibration.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as CalWindow.vbs. Learn how to setup and run the macro.

These commands are used to show and sweep windows and channels:

- `SENS:CORR:COLL:DISP:WIND`
- `SENS:CORR:COLL:SWE:CHAN`
- `SENS:CORR:COLL:DISP:WIND:AOFF`
- `SENS:CORR:COLL:SWE:CHAN:AOFF`
- `SENS:CORR:COLL:GUID:PACQuire`

See Other SCPI Example Programs

```vbnet
Dim app
Dim scpi
' Create / Get the PNA application.
Set app = CreateObject("AgilentPNA835x.Application")
Set scpi = app.ScpiStringParser
' A comment
'Preset the analyzer
' This creates an S11 measurement in channel 1, window 1
scpi.Execute "SYST:PReset"
' Create and turn on window 2
scpi.Execute "DISPlay:WINDow2:STATE ON"
' Define an S21 measurement in channel 2
scpi.Execute "CALCulate2:PARameter:DEFine 'MyMeas',S21"
' Associate ("FEED") the measurement name ('MyMeas') to WINDow2
' and give the new TRACe a number (1).
scpi.Execute "DISPlay:WINDow2:TRACe1:FEED 'MyMeas'"
```
'The following lines are all you need in order to:
'show and sweep the custom Cal windows during a UI Calibration
'If sending ONLY these commands, make sure you know the
'correct window and channel numbers to show and sweep.
'Flag windows 1 and 2 to show during Ch1 calibration
scpi.Execute "SENS:CORR:COLL:DISP:WIND1 ON"
scpi.Execute "SENS:CORR:COLL:DISP:WIND2 ON"
'Flag channels 1 and 2 to sweep during Ch1 calibration
scpi.Execute "SENS1:CORR:COLL:SWE:CHAN1 ON"
scpi.Execute "SENS1:CORR:COLL:SWE:CHAN2 ON"

' ===========================================================
' The following code performs a remote guided Cal on Ch1.
' From a remote cal, the Cal window does not normally show and sweep
' after the previous standard has been acquired.
' This shows how to include the PACQuire (preview) to view and sweep the Cal
Window.
' The Custom window also shows and sweeps due to the flag commands above.
' The flags are cleared at the end of this section.

'Specify the DUT connectors
scpi.Execute "sens:corr:coll:guid:conn:port1 ""APC 3.5 female"" "
scpi.Execute "sens:corr:coll:guid:conn:port2 ""APC 3.5 male"" "
'Select the Cal Kit for each port being calibrated.
scpi.Execute "sens:corr:coll:guid:ckit:port1 ""85052D"" "
scpi.Execute "sens:corr:coll:guid:ckit:port2 ""85052D"" "
'Initiate the calibration and query the number of steps
scpi.Execute "sens:corr:coll:guid:init"
MsgBox "Number of steps is " + CStr(numSteps)
'Measure the standards
For i = 1 to numSteps
step = "Step " + CStr(i) + " of " + CStr(numSteps)
'send the Preview Acquire command, then prompt
scpi.Execute "sens:corr:coll:guid:PACquire STAN" + CStr(i)
' Do NOT send any Guided Cal commands here or the cal window will not sweep
MsgBox strPrompt, vbOKOnly, step
scpi.Execute "sens:corr:coll:guid:acq STAN" + CStr(i)
Next
' Conclude the calibration
scpi.Execute "sens:corr:coll:guid:save"
MsgBox "Cal is done!"

' Remove the Custom Window flags
scpi.Execute "SENS:CORR:COLL:DISP:WIND:AOFF"
' Remove the channel sweep flags
Getting and Putting Data using SCPI

This Visual Basic Program does the following:

- Reads data from the analyzer
- Puts the data back into memory
- To see the data on the analyzer after running the program, from the front panel click:
  Trace - Math/Memory - Memory Trace

To run this program, you need:

- An established GPIB interface connection

See Other SCPI Example Programs

Note: To change the read and write location of data, removing the comment from the beginning of ONE of the
lines, and replace the comment in the beginning of the SDATA and SMEM lines.

Private Sub ReadWrite_Click()
Dim i As Integer
Dim t As Integer
Dim q As Integer
Dim dat As String
Dim cmd As String
Dim datum() As Double

GPIB.Configure
GPIB.Write "SYSTem:PRESet;*wai"

'Select the measurement
GPIB.Write "CALCulate:PARameter:SELect 'CH1_S11_1'"

'Read the number of data points
GPIB.Write "SENSe1:SWEep:POIN?"
numpts = GPIB.Read

'Turn continuous sweep off
GPIB.Write "INITiate:CONTinuous OFF"

'Take a sweep
GPIB.Write "INITiate:IMMediate;*wai"

'Ask for the Data

'PICK ONE OF THESE LOCATIONS TO READ
GPIB.Write "CALCulate:DATA? FDATA" 'Formatted Meas
GPIB.Write "CALCulate:DATA? FMEM" 'Formatted Memory
GPIB.Write "CALCulate:DATA? SDATA" 'Corrected, Complex Meas
GPIB.Write "CALCulate:DATA? SMEM" 'Corrected, Complex Memory
GPIB.Write "CALCulate:DATA? SCORR1" 'Error-Term Directivity

'Number of values returned per data point
'q = 1 ' Pick this if reading FDATA or FMEM
q = 2 ' Otherwise pick this

'Parse the data
ReDim datum(q, numpts)
For i = 0 To numpts - 1
    For t = 0 To q - 1
        'Read the Data
dat = GPIB.Read(20)
        'Parse it into an array
datum(t, i) = Val(dat)
    Next t
Next i

'PUT THE DATA BACK IN
GPIB.Write "format ascii"

'PICK ONE OF THESE LOCATIONS TO PUT THE DATA
'cmd = "CALCulate:DATA FDATA," 'Formatted Meas
'cmd = "CALCulate:DATA FMEM," 'Formatted Memory
'cmd = "CALCulate:DATA SDATA," 'Corrected, Complex Meas
cmd = "CALCulate:DATA SMEM," 'Corrected, Complex Memory
'cmd = "CALCulate:DATA SCORR1," 'Error-Term Directivity

For i = 0 To numpts - 1
    For t = 0 To q - 1
        If i = numpts - 1 And t = q - 1 Then
            cmd = cmd & Format(datum(t, i))
        Else
            cmd = cmd & Format(datum(t, i)) & ","
        End If
    Next t
Next i

GPIB.Write cmd
End Sub
Getting and Putting Data

This Rocky Mountain Basic example does the following:

1. Takes a sweep, and reads the formatted data trace into an array. The trace is read as a definite length block.

2. Instructs you to remove DUT

3. Downloads the trace back to the analyzer as an definite length block.

See Other SCPI Example Programs

```
100 DIM A$[10], Data(1:51)
110 INTEGER Digits, Bytes
120 !
130 COM /Sys_state/ @Hp87xx, Scode
140 ! Identify I/O Port
150 CALL Iden_port
160 !
170 !
180 OUTPUT @Hp87xx; "SYST:PRES"
190 !
200 OUTPUT @Hp87xx; "CALC:PAR:SEL 'CH1_S11_1'"
210 !
220 ! Set up the analyzer to measure 51 data points.
230 OUTPUT @Hp87xx; "SENS1:SWE:POIN 51; *OPC?"
240 ENTER @Hp87xx; Opc
250 !
260 ! Take a single sweep, leaving the analyzer
270 ! in trigger hold mode.
280 OUTPUT @Hp87xx; "ABOR; :INIT1:CONT OFF; :INIT1; *WAI"
290 !
300 ! Select binary block transfer
310 OUTPUT @Hp87xx; "FORM:DATA REAL, 64"
320 !
330 ! Request the channel 1 formatted data array
340 ! from the analyzer.
350 OUTPUT @Hp87xx; "CALC:DATA? FDATA"
360 !
370 ! Turn on ASCII formatting on the I/O path.
380 ! It is needed for reading the header
390 ! information.
400 ASSIGN @Hp87xx; FORMAT ON
410 !
420 ! Get the data header. "A$" will contain the
430 ! "#" character indicating a block data transfer.
440 ! "Digits" will contain the number of characters
```
for the number of bytes value which follows.

ENTER @Hp87xx USING "%,A,D";A$,Digits

! Get the rest of the header. The number of bytes to capture in the data array will be placed in "Bytes". Note the use of "Digits" in the IMAGE string.

ENTER @Hp87xx USING "%,"&VAL$(Digits)&"D";Bytes

PRINT "HEADER",A$,Digits,Bytes

! Turn off ASCII formatting on the I/O path; it is not needed for transferring binary formatted data.

ASSIGN @Hp87xx;FORMAT OFF

! Get the data.

ENTER @Hp87xx;Data1(*)

! Turn on ASCII formatting again.

ASSIGN @Hp87xx;FORMAT ON

! Get the "end of data" character.

ENTER @Hp87xx;A$

! Display the first three numbers in the array.

DISP "Trace: ";Data1(1);Data1(2);Data1(3);"..."

! Use this time to visually compare the numbers to the visible data trace.

WAIT 5

! Prompt the operator to disconnect the test device and how to continue the program.

DISP "Disconnect the test device -- Press Continue"

PAUSE

! Update the display line.

DISP "Taking a new sweep...";

! Take a sweep so the display shows new data.

OUTPUT @Hp87xx;":INIT1;*WAI"

DISP " Done."

WAIT 5

! Send the header for an indefinite block length data transfer.

DISP "Downloading saved trace...";

! The first byte '3' indicates the next three digits equal number of transfer bytes

! The number of transfer bytes equals 8x the number of tracepoints.

OUTPUT @Hp87xx;"CALC:DATA FDATA, #3408";
! Turn off ASCII formatting.
950  ASSIGN @Hp87xx;FORMAT OFF
960 !
970 ! Send the data array back to the analyzer.
980  OUTPUT @Hp87xx;Data1(*),END
990 !
1000 ! Turn on ASCII formatting again.
1010 ASSIGN @Hp87xx;FORMAT ON
1020 DISP " Done!"
1030 END
1040 !
1050 !*****************************************************************************
1060 ! Iden_port: Identify io port to use
1070 ! Description: This routines sets up the I/O port address for the SCPI interface. For "HP 87xx" instruments, the address assigned to @Hp87xx = 800 otherwise, 716.
1100 ! 716.
1110 !*****************************************************************************
1120 SUB Iden_port
1130    COM /Sys_state/ @Hp87xx,Scode
1140 !
1150    IF POS(SYSTEM$("SYSTEM ID"),"HP 87")<>0 THEN
1160        ASSIGN @Hp87xx TO 800
1170        Scode=8
1180    ELSE
1190        ASSIGN @Hp87xx TO 716
1200        Scode=7
1210    END IF
1220 !
1230 SUBEND !Iden_port
1240 !
External Test Set Control using SCPI

This program demonstrates the use of several External Test Set Control commands.

The SCPI commands in this example are sent over a COM interface using the SCPIStringParser object. You do NOT need a GPIB connection to run this example.

This VBScript (*.vbs) program can be run as a macro in the PNA. To do this, copy the following code into a text editor file such as Notepad and save it on the PNA hard drive as ExtTS.vbs. Learn how to setup and run the macro.

```vbs
' Demonstrate some SCPI commands for external testsets.
Dim pna
Set pna = CreateObject("AgilentPNA835x.Application")
Set scpi = pna.ScpiStringParser
' The K64 testset is only usable on a 4-port PNA
If (pna.NumberOfPorts <> 4) Then
    MsgBox("This program only runs on 4-port analyzers.")
Else
' If Help is active, show the measurement window and help
    scpi.Execute("DISP:ARR TILE")
' Return the list of supported test sets
    list = scpi.Execute("SENS:MULT:CATalog?")
    MsgBox(list)
' ************* K64 *****************
' The K64 is connected using the Testset I/O
' connector. There is no handshake information.
' Therefore, a testset need not be connected.
' Load a configuration file.
    scpi.Execute("SENS:MULT1:TYPE 'Z5623AK64'")
    scpi.Execute("SENS:MULT1:ADDR 0")
' return stuff about the test set
' Returns number of input ports
    Inports = scpi.Execute("SENS:MULT1:INCount?")
    MsgBox("Input Ports: " & CStr(Inports))
' Returns number of output ports
    ports = scpi.Execute("SENS:MULT1:COUNT?")
    MsgBox("Output Ports: " & CStr(ports))
' Returns valid output ports for each input port
```
For portNum = 1 To Inports
    ports = scpi.Execute("SENS:MULT1:PORT" & CStr(portNum) & ":CAT?")
    MsgBox("Port " & CStr(portNum) & " catalog: " & (ports))
Next
' Set different port mapping
scpi.Execute("SENS:MULT1:ALLPorts '1 ext R,2 ext R,3 ext R,4 ext R'")
' Return port mapping
portMap = scpi.Execute("SENS:MULT1:ALLPorts?")
MsgBox("Ports will be mapped to " & CStr(portMap))
' Enable external testset control and execute port mapping. This automatically enables status bar display as well.
scpi.Execute("SENS:MULT1:STATe 1")
MsgBox("Z5623A K64 Enabled")
End If
Transfer Data using GPIB

The following RMB examples transfer data to and from a remote PC using the `MMEM:TRANSfer` command.

Transferring data FROM the PNA -- TO a remote PC:

30 !
40 !     Set up I/O paths
50 !
60 ! Network analyzer address
70 ASSIGN @Na TO 716
75 !
77 ! File to be stored on local computer
80 ! First time -- need to create the file.
90 ! After file name, number records set to 0 (ignored by WinOS)
95 ! Use "PURGE" command to delete if desired.
100 CREATE "mytestdata.s2p",0
110 ASSIGN @File TO "mytestdata.s2p"
120 !
122 !   TRANSFER the data (download)
123 !
125 ! Analyzer has file 'testdata.s2p' in default directory
130 OUTPUT @Na;";MMEM:TRAN? ""testdata.s2p"
135 !
137 ! Now read the bytes coming back from the analyzer in four steps
138 ! (1) Read and dump the first character - '
'
140 ENTER @Na USING ";A;$
141 !
142 ! (2) Read the next character which is the number of digits in the file size
150 ENTER @Na USING ";A;Digit$
160 !
161 ! (3) Use the value of the number of digits to read back the file byte size
170 ! Create query string using this number of digits
180 Img$="",&Digit$&"A"
190 !
200 ! Byte$ holds the number of bytes in string format
210 ENTER @Na USING Img$;Byte$
220 !
225 ! (4) Read the file contents into a buffer and store the buffer contents to a local file
230 ! Allocate a buffer for holding the data
240 ALLOCATE Dat$[VAL(Byte$)]
250 !
260 ! Set up a different image for filling the buffer
270 Img$=Byte$&"A"
280 !
290 ! Retrieve the actual file data
300 ENTER @Na USING Img$;Dat$
305 !
307 ! Now save the file locally.
310 OUTPUT @File;Dat$
Transferring data FROM the remote PC - TO the PNA:

40  ! Set up I/O paths
50  !
60  ! Network analyzer address
70  ASSIGN @Na TO 716
77  ! File to be retrieved from local computer
78  ASSIGN @File TO "mytestdata.s2p"
79  !
120  ! TRANSFER the data
123  !
230  ! Allocate a buffer for holding the data
240  ALLOCATE Dat$[26236]
250  !
260  ! Get data from the file and fill Dat$
270  ENTER @File;Dat$
280  !
325  ! Data to be transferred to analyzer file 'testupld.s2p'
326  ! in default directory.
327  !
328  ! A specific block transfer designator must follow the
329  ! file name:
330  ! '6' specifies 6 digits to follow.
331  ! '026236' matches the buffer size allocated above
332  ! not counting <NL><END> (new line and end of file).
325  OUTPUT @Na;":MMEM:TRAN ":"testupld.s2p",#026236",Dat$
520  END

Last Modified:

26-Jul-2007   Added comments to example
Establish a VISA Session

This Visual Basic program demonstrates how to send a SCPI command using VISA and the Agilent IO libraries. To run this program, you need:

- Your PC and PNA both connected to a LAN (for communicating with each other).
- The SICL and VISA components of Agilent’s I/O Libraries software installed on your PC. Both are included when you install the software, unless you already have another vendor’s VISA installed. Then specify Full SICL and VISA installation to overwrite the other vendor’s VISA.
- The module visa32.bas added to your VB project. After you install VISA, the module will be located at C:\VXI\PnP\WINNT (or equivalent)\INCLUDE\Visa32.bas
- A form with two buttons: cmdRun and cmdQuit.
- Your PC configured to be a VISA LAN Client, and the SICL Server capability enabled on the PNA. See Configure for VISA and SICL

See Other SCPI Example Programs

Note: This example is a piece of a larger VISA program that performs a source power calibration.

```vba
' Session to VISA Default Resource Manager
Private defRM As Long

Private viPNA As Long

Private status As Long

Private Sub Form_Load()
    defRM = 0
End Sub

Private Sub cmdRun_Click()
    ' String to receive data from the PNA.
    ' Dimensioned large enough to receive scalar comma-delimited values
    ' for 21 frequency points (20 ASCII characters per point)
    Dim strReply As String * 420

    ' Open the VISA default resource manager
    status = viOpenDefaultRM(defRM)
    If (status < VI_SUCCESS) Then HandleVISAError

    ' Open a VISA session (viPNA) to the SICL LAN server
    ' at “address 16” on the PNA pointed to by the “GPIB0”
    ' VISA LAN Client on this PC.
    ' CHANGE GPIB0 TO WHATEVER YOU PNA IS SET TO
    status = viOpen(defRM, "GPIB0::16::INSTR", 0, 0, viPNA)
    If (status < VI_SUCCESS) Then HandleVISAError

    ' Need to set the VISA timeout value to give all our calls to
```
' myGPIBRead sufficient time to complete before a timeout error occurs.
' For this example, let's try setting the limit to 30000 milliseconds (30 seconds).
status = viSetAttribute(viPNA, VI_ATTR_TMO_VALUE, 30000)
If (status < VI_SUCCESS) Then HandleVISAError

' Preset the PNA
status = myGPIBWrite(viPNA, "SYST:PRES")
If (status < VI_SUCCESS) Then HandleVISAError

' Print the data using a message box
MsgBox strReply
End Sub

Private Sub cmdQuit_Click()
' Close the resource manager session (which also closes the session to the PNA).
If defRM <> 0 Then Call viClose(defRM)

' End the program
End
End Sub

Private Function myGPIBWrite(ByVal viHandle As Long, ByVal strOut As String) As Long
' The "+ Chr$(10)" appends an ASCII linefeed character to the output, for terminating the write transaction.
myGPIBWrite = viVPrintf(viHandle, strOut + Chr$(10), 0)
End Function

Private Function myGPIBRead(ByVal viHandle As Long, strIn As String) As Long
myGPIBRead = viVScanf(viHandle, "%t", strIn)
End Function

Sub HandleVISAError()
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr, vbExclamation
End
End Sub
This Visual Basic program demonstrates two methods of reading the analyzer’s status registers:

- Polled Bit Method - reads the Limit1 register continuously.
- SRQ Method - enables an interrupt of the program when bit 6 of the status byte is set to 1. The program then queries registers to determine if the limit line failed.

To run this program, you need:

- An established GPIB interface connection
- A form with two buttons: Poll and SRQ Method
- A means of causing the limit line to fail, assuming it passes initially.

```vba
Private Sub Poll_Click()
    ' POLL THE BIT METHOD
    ' Clear status registers
    GPIB.Write "*CLS"

    'Loop FOREVER
    Do
        DoEvents
        GPIB.Write "STATus:QUEStionable:LIMit1:EVENt?"
        onn = GPIB.Read
    Loop Until onn = 2

    MsgBox "Limit 1 Failed"
End Sub

Private Sub SRQMethod_Click()
    'SRQ METHOD
    GPIB.Write "SYSTem:PRESet"
    GPIB.Write "CALCulate:PARameter:SELect 'CH1_S11_1'"
    'slow down the trace
    GPIB.Write "SENS:BWID 150"

    'Setup limit line
    GPIB.Write "CALC:LIM:DATA 2,3e9,6e9,-2,-2"
    GPIB.Write "CALC:LIMit:DISP ON"
    GPIB.Write "CALC:LIMit:STATe ON"

    ' Clear status registers.
    GPIB.Write "*CLS;*wai"
    ' Clear the Service Request Enable register.
    GPIB.Write "*SRE 0"
    ' Clear the Standard Event Status Enable register.
    GPIB.Write "*ESE 0"
```
' Enable questionable register, bit(10) to report to the status byte.
GPIB.Write "STATus:QUEStionable:ENABle 1024"

' Enable the status byte register bit3 (weight 8) to notify controller
GPIB.Write "*SRE 8"

' Enable the onGPIBNotify event
GPIB.NotifyMask = cwGPIBRQS
GPIB.Notify
End Sub

----------------------------------------------------
Private Sub GPIB_OnGPIBNotify(ByVal mask As Integer)
' check to see what failed
' was it the analyzer?
GPIB.Write "*STB?"
onn = GPIB.Read
If onn <> 0 Then
  ' If yes, then was it the questionable register?
  GPIB.Write "STATus:QUEStionable:EVENt?"
  onn = GPIB.Read
  ' Determine if the limit1 register, bit 8 is set.
  If onn = 1024 Then
    ' if yes, then was it trace 1?
    GPIB.Write "STAT:QUES:LIMIT1:EVEN?"
    onn = GPIB.Read
    If onn = 2 Then MsgBox ("Limit Line1 Failed")
  End If
End If
End Sub
Create a Custom Power Meter Driver

This topic requires that you have a working knowledge of Visual Basic.

This topic will help you create your own power meter driver for use with Source Power Calibration on the PNA. If you are using an Agilent Power Meter to perform a Source Power Calibration, you do NOT need to create your own driver.

Your Power Meter driver will be created from a template written in Visual Basic using VISA over the GPIB bus.

**Note:** This procedure applies to Visual Basic 6.0. Applicability to Visual Basic .NET has not yet been investigated.

- **Prepare Template Files**
- **Modify Template Files**
- **Compile, Copy, and Register, Your New Driver**
- **Test Your new Driver**

Other SCPI Example Programs

### Prepare Template Files

1. Copy all the files from the PNA hard drive C:\Program Files\Agilent\Network Analyzer\Automation\Power Meter Driver Template folder, to a folder on your development PC.

2. In Visual Basic click **File**, then **Open Project…**, find **MyPowerMeter.vbp** (a file you copied from the PNA). Click **Open**. This is a VB ActiveX EXE template, which you will fill in to become your driver.

3. Click **Project**, then **MyPowerMeter Properties**. Click the **General** tab.

4. Overwrite the Project Name with a name of your own choosing. This will be the name of your driver’s type library (also the default name of your exe).

   **Note** If the name of your exe does not match the VB Project Name with which it was compiled, registration of the exe on the PNA will not succeed.

5. Set the Project Description. After building your driver if you wish to test it using VB, this is the string that will show up in the VB References list of your test project, and also in the lower pane of the VB Object Browser.

6. Set the Thread Pool size to 1 thread.

7. Click **OK** to close the project properties dialog.

8. From the VB **Project** menu, click **References…**. Ensure that **Agilent PNA Power Meter 1.0 Type Library** and **VISA Library** are checked. Click **OK**.

   **Note:** Agilent's implementation of VISA is installed as part of the Agilent I/O Libraries on the PNA. For help on
VISA, go to the Windows Start button on your PNA, select Programs, Agilent IO Libraries, VISA Help.

Modify Template Files

From Visual Basic View menu click Project Explorer. Expand the Modules and Class Modules folders. Ensure there is one module (WinAPI) and one class module (PowerMeter).

Let's look at the WinAPI module first.

1. In the Project Explorer window, click WinAPI.
2. From the View menu click Code.

There is only one line of code you should need to modify in this module: the value of the string constant named sIDSEARCH. The comments preceding the declaration of that string describe how to change it. The rest of this module contains functions which will use the Microsoft Windows API to insure proper registration of your driver on the PNA. If you know of other Windows API functions you feel might be helpful to call from within your PowerMeter class module (to help in formatting data, for example), this module would be the place to declare them.

Now let's look at the class module.

1. In the Project Explorer window, click PowerMeter.
2. From the View menu click Properties Window. The Instancing property must be set to MultiUse. This allows other applications to create objects from this class, such that one instance of your driver EXE can supply more than one such object at a time.
3. From the View menu click Code.

Do NOT modify the Interfaces to IPowerMeter subroutines and functions. PNA source power cal expects to find these interfaces as they are currently defined.

The only members that you need to supply code to are those containing “Your code here” comments.

In addition, comments have been provided at the beginning of each member to describe the information that member needs to be read from or written to the power meter.

To get an idea of how communicate with the power meter using the VISA functions viWrite and viRead, examine the code which has been implemented for you in IPowerMeter_Connect, IPowerMeter_QueryMeter, and IPowerMeter_WriteMeter.

Compile, Copy, and Register Your New Driver

When your driver is ready to run, you will first need to compile it into an EXE.

From the File menu select Make exe.

After compiling, the following will instruct VB to use the same ID (GUID) every time you re-compile your project.

1. From the Project menu, click PowerMeter Properties.
2. On the Component tab, select Binary Compatibility and click ...
3. Browse to and select your project EXE. Click Open.
4. Click **OK** to close **Project Properties**.

5. Save your project.

6. Copy your driver EXE file to a folder on your PNA (do NOT use C:\Program Files\Agilent\Network Analyzer\Automation\Power Meter Driver Template folder).

7. Run the EXE file. A message box will pop up reporting whether or not registration was successful. If not successful, it will make a suggestion on what to fix.

When your driver is properly registered, PNA Source Power Cal should be able to associate it with the ID string of your power meter.

---

**Test Your Power Meter Driver**

We have also provided a Visual Basic project to test your new Power Meter driver. This project individually calls every IPowerMeter method and property in your driver to verify that it performs correctly. Before running the test your PC and PNA must be configured to communicate using DCOM.

1. Connect your PC and the PNA to LAN.

2. Add your PC logon to the PNA. Both logons and password must match to communicate using DCOM. See [Additional PNA users](#).

3. Configure your driver using DCOM Config on the PNA. This will give you permission to launch and access the driver. See [Configure for COM-DCOM Programming](#).

**Modify the Test Project**

1. In Visual Basic click **File**, then **Open Project...**, find **MyPowerMeterTest.vbp** (a file you copied from the PNA). Click **Open**.

2. From the **Project** menu, click **References...** From the list, find and check your new Power Meter Driver. (It should have been registered on your PC when you successfully made your driver EXE.) Click **OK**.

3. From the **View** menu click **Code**.

4. Modify the **CreateObject** line as follows:
   "Replace **MyPowerMeter** with the Project Name that you chose for your driver"
   "Replace **MyPNA** with the Computer Name of your PNA."
   "For example:

   ```vba
   Set PowerMeterObj = CreateObject("AcmeBrand.PowerMeter", "AGILENT-PNA123")
   ```

   (This assumes that you kept **PowerMeter** as class module name in your driver.)

**Run the Test Project**

Ensure your power meter is connected to the PNA with a GPIB cable.

Put the PNA in system controller mode:
1. From the PNA System menu point to Configure then click SICL/GPIB.

2. In the GPIB box click System Controller.

Run the test project. If there are no errors, the driver is created successfully. If there are errors, try to figure out what went wrong and fix it. Then re-compile, re-copy the .exe to the PNA, and re-run the test. You should not need to re-register the driver or re-modify the test program.
The SCPI `SYSTem` commands used in this example allow you to send GPIB commands to another GPIB device through the PNA. The other GPIB device cannot be connected to the GPIB bus through the PNA rear panel if the PNA is being controlled by a remote PC using that connector. The other device would typically be connected to the PNA using a USB/GPIB interface.

This VB Script example uses the COM `SCPIStringParser` object. However, this is not critical to the use of these commands; they can be sent using the normal syntax of your programming environment. Using the `SCPIStringParser` over LAN allows you to communicate with GPIB devices without requiring your remote PC to have a GPIB interface card installed.

Although this method of pass-through works for most applications, there are a couple of limitations:

- All data is transferred using ASCII format. Therefore, transferring large blocks of data is very slow.
- Only read and write functions are possible. Service Interrupts are not supported.

See Other SCPI Example Programs

```vbnet
option explicit
dim app
set app = CreateObject("AgilentPNA835x.Application")

dim p
set p = app.ScpiStringParser

' Open a new GPIB session on Bus:2 Device:14 Timeout: 100ms
p.Parse "SYST:COMM:GPIB:RDEV:OPEN 2,14,100"
dim handleAsStr

' Retrieve the handle (ID number)
handleAsStr = p.Parse("SYST:COMM:GPIB:RDEV:OPEN?")

' Convert the handle to an integer
dim handleAsInt
handleAsInt = CInt(handleAsStr)

' Send the "*IDN?" query
p.Parse "SYST:COMM:GPIB:RDEV:WRITE " & handleAsInt & ","*IDN?"

' Read its results
dim idn
msgbox idn

' Close the GPIB session
p.Parse "SYST:COMM:GPIB:RDEV:CLOSE " & handleAsInt
```
PNA as Controller and Talker / Listener

This Visual Basic Program uses VISA to do the following:

- Control the PNA using a VISA LAN Client interface on the PNA.
- Control another instrument using the PNA as GPIB controller.
- Queries both the analyzer and other instrument to identify themselves with *IDN?

**Note:** This program can be modified to work from a remote PC to control both instruments. In that case, set up the PNA to be a talker/listener.

To run this program, you need to do the following:

- Add module **visa32.bas** to the VB project. It is located on the analyzer at C:\Program Files\HP\VXIIPN\WINNT\Include\VISA32.bas
- **Configure the PNA for VISA / SICL**
- Set up the PNA to be GPIB system controller.
  1. On the **System** menu, point to **Configure**. Click **SICL / GPIB**
  2. Click **System Controller**
- Connect another instrument to the analyzer through a GPIB cable with Primary address of 13 on GPIB0 interface

**See Other SCPI Example Programs**

Sub main()

' This application run from onboard the PNA
' can control both the PNA and another GPIB instrument.
'
' To run this program the module visa32.bas must be added
to the project.

'VISA function status return code
Dim status As Long
' Session to Default Resource Manager
Dim defRM As Long
' Session to instrument
Dim viPNA As Long
' Session to other GPIB instrument
Dim viInstrument As Long
' String to hold results
Dim strRes As String * 200
On Error GoTo ErrorHandler

status = viOpenDefaultRM(defRM)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Open the session to the PNA
status = viOpen(defRM, "GPIB1::16::INSTR", 0, 0, viPNA)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Ask for the PNA's ID.
status = viVPrintf(viPNA, "*IDN?" + Chr$(10), 0)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Read the ID as a string.
status = viVScanf(viPNA, "%t", strRes)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Display the results
MsgBox "PNA is: " + strRes

'Open the session to the other instrument
status = viOpen(defRM, "GPIB0::13::INSTR", 0, 0, viInstrument)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Ask for the instrument's ID.
status = viVPrintf(viInstrument, "*IDN?" + Chr$(10), 0)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Read the ID as a string.
status = viVScanf(viInstrument, "%t", strRes)
If (status < VI_SUCCESS) Then GoTo VisaErrorHandler

'Display the results
MsgBox "Other instrument is: " + strRes
' Close the resource manager session (which closes everything)
Call viClose(defRM)
End

ErrorHandler:
'Display the error message
MsgBox "*** Error : " + Error$, MB_ICONEXCLAMATION
End

VisaErrorHandler:
Dim strVisaErr As String * 200
Call viStatusDesc(defRM, status, strVisaErr)
MsgBox "*** Error : " + strVisaErr
End
End Sub
**Socket Client**

The following C# example demonstrates how to send a SCPI program to an Agilent TCP socket-enabled instrument such as the PNA or ENA network analyzer. If the command is a query, the program will read the instrument's response. You can add or replace the SCPI commands in this program with your own.

Learn how to enable Sockets communication on the PNA.

For both of the following methods, first copy the example text below into a Notepad file and name it SocketClient.cs.

**To run using Microsoft Visual Studio 2003 or 2005**

1. From the Visual Studio File menu, select New, then Project.

2. In the New Project window, select the following items (noting the location of the file folder it is creating for you) then click OK.

   - Project Type: Visual C#
   - Template: Console Application
   - Project Name: SocketClient

1. Copy SocketClient.cs into the folder that was created in the previous step.

2. In the Solution Explorer window pane, right-click Class1.cs (if Visual Studio 2003) or Program.cs (if Visual Studio 2005). Select Delete to delete that file.

3. In the Solution Explorer, right-click SocketClient, and select Add, then Existing Item....

4. Browse to select SocketClient.cs and click OK.

You should then be able to build the project, and test the resulting SocketClient.exe from a command prompt (shell) window.

**To run using Mono**

Mono is a cross-platform version of .NET. You can download a free version of Mono at http://www.mono-project.com. Once downloaded and installed:

1. Run the Mono command prompt (shell) window.

2. Navigate to the directory where the example SocketClient.cs is stored.

3. Type: MCS SocketClient.cs (builds the .exe and saves in that same folder.)

4. Type mono SocketClient.exe <PNA name or IP address>

This example was compiled and tested successfully with Mono version 1.1.13. It was run on a PC using the Red
Hat version 9.0 distribution of the Linux operating system. It was also run on a PC using Windows XP. This program has not been tested with other versions of Mono, or on other operating systems.

**To run with Agilent T&M Toolkit**

Agilent T&M Toolkit 2.0 is the first version to support communication using Sockets.

Use the following to address the Sockets port: **TCP10::<PNA name or IP address>::5025::SOCKET**

```csharp
using System;
using System.Net;
using System.Net.Sockets;

// This C# "Console Application" example program demonstrates sending
// SCPI commands to an Agilent TCP socket-enabled instrument
// (for example, a PNA or ENA network analyzer), and reading back the
// instrument's response if the command is a query.
namespace CSharpSocketClient
{
    /// <summary>
    /// The class supporting the main entry point for the application.
    /// </summary>
    class MainClass
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static int Main(string[] args)
        {
            try
            {
                if (args.Length != 1)
                {
                    Console.WriteLine("");
                    Console.WriteLine("Usage -- with Microsoft's .NET runtime:");
                    Console.WriteLine("SocketClient servernameoraddress");
                    Console.WriteLine("Example: SocketClient 192.168.0.1");
                    Console.WriteLine("");
                    Console.WriteLine("Usage -- with Mono's (www.mono-project.com) .NET runtime:");
                }
```
Console.WriteLine("mono SocketClient.exe servernameoraddress");
Console.WriteLine("Example: mono SocketClient.exe 192.168.0.1");
return 1;
}

string server = args[0];
Int32 port = 5025; // default socket port number for the PNA
// Create a TcpClient for the server instrument
// and associated with the necessary port number.
TcpClient client = new TcpClient(server, port);
// Send a preset command to the instrument.
Parse(client, "SYST:PRES");
// Query the instrument ID.
string id = Parse(client, "*IDN?");

// Close the client session.
client.Close();
}
catch (ArgumentNullException e)
{
    Console.WriteLine("ArgumentNullException: {0}", e);
}
catch (SocketException e)
{
    Console.WriteLine("SocketException: {0}", e);
}
Console.WriteLine("\nPress Enter to continue...");
Console.Read();
return 0;
}

static string Parse(TcpClient client, string command)
{
    // Translate the passed command into ASCII and store it as a Byte array.
    Byte[] data = System.Text.Encoding.ASCII.GetBytes(command);
    // Get a client stream for reading and writing.
    NetworkStream stream = client.GetStream();
// Send the command to the socket-enabled instrument.
stream.Write(data, 0, data.Length);

// Has to be followed by a linefeed character as terminator.
Byte[] lf = {(Byte)'
'};
stream.Write(lf, 0, 1);
Console.WriteLine("Sent: {0}", command);

// If the message was a query (involved a question mark)
// receive the instrument response.
if (command.IndexOf("?") >= 0)
{
    // Buffer to store the response bytes.
    // For simplicity of this example, we allocate just for a 256-byte maximum
    // response size.
data = new Byte[256];
    // String to store the response ASCII representation.
    string responseData = String.Empty;
    // Read the batch of response bytes.
    Int32 bytes = stream.Read(data, 0, data.Length);
    responseData = System.Text.Encoding.ASCII.GetString(data, 0, bytes);
    Console.WriteLine("Received: {0}", responseData);
    return responseData;
}

return "";
GPIB Fundamentals

The General Purpose Interface Bus (GPIB) is a system of hardware and software that allows you to control test equipment to make measurements quickly and accurately. This topic contains the following information:

- The GPIB Hardware Components
- The GPIB / SCPI Programming Elements
- Specifications
- GPIB Interface Capability Codes

Note: All of the topics related to programming assume that you already know how to program, preferably using a language that can control instruments.

Other Topics about GPIB Concepts

The GPIB Hardware Components

The system bus and its associated interface operations are defined by the IEEE 488 standard. The following sections list and describe the main pieces of hardware in a GPIB system:

GPIB Addresses

Every GPIB instrument must have its own unique address on the bus. The PNA address (716) consists of two parts:

1. The Interface select code (typically 7) indicates which GPIB port in the system controller is used to communicate with the device.
2. The primary address (16) is set at the factory. You can change the primary address of any device on the bus to any number between 0 and 30. To change the analyzer address click System \ Configure \ SICL-GPIB.

The secondary address is sometimes used to allow access to individual modules in a modular instrument system, such as a VXI mainframe. The analyzer does not have secondary addresses.

Controllers

Controllers specify the instruments that will be the talker and listener in a data exchange. The controller of the bus must have a GPIB interface card to communicate on the GPIB.

- The Active Controller is the computer or instrument that is currently controlling data exchanges.
- The System Controller is the only computer or instrument that can take control and give up control of the GPIB to another computer or instrument, which is then called the active controller.

The PNA can NOT be passed control of the GPIB. However, you can communicate with other GPIB devices through the PNA using one of, or a combination of, the following methods:
- Use the SCPI `SYST:COMM:GPIB:RDEV` commands.
- Use VISA or SICL over LAN to accomplish this. See an example.
- Use USB / GPIB Interface
- Use a PNA with dedicated Controller and Talker/Listener GPIB ports.

**Talker / Listener Instruments**
The PNA is configured as a Talker / Listener by default.

- **Talkers** are instruments that can be addressed to send data to the controller.
- **Listeners** are instruments that can be addressed to receive a command, and then respond to the command.
  All devices on the bus are required to listen.

**Cables**
GPB Cables are the physical link connecting all of the devices on the bus. There are eight data lines in a GPIB cable that send data from one device to another. There are also eight control lines that manage traffic on the data lines and control other interface operations.

You can connect instruments to the controller in any arrangement with the following limitations:

- Do not connect more than 15 devices on any GPIB system. This number can be extended with the use of a bus extension.
- Do not exceed a total of 20 meters of total cable length or 2 meters per device, whichever is less.
- Avoid stacking more than three connectors on the back panel of an instrument. This can cause unnecessary strain on the rear-panel connector.

**The GPIB / SCPI Programming Elements**
The following software programming elements combine to become a GPIB program:

- **GPIB / SCPI Commands**
- **Programming Statements**
- **Instrument Drivers**

**GPIB Commands**
The GPIB command is the basic unit of communication in a GPIB system. The analyzer responds to three types of GPIB commands:

1. **IEEE 488.1 Bus-management Commands**
These commands are used primarily to tell some or all of the devices on the bus to perform certain interface operations.
All of the functions that can be accomplished with these commands can also be done with IEEE 488.2 or SCPI commands. Therefore, these commands are not documented in this Help system. For a complete list of IEEE 488.1 commands refer to the IEEE 488 standard. Examples of IEEE 488.1 Commands

- **CLEAR** - Clears the bus of any pending operations
- **LOCAL** - Returns instruments to local operation

2. IEEE 488.2 Common Commands

These commands are sent to instruments to perform interface operations. An IEEE 488.2 common command consists of a single mnemonic and is preceded by an asterisk (*). Some of the commands have a query form which adds a "?" after the command. These commands ask the instrument for the current setting. See a complete list of the Common Commands that are recognized by the analyzer. Examples of IEEE 488.2 Common Commands

- ***OPC** - Operation Complete
- ***RST** - Reset
- ***OPT?** - Queries the option configuration

3. SCPI Commands

The Standard Commands for Programmable Instruments (SCPI) is a set of commands developed in 1990. The standardization provided in SCPI commands helps ensure that programs written for a particular SCPI instrument are easily adapted to work with a similar SCPI instrument. SCPI commands tell instruments to do device specific functions. For example, SCPI commands could tell an instrument to make a measurement and output data to a controller. Examples of SCPI Commands:

- **CALCULATE:AVERAGE:STATE ON**
- **SENSE:FREQUENCY:START?**

For more information on SCPI:

- **The Rules and Syntax of SCPI Commands** provides more detail of the SCPI command structure.
- **SCPI Command Tree** is a complete list of the SCPI commands for the analyzer

**Programming Statements**

SCPI commands are included with the language specific I/O statements to form program statements. The programming language determines the syntax of the programming statements. SCPI programs can be written in a variety of programming languages such as VEE, HP BASIC, or C++. Example of a Visual Basic statement:

- **GPIB.Write "SOURCE:FREQUENCY:FIXED 1000 MHz"**

**Note about examples**

**Instrument Drivers**

Instrument drivers are subroutines that provide routine functionality and can be reused from program to program. GPIB industry leaders have written standards for use by programmers who develop drivers. When programmers write drivers that comply with the standards, the drivers can be used with predictable results. To comply with the
standard, each instrument driver must include documentation describing its functionality and how it should be implemented.

**GPIB Specifications**

**Interconnected devices** - Up to 15 devices (maximum) on one contiguous bus.

**Interconnection path** - Star or linear (or mixed) bus network, up to 20 meters total transmission path length or 2 meters per device, whichever is less.

**Message transfer scheme** - Byte-serial, bit-parallel, asynchronous data transfer using an interlocking 3-wire handshake.

**Maximum data rate** - 1 megabyte per second over limited distances, 250 to 500 kilobytes per second typical maximum over a full transmission path. The devices on the bus determine the actual data rate.

**Address capability** - Primary addresses, 31 Talk and 31 Listen; secondary addresses, 961 Talk and 961 Listen. There can be a maximum of 1 Talker and up to 14 Listeners at a time on a single bus. See also previous section on GPIB addresses.

**GPIB Interface Capability Codes**

The IEEE 488.1 standard requires that all GPIB compatible instruments display their interface capabilities on the rear panel using codes. The codes on the analyzer, and their related descriptions, are listed below:

- **SH1** full source handshake capability
- **AH1** full acceptor handshake capability
- **T6** basic talker, serial poll, no talk only, unaddress if MLA (My Listen Address)
- **TEO** no extended talker capability
- **L4** basic listener, no listen only, unaddress if MTA (My Talk Address)
- **LEO** no extended listener capability
- **SR1** full service request capability
- **RL1** full remote / local capability
- **PPO** no parallel poll capability
- **DC1** full device clear capability
- **DT1** full device trigger capability
- **C1** system controller capability
- **C2** send IFC (Interface Clear) and take charge controller capability
- **C3** send REN (Remote Enable) controller capability
C4  respond to SRQ (Service Request)
The Rules and Syntax of SCPI

Most of the commands used for controlling instruments on the GPIB are SCPI commands. The following sections will help you learn to use SCPI commands in your programs.

Branches on the Command Tree

Command and Query

Multiple Commands

Command Abbreviation

Bracketed (Optional) Keywords

Vertical Bars (Pipes)

MIN and MAX Parameters

Other Topics about GPIB Concepts

Branches on the Command Tree

All major functions on the analyzer are assigned keywords which are called ROOT commands. (See GPIB Command Finder for a list of SCPI root commands). Under these root commands are branches that contain one or more keywords. The branching continues until each analyzer function is assigned to a branch. A root command and the branches below it is sometimes known as a subsystem.

For example, under `SOURCE:POWER` are several branch commands.

Sometimes the same keyword, such as `STATE`, is used in several branches of the command tree. To keep track of the current branch, the analyzer's command parser uses the following rules:

- **Power On and Reset** - After power is cycled or after `*RST`, the current path is set to the root level commands.

- **Message Terminators** - A message terminator, such as a `<NL>` character, sets the current path to the root command level. Many programming language output statements send message terminators automatically. Message terminators are described in Sending Messages to the Analyzer.

- **Colon (:)** - When a colon is between two command keywords, it moves the current path down one level in the command tree. For example, the colon in `:SOURCE:POWER` specifies that `POWER` is one level below `SOURCE`. When the colon is the first character of a command, it specifies that the following keyword is a root level command. For example, the colon in `:SOURCE` specifies that `source` is a root level command.

**Note:** You can omit the leading colon if the command is the first of a new program line. For example, the following two commands are equivalent:

```
SOUR:POW:ATT:AUTO
:SOUR:POW:ATT:AUTO
```
- <WSP> - Whitespace characters, such as <tab> and <space>, are generally ignored. There are two important exceptions:
  
  o Whitespace inside a keyword, such as :CALCULATE, is not allowed.
  
  o Most commands end with a parameter. You must use whitespace to separate these ending parameters from commands. *Always refer to the command documentation*. In the following example, there is whitespace between STATE and ON.

```plaintext
CALCULATE1:SMOOTHING:STATE ON
```

- **Comma (,)** - If a command requires more than one parameter, you must separate adjacent parameters using a comma. For example, the `SYSTEM:TIME` command requires three values to set the analyzer clock: one for hours, one for minutes, and one for seconds. A message to set the clock to 8:45 AM would be `SYSTEM:TIME 8,45,0`. Commas do not affect the current path.

- **Semicolon(;)** - A semicolon separates two commands in the same message without changing the current path. See Multiple Commands later in this topic.

- **IEEE 488.2 Common Commands** - Common commands, such as *RST, are not part of any subsystem. An instrument interprets them in the same way, regardless of the current path setting.

### Command and Query

A SCPI command can be an Event command, Query command (a command that asks the analyzer for information), or both. The following are descriptions and examples of each form of command. GPIB Command Finder lists every SCPI command that is recognized by the analyzer, and its form.

<table>
<thead>
<tr>
<th>Form</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Event commands</strong> - cause an action to occur inside the analyzer.</td>
<td><strong>:INITIATE:IMMEDIATE</strong></td>
</tr>
<tr>
<td><strong>Query commands</strong> - query only; there is no associated analyzer state to set.</td>
<td><strong>:SYSTem:ERRor?</strong></td>
</tr>
</tbody>
</table>
| **Command and query** - set or query an analyzer setting. The query form appends a question mark (?) to the set form | **:FORMat:DATA ! Command**  
**:FORMat:DATA? ! Query** |

### Multiple Commands

You can send multiple commands within a single program message. By separating the commands with semicolons the current path does not change. The following examples show three methods to send two commands:

1. **Two program messages:**

    ```plaintext
    SOURCE:POWER:START 0DBM  
    SOURCE:POWER:STOP 10DBM
    ```
2. **One long message.** A colon follows the semicolon that separates the two commands causing the command parser to reset to the root of the command tree. As a result, the next command is only valid if it includes the entire keyword path from the root of the tree:

```
SOURCE:POWER:START 0DBM; SOURCE:POWER:STOP 10DBM
```

3. **One short message.** The command parser keeps track of the position in the command tree. Therefore, you can simplify your program messages by including only the keyword at the same level in the command tree.

```
SOURCE:POWER:START 0DBM; STOP 10DBM
```

### Common Commands and SCPI Commands

You can send Common commands and SCPI commands together in the same message. (For more information on these types of commands see [GP-IB Fundamentals](#).) As in sending multiple SCPI commands, you must separate them with a semicolon.

**Example** of Common command and SCPI commands together

```
*RST; SENSE:FREQUENCY:CENTER 5MHz; SPAN 100kHz
```

### Command Abbreviation

Each command has a long form and an abbreviated short form. The syntax used in this Help system use uppercase characters to identify the short form of a particular keyword. The remainder of the keyword is lower case to complete the long form.

- **SOUR** - Short form
- **SOURCE** - Long form

Either the complete short form or complete long form must be used for each keyword. However, the keywords used to make a complete SCPI command can be a combination of short form and long form.

The following is **unacceptable** - The first three keywords use neither short or long form.

```
SOURc:Powe:Atten:Auto on
```

The following is **acceptable** - All keywords are either short form or long form.

```
SOUR:POWer:ATT:AUTO on
```

In addition, the analyzer accepts lowercase and uppercase characters as equivalent as shown in the following equivalent commands:

```
source:POW:att:auto ON
Source:Pow:Att:Auto on
```

### Optional [Bracketed] Keywords

You can omit some keywords without changing the effect of the command. These optional, or default, keywords are used in many subsystems and are identified by brackets in syntax diagrams.

**Example** of Optional Keywords

The **HCOPY** subsystem contains the optional keyword **IMMediate** at its first branching point. Both of the following commands are equivalent:
Vertical Bars | Pipes

Vertical bars, or "pipes", can be read as "or". They are used in syntax diagrams to separate alternative parameter options.

Example of Vertical Bars:

```
SOURce:POWer:ATTenuation:AUTO <on|off>
```

Either **ON** or **OFF** is a valid parameter option.

**MIN and MAX Parameters**

The special form parameters "MINimum" and "MAXimum" can be used with commands that specify single frequency (Hz) and time (seconds) as noted in the command documentation. **Note:** Also with these commands, kHZ, MHz, and GHz are accepted as suffixes/units.

The short form (min) and long form (minimum) of these two keywords are equivalent.

- **MAXimum** refers to the largest value that the function can currently be set to
- **MINimum** refers to the smallest value that the function can currently be set to.

**For example**, the following command sets the start frequency to the smallest value that is currently possible:

```
SENS:FREQ:START MIN
```

In addition, the max and min values can also be queried for these commands.

**For example**, the following command returns the smallest value that Start Frequency can currently be set to:

```
SENS:FREQ:START? MIN
```

An error will be returned if a numeric parameter is sent that exceeds the MAX and MIN values.

**For example**, the following command will return an "Out of range" error message.

```
SENS:FREQ:START 1khz
```

Last Modified:

10-Jul-2007  Removed image
### Configure for GPIB, SCPI, and SICL

The following settings are used to configure the PNA for remote control using SCPI commands.

#### How to Configure for SICL / GPIB Operation

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>PNA Menu using a mouse</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For PNA-L and E836x models</strong></td>
<td><strong>Programming Commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click System</td>
</tr>
<tr>
<td></td>
<td>2. then Configure</td>
</tr>
<tr>
<td></td>
<td>3. then SICL/GPIB/SCPI</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>For PNA-X and 'C' models</th>
<th><strong>Programming Commands</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press SYSTEM</td>
<td>1. Click Utility</td>
</tr>
<tr>
<td>2. then [Configure]</td>
<td>2. then System</td>
</tr>
<tr>
<td>3. then [SICL/GPIB]</td>
<td>3. then Configure</td>
</tr>
<tr>
<td></td>
<td>4. then SICL/GPIB/SCPI</td>
</tr>
</tbody>
</table>
Dialog box for **PNA models with 2 GPIB ports**:

**SICL / GPIB**

- **GPIB**
  - **Talker/Listener** Address
  - **System Controller**

**SICL**

- **SICL Enabled**
  - (Standard Instrument Control Library)
  - Address
  - Automatically Enable on Startup

**LAN Sockets/Telnet**

- **Sockets Enabled**
- **Telnet Enabled**

**SCPI Monitor/Input**

- **GPIB Command Processor Console**
- **Monitor GPIB Bus**
- **Show GPIB Bus Monitor Window**

**OK**  **Cancel**  **Help**

**SICL / GPIB** dialog box help

**GPIB**

PNA models with 2 GPIB ports have an Address setting for ONLY the Talker/Listener Port. Learn more about the [PNA models with 2 GPIB ports](#).

- **Talker/Listener** Sets the PNA to receive and send GPIB/SCPI messages to the system controller (external computer).
- **Talker/Listener Address** Sets the PNA talker/listener GPIB address.
- **System Controller** Sets the PNA as the system controller, controlling GPIB communications of external devices. Learn about the [PNA as controller](#). Use the National Instruments interface or the ACE (Agilent Connection Expert) interface to change the System Controller address.

**SICL**

- **SICL Enabled** When checked, the analyzer is capable of running GPIB programs on its computer to control analyzer functions. The programs must be run from a GPIB-capable programming environment (VEE, Visual...
Basic). This mode does not allow control of external GPIB instruments. To uncheck this box, exit the PNA application - (Click File, then Exit). The PNA restarts with the SICL enabled box unchecked unless Automatically Enable on Startup is checked.

Learn more about Configuring for VISA and SICL.

**Address**  Sets the PNA address.

**Automatically Enable on Startup**  When checked, SICL Enabled is automatically selected when starting the PNA application.

**LAN Sockets/Telnet**
Provides ability to communicate with the PNA from a PC that uses a Windows, or non-Windows, operating system.

- These settings are checked by default. If you have security concerns, clear these check boxes.
- These settings remain after the PNA is shutdown and restarted.

**Sockets Enabled**  When checked, provides the ability to control the PNA from a remote SCPI program using port number 5025. See the C# example that illustrates how this is done.

**Telnet Enabled**  When checked, provides the ability to send single SCPI commands from a remote Windows, or non-Windows, PC to the PNA using port number 5024.

How to send single SCPI commands using Telnet:

1. On the remote PC, click **Start**, then **Run**

2. Type: `telnet <computer name> 5024`
   where `<computer name>` is the full computer name of the PNA. See how to find the computer name of the PNA.

3. A Telnet window with a **SCPI>** prompt should appear on the remote PC screen.

4. From the SCPI prompt:
   - Type single SCPI commands
   - If an invalid SCPI command is sent, the prompt will disappear. Press **Enter** or **Ctrl C** to recover the SCPI prompt.
     - To exit the telnet window click `X` in the upper-right corner.
     - To get a normal telnet prompt, press `Ctrl ]` (closing bracket).
     - To close the normal telnet window, type **Quit** and press **Enter**.

**SCPI Monitor / Input**

**GPIB Command Processor Console**  Launches a window that is used to send single SCPI/GPIB commands from the PNA keyboard.

**Note:**  Press **Control+Z**, then enter, to close the console window.
- Type a valid command, with appropriate arguments and press enter.
- Use the arrow keys to recall previous commands.
- The console window may launch behind the PNA application. Press Control+Tab to bring the console window to the top.

**Monitor GPIB Bus** Enables monitoring activity on the GPIB.

**Show GPIB Bus Monitor Window** Shows and hides the window monitoring GPIB activity.

---

### Local and Remote Operation

The analyzer **LCL** and **RMT** (Local and Remote) operation labels appear in the lower right corner of the status bar.

**Note:** The status bar is NOT visible when the analyzer is preset. See [how to make the status bar visible](#).

- **LCL** appears when NOT under SCPI control
- **RMT** appears when under SCPI control. The RMT label does NOT appear when under COM control

Remote operation disables the front panel keys except for the **Macro/Local** key.

To return to Local (front panel) operation, press the **Macro/Local** key

Sending the GPIB "GTL" (go to local) command also returns the analyzer to Local operation.

Sending the GPIB "LLO" (local lockout) command disables the front panel Local button.

---

Last Modified:

21-Feb-2008    Include Windows OS in Telnet/Sockets
Getting Data from the Analyzer

Data is sent from the analyzer in response to program queries. Data can be short response messages, such as analyzer settings, or large blocks of measurement data. This topic discusses how to read query responses and measurement data from the analyzer in the most efficient manner.

Response Message Syntax

Clearing the Output Queue

Response Data Types

Transferring Measurement Data

Note: Some PCs use a modification of the IEEE floating point formats with the byte order reversed. To reverse the byte order for data transfer into a PC, use the FORMat:BORDer command.

Other Topics about GPIB Concepts

Response Message Syntax

Responses sent from the analyzer contain data, appropriate punctuation, and message terminators. <NL><^END> is always sent as a response message terminator. Most programming languages handle these terminators transparent to the programmer.

Response messages use commas and semicolons as separators in the following situations:

- A comma separates response data items when a single query command returns multiple values

  FORM:DATA? 'Query
  ASC, +0 'Analyzer Response

- A semicolon separates response data when multiple queries are sent within the same messages

  SENS:FREQ:STAR?;STOP? --Example Query
  +1.23000000E+008; +7.89000000E+008<NL><^END> 'Analyzer Response

Clearing the Output Queue

After receiving a query, the analyzer places the response message in it's output queue. Your program should read the response immediately after the query is sent. This ensures that the response is not cleared before it is read. The response is cleared when one of the following conditions occur:

- When the query is not properly terminated with an ASCII carriage return character or the GPIB <^END> message.
- When a second program query is sent.
- When a program message is sent that exceeds the length of the input queue
- When a response message generates more response data than fits in the output queue.
- When the analyzer is switched ON.

**Response Data Types**

The analyzer sends different response data types depending on the parameter being queried. You need to know the type of data that will be returned so that you can declare the appropriate type of variable to accept the data. For more information on declaring variables see your programming language manual. The GPIB Command Finder lists every GPIB command and the return format of data in response to a query. The analyzer returns the following types of data:

- **Numeric Data**
- **Character Data**
- **String Data**
- **Block Data**

**Numeric Data**

All numeric data sent over the GPIB is ASCII character data. Your programming environment may convert the character data to numeric data for you. Boolean data (1 | 0 ) is a type of numeric data.

**Character Data**

Character data consists of ASCII characters grouped together in mnemonics that represent specific analyzer settings. The analyzer always returns the short form of the mnemonic in upper-case alpha characters. Character data looks like string data. Therefore, refer to the GPIB Command Finder to determine the return format for every command that can be queried.

**Example** of Character Data

MLOG

**String Data**

String data consists of ASCII characters. String parameters can contain virtually any set of ASCII characters. When sending string data to the analyzer, the string must begin with a single quote ( ' ) or a double quote ( " ) and end with the same character (called the delimiter).

**Note**: The analyzer responds best to all special characters if the string is enclosed in single quotes. If quotes are not used, the analyzer will convert the text to uppercase. The analyzer may not respond as you expect.

The analyzer always encloses data in double quotes when it returns string data.

**Example** of String Data

GPIB.Write "DISP:WINDOW:TITLE:DATA?"

"This is string response data."

**Block Data**
Block data is used to transfer measurement data. Although the analyzer will accept either definite length blocks or indefinite length blocks, it always returns definite length block data in response to queries unless the specified format is ASCII. The following graphic shows the syntax for definite block data:

```
#<num_digits><byte_count><data bytes><NL><END>
```

- `<num_digits>` specifies how many digits are contained in `<byte_count>`
- `<byte_count>` specifies how many data bytes will follow in `<data bytes>`

**Example** of Definite Block Data

```
#17ABC+XYZ
```

Where:

- `#` - always sent before definite block data
- `1` - specifies that the byte count is one digit (7)
- `7` - specifies the number of data bytes that will follow, not counting `<NL><END>`
- `ABC+XYZ` - Data
- `<NL><END>` - always sent at the end of block data

**Transferring Measurement Data**

Measurement data is blocks of numbers that result from an analyzer measurement. Measurement data is available from various processing arrays within the analyzer. For more information on the analyzer's data processing flow, see [Accessing Data Map](#). Regardless of which measurement array is read, transferring measurement data is done the same.

**See an example.**

When transferring measurement data, the **FORMat DATA** command allows you to choose from the following two data types:

- REAL
- ASCII

The following graphic shows the differences in transfer times between the two:
REAL Data

REAL data (also called floating-point data) types transfer faster. This is because REAL data is binary and takes about half the space of ASCII data. The disadvantage of using REAL data is that it requires a header that must be read. See definite length block data. The binary floating-point formats are defined in the IEEE 754-1985 standard. The following choices are available in REAL format:

- **REAL,32** - IEEE 32-bit format - single precision (not supported by HP BASIC)
- **REAL,64** - IEEE 64-bit format - double precision

ASCII Data

The easiest and slowest way to transfer measurement data is to use ASCII data. ASCII data is sent if the data contains both numbers and characters (the setting of FORMat:DATA is ignored). ASCII data is separated by commas.

Last Modified:

26-Jul-2007  Added link to example
Synchronizing the PNA and Controller

Synchronizing the PNA and Controller means to keep PNA and the controller working at approximately the same pace.

The Problem

The controller sends commands to the PNA as fast as the GPIB bus will allow. The PNA stores these commands in the PNA Input queue. However, the PNA executes those commands at a slower rate than they are accepted. If left unchecked, the PNA input buffer will contain a long list of commands waiting to be executed. At some point, the controller will send a query command which requires a response from the PNA. The controller will not send more commands until a response is received. It will wait for a response from the PNA for the amount of time set by the Timeout setting. If the PNA is working off a long list of commands in the input buffer, if may not execute and respond to the query command until the controller has quit waiting, or "timed out".

The Solution

The easiest way to keep the controller and the PNA "synched" is to send query commands often. This stops the controller from sending more commands until the PNA executes and responds to the query. This limits the number of commands that are waiting in the PNA input queue to be processed. Although any query will stop the controller from sending more commands, a good practice is to send *OPC? Most of the time, as soon as this query is executed, it will immediately reply. The exception to this is the Overlapped command.

- **Sequential** commands are executed quickly and in the order in which they are received.
- **Overlapped** commands take longer to execute. Therefore, they allow the PNA to execute other commands while waiting. However, the programmer may want to prevent the analyzer from processing new commands until the overlapped command has completed. If the PNA is executing an overlapped command when a *OPC? is received, it will wait until the overlapped command is complete.

Note: The analyzer has two overlapped commands:

- INITIate:IMMediate
- SENSE:SWEep:MODE GROUPS (when INIT:CONT is ON)

Analyzer Queues

Queues are memory buffers that store messages until they can be processed. The analyzer has the following queues:

- **Input Queue**
- **Output Queue**
- **Error Queue**

Input Queue
The controller sends statements to the analyzer without regard to the amount of time required to execute the statements. The input queue is very large (31k bytes). It temporarily stores commands and queries from the controller until they are read by the analyzer's command parser. The input queue is cleared when the analyzer is switched ON.

**Output Queue**

When the analyzer parses a query, the response is placed in the output queue until the controller reads it. Your program should immediately read the response or it may be cleared from the output queue. The following conditions will clear a query response:

- When a second query is sent before reading the response to the first. This does not apply when multiple queries are sent in the same statement.
- When a program statement is sent that exceeds the length of the input queue.
- When a response statement generates more data than fits in the output queue.
- When the analyzer is switched ON.

**Error Queue**

Each time the analyzer detects an error, it places a message in the error queue. When the `SYSTEM:ERROR?` query is sent, one message is moved from the error queue to the output queue so it can be read by the controller. Error messages are delivered to the output queue in the order they were received. The error queue is cleared when any of the following conditions occur:

- When the analyzer is switched ON.
- When the `*CLS` command is sent to the analyzer.
- When all of the errors are read.

If the error queue overflows, the last error is replaced with a "Queue Overflow" error. The oldest errors remain in the queue and the most recent error is discarded.

**Synchronization Methods**

The following common commands are used to synchronize the analyzer and controller. Examples are included that illustrate the use of each command in a program. See the SCPI command details to determine if a command is an overlapped command.

- `*WAI`
- `*OPC?`
- `*OPC`

**WAI**

The `*WAI` command:

- **Stops the analyzer** from processing subsequent commands until all overlapped commands are completed.
It does NOT stop the controller from sending commands to this and other devices on the bus. This is the easiest method of synchronization.

**Example** of the *WAI command

```
GPIB.Write "ABORT;:INITIATE:IMMEDIATE" 'Restart the measurement.
GPIB.Write "CALCULATE:MARKER:SEARCH:MAXIMUM" 'Search for max amplitude.
GPIB.Write "CALCULATE:MARKER:X?" 'Which frequency?
```

The following timeline shows how the processing times of the three commands relate to each other:

```
<table>
<thead>
<tr>
<th>INITIATE:IMMEDIATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCULATE:MARKER:MAXIMUM</td>
</tr>
<tr>
<td>CALCULATE:MARKER:X?</td>
</tr>
</tbody>
</table>
```

INITIATE:IMMEDIATE is an overlapped command; it allows the immediate processing of the sequential command, CALCULATE:MARKER:SEARCH:MAXIMUM. However, the INITIATE:IMMEDIATE is not considered complete until the measurement is complete. Therefore, the marker searches for maximum amplitude before the measurement completes. **The CALCULATE:MARKER:X? query could return an inaccurate value.**

To solve the problem, insert a *WAI command.

```
GPIB.Write "ABORT;:INITIATE:IMMEDIATE" 'Restart the measurement.
GPIB.Write "*WAI" 'Wait until complete.
GPIB.Write "CALCULATE:MARKER:MAXIMUM" 'Search for max amplitude.
GPIB.Write "CALCULATE:MARKER:X?" 'Which frequency
```

The timeline now looks like this:

```
<table>
<thead>
<tr>
<th>ABORT;:INITIATE:IMMEDIATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>*WAI</td>
</tr>
<tr>
<td>CALCULATE:MARKER:MAXIMUM</td>
</tr>
<tr>
<td>CALCULATE:MARKER:X?</td>
</tr>
</tbody>
</table>
```

The *WAI command keeps the MARKER:SEARCH:MAXIMUM from taking place until the measurement is completed. **The CALCULATE:MARKER:X? query returns the correct value.**

**Note:** Although *WAI stops the analyzer from processing subsequent commands, it does not stop the controller. The controller could send commands to other devices on the bus.

**OPC?**

The *OPC? query stops the controller until all pending commands are completed.

In the following example, the Read statement following the *OPC? query will not complete until the analyzer responds, which will not happen until all pending commands have finished. Therefore, the analyzer and other devices receive no subsequent commands. A "1" is placed in the analyzer output queue when the analyzer completes processing an overlapped command. The "1" in the output queue satisfies the Read command and the
Example of the *OPC? query

This program determines which frequency contains the maximum amplitude.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIB.Write &quot;ABORT; :INITIATE:IMMEDIATE&quot;!</td>
<td>Restart the measurement</td>
</tr>
<tr>
<td>GPIB.Write &quot;*OPC?&quot;</td>
<td>Wait until complete</td>
</tr>
<tr>
<td>Meas_done = GPIB.Read</td>
<td>Read output queue, throw away result</td>
</tr>
<tr>
<td>GPIB.Write &quot;CALCULATE:MARKER:MAX&quot;</td>
<td>Search for max amplitude</td>
</tr>
<tr>
<td>GPIB.Write &quot;CALCULATE:MARKER:X?&quot;</td>
<td>Which frequency?</td>
</tr>
<tr>
<td>Marker_x = GPIB.Read</td>
<td></td>
</tr>
<tr>
<td>PRINT &quot;MARKER at &quot; &amp; Marker_x &amp; &quot; Hz&quot;</td>
<td></td>
</tr>
</tbody>
</table>

**OPC**

The *OPC command allows the analyzer and the controller to process commands while processing the overlapped command.

When the analyzer completes processing an overlapped command, the *OPC command sets bit 0 of the standard event register to 1. This requires polling of status bytes or use of the service request (SRQ) capabilities of your controller. See Reading the Analyzer's Status Registers for more information about the standard event status register, generating SRQs, and handling interrupts.

**Note:** Be careful when sending commands to the analyzer between the time you send *OPC and the time you receive the interrupt. Some commands could jeopardize the integrity of your measurement. It also could affect how the instrument responds to the previously sent *OPC.

Example of polled bit and SRQ processes.

When To Synchronize the Analyzer and Controller

The need to synchronize depends upon the situation in which the overlapped command is executed. The following section describes situations when synchronization is required to ensure a successful operation.

- **Completion of a Measurement**
- **Measurements with External Trigger**
- **Averaged Measurements**

Completion of a Measurement

To synchronize the analyzer and controller to the completion of a measurement, use the ABORT; INITIATE:IMMEDIATE command sequence to initiate the measurement.

This command sequence forces data collection to start (or restart) under the current measurement configuration. A restart sequence, such as ABORT; INITIATE:IMMEDIATE is an overlapped command. It is complete when all operations initiated by that restart command sequence, including the measurement, are finished. The *WAI,*OPC? and *OPC commands allow you to determine when a measurement is complete. This ensures that valid measurement data is available for further processing.

Measurements with External Trigger

To use an external trigger, synchronize the analyzer and controller before the trigger is supplied to the measurement. Setup the analyzer to receive a trigger from an external source (wired to the EXTERNAL TRIGGER
connector on the rear panel. The trigger system is armed by GPIB with INITIATE:IMMEDIATE. Because the source of the trigger has been specified as external, this command "readies" the analyzer for a trigger but it does not actually generate the trigger.

**Averaged Measurements**

Averaged measurements are complete when the average count is reached. The average count is reached when the specified number of individual measurements is combined into one averaged measurement result. Use synchronization to determine when the average count has been reached.

If the analyzer continues to measure and average the results after the average count is reached, use synchronization to determine when each subsequent measurement is complete.
Calibrating the PNA Using SCPI

There are several ways to calibrate the PNA using SCPI depending on your measurement needs. As from the Cal Wizard, you can perform a Guided Cal, Unguided Cal, or ECal. This topic explains the differences in these calibration choices when using SCPI commands.

- Guided Calibrations
- ECal
- Creating Cal Sets
- Applying Cal Sets and Cal Types
- Uploading Error Terms
- Unguided Cals and Calibration Classes

See SCPI Calibration Examples

Guided Calibrations

- These commands calibrate the ACTIVE channel. Activate a channel by selecting a measurement on the channel to be calibrated using Calc:Par:Select.
- Full 1,2,3,4-port SOLT and TRL calibrations - No response cals.
- All of the advanced calibration features (Thru method, specify DUT connectors and Cal kits for each port, port pairings).
- A Cal Set is applied to the channel and saved at the completion of a guided cal according to the preference setting SENS.CORR.PREF:CSET:SAVE

ECal

From the Cal Wizard or from a SCPI program, ECal is fast, accurate, and very repeatable. Unlike from the Cal Wizard, you can use SCPI to perform ECal using either the Guided or Unguided commands. The Unguided commands are easiest to use. However, the following situations require that you use the Guided commands.

- To maximize accuracy, all ECal calibrations on the PNA perform an Unknown Thru measurement of the ECal module Thru state IF the PNA model being used has 1 reference receiver per port. If your PNA does NOT have 1 reference receiver per port, use Guided ECal commands and specify a Thru method.
- If your ECal module connectors do NOT match the DUT connectors, and you choose not to perform a User Characterization, use Guided ECal commands and specify the Thru method.
ECAL Notes:

- When using either Guided or Unguided ECal commands under low power situations, use the Orientation settings. The Guided example shows the use of these commands. When using Unguided, they must appear before the Acquire command.

- The frequency range of the measurement must be within the range of the ECal module. Otherwise, the calibration will fail.

- Although we recently provided the command, you do NOT have to send the ECal module state command. The ECal algorithm switches ECal states automatically.

- All of these ECal choices are listed in the Programming Command Search function in this Help file.

See Using ECal to learn about all of the ECal features.

Creating Cal Sets

There are several ways to store guided cal data into a unique Cal Set. The following is probably the easiest. It does not require the name of an existing Cal Set and it allows you to name the Cal Set.

```
SENS:CORR:COLL:GUID:INIT  'start the cal with no cal set argument
'Select the cal
SENS:CORR:COLL:GUID:SAVE  'create cal set with auto-generated name or to cal register
SENS:CORR:CSET:NAME 'MyCalSet'   'name the current cal set.
```

Applying Cal Sets and Cal Types

A Cal Set is applied to the channel and saved at the completion of a guided cal according to the preference setting SENS:CORR:PREF:CSET:SAVE. When you select a Cal Set to apply to an uncalibrated channel, the PNA attempts to find the most comprehensive calibration type in the Cal Set and turn it ON. In addition, changing a measurement parameter (for example, from S11 to S21) will also initiate an attempt to apply the best Cal Type and turn correction ON.

There may be times when you do not want the most comprehensive Cal Type. For example, say there is a Full 2-port Cal Set applied, but there is only an S11 measurement displayed. If measurement speed is a concern, you can apply a Full 1-Port Cal Type from that same Cal Set and save time by not doing the extra background sweeps. Learn more about background sweeps.

If you change the measurement parameter, the PNA will reapply the Full 2-Port Cal Type.

See the SCPI and COM commands for Cal Sets and Cal Types.

Uploading Error Terms

There are two ways to upload error terms using SCPI: the old way and the recommended way. The old way will still works but requires a 'preference' setting.

The old way is this:
This technique, used in WinCal software, starts a calibration and immediately saves it without acquiring any standards. In PNA Rev 6.0, executing SAVE without acquiring data will return an error. To suppress the error and continue to use the above technique to upload error terms, send the following command to set the preference:

```
SENS:CORR:PREF:SIMCal 1
```

Or you can execute the script that is saved on the PNA at C:\Program Files\Agilent\Network Analyzer\System\wincal32.reg.

Setting this preference defeats some error checking when performing unguided cal using SCPI. This is not recommended unless needed for backward compatibility.

The recommended way is to upload error terms into a created or selected Cal Set:

```
SENS:CORR:CSET:CREATE or SENS:CORR:CSET:GUID
SENS:CORR:CSET:Data <term> <port> <port> <data>
SENS:CORR:CSET:SAVE
```

This method puts error terms into a Cal Set, outside of a Guided or Unguided calibration session. The Cal Set can then be applied at any time.

See SENS:CORR:CSET commands.

**Unguided Cals and Calibration Classes**

- Use Sens:Correction commands.
- 1-port, 2-port, Response.
- Can select 2 sets of standards.
- TRL is NOT recommended.

The following describes how to perform an unguided calibration using SCPI. The objective here is to make clear the relationship between the physical port on which a standard is being measured, the actual device in the cal kit, and the SCPI command used to acquire the device.

Calibration standards classes are ‘categories’ of standard types. To perform a 2 port calibration, the cal wizard requires the following types of standards to be measured:

**3 reflection standards on the forward port:**

- Class S11A typically an open
- Class S11B typically a short
- Class S11C typically a load

Likewise, 3 reflection standards are required for the reverse port:

- Class S22A typically an open
- Class S22B typically a short
- Class S22C typically a load

There is also a transmission standard that is measured in both directions:

- Class S21T typically a thru

The following illustrates the relationship between cal kit physical standards and calibration classes. Here is a list of the physical devices in my calibration kit.

Standard #1 = "3.5 mm male short"
Standard #2 = "3.5 mm male open"
Standard #3 = "3.5 mm male broadband load"
Standard #4 = "Insertable thru standard"
Standard #5 = "3.5 mm male sliding load"
Standard #6 = "3.5 mm male lowband load"
Standard #7 = "3.5 mm female short"
Standard #8 = "female to female characterized thru adapter"
Standard #9 = "0-2 Load"
Standard #10 = "Open"
Standard #11 = "Non-insertable thru"
Standard #12 = "3.5 mm female lowband load"
Standard #13 = "3.5 mm female sliding load"
Standard #14 = "3.5 mm female broadband load"
Standard #15 = "3.5 mm female open"

When you perform a calibration remotely using SCPI, you don’t specify the device number directly. Rather, you specify the class you want to measure. Each device in the calibration kit is assigned to a class. And since more than one device can be assigned to the same class, each class contains an ordered list of devices. The class assignments are set using the Advanced Modify Cal Kit dialog or the SCPI command:

```
SENS:CORR:COLL:CKIT:ORDER <class>, <std>, <std>, <std>, <std>, <std>, <std>, <std>, <std>
```

The 85052B kit used in the example program has the following standard list for each class: The list was obtained by issuing the corresponding SCPI query:

```
SENS:CORR:COLL:CKIT:OLIST1?  S11A = +2,+15,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST2?  S11B = +1,+7,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST3?  S11C = +6,+5,+3,+12,+13,+14,+0
SENS:CORR:COLL:CKIT:OLIST4?  S21T = +4,+8,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST5?  S22A = +2,+15,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST6?  S22B = +1,+7,+0,+0,+0,+0,+0
SENS:CORR:COLL:CKIT:OLIST7?  S22C = +6,+5,+3,+12,+13,+14,+0
```
SENS:CORR:COLL:OLIST8?  S12T = +4,+8,+0,+0,+0,+0,+0

When you perform the calibration, you acquire data by issuing the ACQuire command:

```
SENS:CORR:COLL:ACQ <class>[, <subst> ]
```

For example:

```
SENS:CORR:COLL:SFOR 1
SENS:CORR:COLL:ACQ STANA, SST2
```

The SFOR command tells the wizard to make the next acquisition in the forward direction. The ACQuire command specifies that we are measuring the 2nd device in the list for STANA. And since we are measuring SFORward, then STANA refers to class #1 or S11A. The list of devices for this class are specified in the OLIST1 query above.

Alternately, you could modify the device order for the S11A class to move device #15 into the first position (SENS:CORR:COLL:CKIT:ORDER1). When the desired device is in the first position, you need not specify the order number in the ACQuire command. The default is the first device in the OLIST. This works well for two port network analyzers where the order for S11A,B,C classes is set up for port 1 and the order for S22A,B,C is set up for port 2. With the kit set up in the proper order, you eliminate the need to specify the substandard number (SST<n>).

See an example: Perform an Un guided 2-port Cal on a 4-port PNA.
The PNA as a USB Device

Use of the PNA as a USB Device is not supported at this time.
Reading the Analyzer's Status Register

The PNA has several status registers that your program can read to know when specific events occur. There are two methods of reading the status registers in the analyzer: the Polled Bit method and the Service Request method.

- **Polled Bit Method**
- **Service Request Method**
- **Setting and Reading Bits in Status Registers**
- **Positive and Negative Transitions**
- **Status Commands**

### Other Topics about GPIB Concepts

Most of the status registers in the analyzer have sixteen bits. For simplicity, this topic will illustrate their use with 8-bit registers. Bits in registers represent the status of different conditions inside of the analyzer. In the following graphic, a register is represented by a row of boxes; each box represents a bit. Bit 3 is ON.

![Status Register Graphic](image)

### The Polled Bit Method

With the Polled Bit Method, your program **continually** monitors a bit in the status register that represents the condition of interest to you. When the analyzer sets the bit to 1, your program immediately sees it and responds accordingly.

**Advantage:** This method requires very little programming.

**Disadvantage:** This method renders your program unavailable to do anything other than poll the bit of interest until the condition occurs.

**Procedure:**

1. Decide which condition to monitor. The **Status Commands** topic lists all of the possible conditions that can be monitored in the analyzer.
2. Determine the command and the bit that will monitor the command.
3. Construct a loop to poll that bit until it is set.
4. Construct the routine to respond when the bit is set.

### The Service Request (SRQ) Method

Your program enables the bits in the status registers representing the condition of interest. When the condition occurs, the analyzer actively interrupts your program from whatever it is doing, and an event handler in your program responds accordingly. Do this method if you have several conditions you want to monitor or the conditions
are such that it is not practical to wait for the condition to occur.

**Advantage:** This method frees your program to do other things until the condition occurs. The program is interrupted to respond to the condition.

**Disadvantage:** This method can require extensive programming depending on the number and type of conditions that you want to monitor.

**Procedure:**

1. Decide which conditions to monitor. The Status Commands topic lists all of the possible analyzer conditions that can be monitored.

2. Set the enable bits in the **summary** registers and the **status byte** register.

   **Enabling** is like making power available to a light - without power available, the switch can be activated, but the light won't turn ON. In the analyzer, without enabling, the condition may occur, but the controller won't see it unless it is enabled.

   The condition, and the bit in the **summary** registers in the reporting path, must be enabled. This is like streams (conditions) flowing into rivers (summary registers), and rivers flowing into the ocean (controller). See the diagram of status registers in Status Commands.

   Bit 6 of the **status byte** register is the only bit that can interrupt the controller. When **any** representative bit in the status byte register goes ON, bit 6 is automatically switched ON.

3. Enable your program to interrupt the controller. This is done several ways depending on the programming language and GPIB interface card you use. An example program is provided showing how this is done with in Visual Basic with a National Instruments GPIB card.

4. Construct a subroutine to handle the interrupt event. If you are monitoring more than one condition in your system, your event handler must determine which condition caused the interrupt. Use the *SPE command to determine the instrument that caused the interrupt, then poll the summary registers, then poll condition registers to determine the cause of the interrupt.

---

### Setting and Reading Bits in Status Registers

Both methods for reading status registers requires that you read bits out of the status registers. Most of the PNA status registers contain 16 bits, numbered 0 to 15. Each bit has a weighted value. The following example shows how to set the bits in a 8-bit status register.

8-bit register

<table>
<thead>
<tr>
<th>Bit</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Weight</strong></td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>16</td>
<td>32</td>
<td>64</td>
<td>128</td>
</tr>
</tbody>
</table>

We want to set bits 4 and 5 in the Standard Event Status Enable register.

**Step** **Example**

1. Read the weighted bit value for these bits weights 16 and 32 (respectively)

2. Add these values together 16 + 32 = 48

3. Send this number as an argument in the appropriate command. (see Status Commands) STAT:QUES:LIMIT1:ENAB 48
**Positive and Negative Transitions**

Transition registers control what type of in a condition register will set the corresponding bit in the event register.

- **Positive** transitions (0 to 1) are only reported to the event register if the corresponding positive transition bit is set to 1.
- **Negative** transitions (1 to 0) are only reported to the event register if the corresponding negative transition bit is set to 1.
- Setting both transition bits to 1 causes both positive and negative transitions to be reported.

Transition registers are read-write and are unaffected by *CLS (clear status) or queries. They are reset to their default settings at power-up and after *RST and SYSTem:PRESet commands. The **following are the default settings** for the transition registers:

- All Positive Transition registers = 1
- All Negative Transition registers = 0

This means that, by default, the analyzer will latch all event registers on the negative to positive transition (0 to 1).

The following is an example of why you would set transition registers:

A critical measurement requires that you average 10 measurements and then restart averaging. You decide to poll the averaging bit. When averaging is complete, the bit makes a positive transition. After restart, you poll the bit to ensure that it is set back from 1 to 0, a negative transition. You set the negative transition bit for the averaging register.
PNA Supported Interfaces

The PNA supports the following interfaces for SICL / VISA communication:

- **LAN** - as a remote GPIB interface. The PNA LAN is presented as a virtual GPIB interface. It does NOT support simple TCPIP-based control. Therefore, when configuring the Agilent IO libraries on your PC, add a **REMOTE GPIB** interface, which uses the LAN client interface.

- **GPIB** - requires that your external controller have a GPIB card.

**Note:** For optimum LAN interface performance, use COM to control the PNA. SCPI commands can be sent to the PNA using the COM `SCPIStringParser` object.

The following interfaces are NOT supported:

- **USB**
- **Serial**
Important Note:
To enable VISA or SICL communication over LAN, you must do the following:

1. On the PNA, click **System**, point to **Configure**, then click **SICL/GPIB**.

2. Check **SICL Enabled**. To automatically enable SICL when the PNA is booted, check **Automatically enable on Startup**.

3. Click **OK**.

The PNA is now ready to be controlled over LAN.

Learn more about this dialog box.

Agilent I/O Libraries

The Agilent I/O libraries includes the drivers to allow you to communicate with Agilent test instruments. Every PNA is shipped with the Agilent I/O libraries installed. We recommend you do NOT upgrade the Agilent I/O libraries on the PNA as unexpected results may occur. If you choose to upgrade the Agilent I/O libraries on the PNA, do NOT change the default folder path in the InstallShield Wizard.

To communicate with the PNA, the Agilent I/O libraries must also be installed on your external controller. To purchase the Agilent I/O libraries, or download a free upgrade, go to www.agilent.com and search for IO Libraries. Scroll to find Software, Firmware & Drivers.

SICL / VISA Programs Running on the PNA

You can run your SICL / VISA program on the PNA to control the PNA. Although the Agilent I/O libraries are already installed on the PNA, it is configured as the **Host**. You must also configure a SICL or VISA LAN **Client** interface on the PNA, specifying the LAN hostname of that same PNA.

If your program uses the COM interface to VISA, and is compiled on a PC with the Agilent IO Libraries Suite (version 14 or later), and the resulting executable is copied and run on the PNA, it will produce a "type mismatch error". This is because the PNA has the ‘M’ version of Agilent I/O libraries. The following Visual Basic code is an example of how to avoid this error when communicating with the PNA from within the PNA:

```vbnet
Dim rm As IResourceManager
Dim fmio As IFormattedIO488
Set rm = CreateObject("AgilentRM.SRMCls")
Set fmio = CreateObject("VISA.BasicFormattedIO")
Set fmio.IO = rm.Open("GPIB0::22")
f mio.WriteString "*IDN?" & Chr(10)
MsgBox fmio.ReadString()
```

Controlling the PNA over LAN while controlling other instruments over GPIB

The PNA can NOT be both a controller and talker/listener on the same GPIB bus. Using SICL / VISA, you can use LAN to control the PNA, leaving the PNA free to use the rear-panel GPIB interface to control other GPIB devices.
Configure the PNA for SICL / VISA

1. On the PNA, click System then check Windows Taskbar

2. Click Start then point to Program Files, Agilent IO Libraries, then click IO Config

3. Select each GPIB Interface and click Edit to verify (or make) the default settings in the following table. These settings are REQUIRED when using a 82357A USB / GPIB Interface with the PNA.

4. When complete, click OK to close the edit dialog.

5. Click OK to close the IO Config dialog.

<table>
<thead>
<tr>
<th>VISA Interface Name</th>
<th>SICL Interface Name</th>
<th>Dialog box title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIB0</td>
<td>gpib0</td>
<td>GPIB Using NI-488.2</td>
<td>PNA Rear-panel GPIB connector. This GPIB interface can be used to control the PNA OR for the PNA to control external equipment. IT CAN NOT DO BOTH IN THE SAME PROGRAM. Learn more about pass-through options.</td>
</tr>
<tr>
<td>GPIB1</td>
<td>hpib7</td>
<td>Internal Instrument Configuration</td>
<td>Internal interface for programs running on the PNA to control itself.</td>
</tr>
</tbody>
</table>

Configure the External Controller

Please refer to the Agilent I/O libraries documentation to learn how to configure your controller to communicate with the PNA. These links can show you how to find the following PNA information:

- PNA full computer name
- GPIB Address
- IP Address

This example program can help test your VISA configuration.
Agilent VEE Pro RunTime Installed

Beginning in Dec. 2005, Agilent VEE Pro RunTime is installed on new PNAs. This means that programs written with Agilent VEE (.vxe files) can be run directly on the PNA. New PNA models with 1.1 GHz CPU have VEE 7.5 runtime with Agilent I/O Libraries suite 14.

PNAs without Agilent VEE installed can go to the Agilent VEE website and download Agilent VEE Pro 6.2 RunTime to the PNA and begin to run VEE programs directly on the PNA. This version does not require Agilent I/O Libraries suite 14. Do NOT upgrade to Agilent I/O libraries suite 14 on the PNA.

With Agilent VEE Pro RunTime installed on the PNA, the following examples can be run directly on the PNA:

- Basic Control of the PNA

For more VEE examples, see the PNA support website.

For more information on Agilent VEE, see www.agilent.com/find/VEE
Basic Control using VEE

This VEE Pro 6.0 example does the following:

- Controls PNA windows and traces.
- Changes stimulus settings.
- Measures all four S parameters.
- Create markers and displays marker readout.

If this Help file is on a PNA and VEE Pro RunTime is installed, then:

1. Run the BasicControl.vxe example
2. Then click Open on the following dialog box to run the program.

Otherwise, you can modify the example program using VEE, save the VEE BasicControl.vee
Learn how to run this program as a Macro on the PNA.

The following dialog box will be visible on the PNA when the example program is running.

- Click Fwd to activate the Forward (S11 and S21) measurements.
- Click Rev to activate the Reverse (S22 and S12) measurements.
- Click Update Markers to sweep the PNA.
- Type values to change Marker Frequencies.
ECal with Confidence Check using VEE

This VEE Pro 6.0 example performs an ECal and subsequent ECal confidence Check.

If this Help file is on a PNA and VEE Pro RunTime is installed:

- Run the .vxe example

- Then click Open on the following dialog box to run the program.

Or to modify the example program using VEE, save the VEE BasicControl.vee

Learn how to run this program as a Macro on the PNA.

The following dialog box will be visible on the PNA when the example program is running.

- Click Fwd to activate the Forward (S11 and S21) measurements.

- Click Rev to activate the Reverse (S22 and S12) measurements.

- Click Update Markers to sweep the PNA.

- Type values to change Marker Frequencies.
Interface Control

The Interface Control feature allows you to send remote commands and data to the following PNA rear-panel Interfaces: GPIB, Material Handler I/O, Test Set I/O, and Auxiliary I/O.

- **Overview**
- **How to Access Interface Control Settings**
- **Interface Control Dialog Box**
- **Z5623A H08 Test Set Commands**

**Other External Device Control Topics**

**Overview**

The Interface Control feature allows you to send data to control external equipment such as GPIB instruments, a material handler, test set, or other equipment, without needing to create a remote program. The PNA manages the timing and required interface setup. See **Rear Panel Tour**.

- A unique set of control data can be sent for each channel. In addition, a unique set of control data can be sent before the channel sweep starts, and after the sweep ends.
- Interface Control settings can be saved and recalled from the **Interface Control dialog box**, or with **Instrument State Save and Recall**.
- Interface Control settings can be copied to other channels using **Copy Channels**.
- Control data can only be WRITTEN to the interfaces, NOT READ from the interfaces.
- Control data is sent in the following order. This order cannot be changed.

1. **GPIB Interface**
2. **Material Handler Interface**
3. **Test Set Interface**
4. **Aux Interface**
5. **Dwell Time**
## How to access Interface Control settings

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Channel</td>
</tr>
<tr>
<td></td>
<td>2. then Interface Control</td>
</tr>
</tbody>
</table>

| **For PNA-X and 'C' models**                |                              |
| 1. Press TRACE/CHAN                         | 1. Click Trace/Chan           |
| 2. then [Channel]                           | 2. then Channel              |
| 3. then [More]                              | 3. then More                 |
| 4. then [Interface Control]                 | 4. then Interface Control    |
An Instrument Preset will reset all of the fields to their default settings.

**Note:** If an error is encountered when sending Interface Control data, an error message is displayed on the PNA screen. The Channel Trigger State is set to Hold. You must fix the condition that caused the error, then change the Channel Trigger State to its original setting.

**Enable Interface Control** Enables and disables ALL Interface Control communication. When cleared (default setting) Interface Control is disabled and NO data is sent. To send data, the individual interfaces must also be enabled.

**Channel** Specifies the channel number for dialog settings. Each channel is configured individually. The list box shows the channels that currently have measurements. There must be at least one measurement present in order to make settings.

**Channel Label** Specifies the label to be displayed on the second status bar at the bottom of the PNA screen. This field is shared with External Testset control. The second status bar is automatically displayed when Interface Control is enabled.

Learn about the primary status bar.
Before Sweep Start - After Sweep End Tabs

Commands / data for all four interfaces can be sent both Before Sweep Start and After Sweep End. However, they are configured and enabled on separate tabs of the Interface Control dialog box. For example, to send GPIB commands both Before and After a PNA sweep, the Enable Control checkbox must be selected and commands entered on BOTH the Before Sweep Start and After Sweep End tabs.

Before Sweep Start  The data is sent BEFORE the first trace on the channel begins sweeping.

After Sweep End  The data is sent AFTER the last trace on the channel completes sweeping.

GPIB Commands

Notes:

- While using the rear-panel GPIB port with Interface Control, the PNA must be in GPIB System Controller mode. If the PNA is NOT in System Controller mode, an error message appears AND Interface Control is disabled. To correct this situation,
  
  1. Put the PNA in System Controller mode, and
  2. Re-enable Interface Control.

- GPIB instruments CAN be connected to the PNA using a USB/GPIB adapter. In this case, the PNA can be in talker-listener mode.

- GPIB Queries are NOT supported.

Enable Control  Enables and disables sending commands out the GPIB interface.

Multi-line edit control  Each line contains a GPIB command using the following syntax:

<table>
<thead>
<tr>
<th>address</th>
<th>command</th>
</tr>
</thead>
</table>

Where:

  - address  a number between 0 and 31. The PNA will look through all of the GPIB interfaces for an instrument connected to the specified address. If an instrument with that address is not recognized, an error is returned.

  - command  a GPIB command, with or without enclosing quotes. Enclosing quotes are ignored.

Address and command are separated by at least one space.

Commands should be separated by a new line, or carriage return. For example:

| 19 "init:cont off" |
| 16 init:imm |

The front-panel Enter key inserts a new line into the field.

The number of GPIB commands that can be entered is limited only by the available memory of the PNA.

See Z5326A H08 Test Set Commands.
**Material Handler I/O**

**Enable Control** Enables and disables sending data out the Material Handler I/O connector.

**Ports A, B, C, D** Sends values to the respective Handler I/O port. Although ports C and D are normally bidirectional, ONLY Output mode is allowed using the Interface Control feature. It cannot read from these, or any other, ports.

---

**Test Set I/O**

Note: The PNA has a separate interface for controlling the E5091A Test Set.

**Enable Control** Enables and disables sending data out the External Test Set I/O connector.

**Multi-line edit control** Each line contains a Write command using the following syntax:

```
address.value
```

Where:

- **address** any positive integer.
- **value** numeric character. Entries that require alpha characters should use the GPIB interface.

Address and value are separated by a period. For example:

```
18.2
27.3
```

Entries should be separated by a new line, or carriage return. The PNA front-panel Enter key inserts a new line into the field.

All entries are sent out the Test Set I/O port using the **WriteData Method**.

The number of entries is limited only by the available memory of the PNA.

---

**Aux I/O**

**Enable Control** Enables and disables sending data out the Auxiliary I/O connector.

**DAC1, DAC2** Sets voltages on the Aux I/O connector pins 2 (DAC1) and pin 3 (DAC2).

**Dwell After Command** Specifies a wait time, in milliseconds, after all commands to all interfaces are sent. Any positive integer is allowed. This is used to allow all external devices to settle before beginning a measurement. An erratic trace could indicate that more settling time is necessary.

**Reset All** Sets ALL fields on ALL channels to their default values.

**Save and Recall** Saves and recalls the contents of this dialog box. If the Interface Control dialog box is populated with settings during an Instrument State Save, the settings are automatically recalled with the Instrument State settings.

Interface control uses an *.xml file type. An example file is stored on the PNA hard drive. You can recall it into the dialog, or you can open and edit it with a word processor, such as Word Pad.

**OK** Applies the settings and closes the dialog box.

**Cancel** Does not apply changes that were made, and closes the dialog box.
The following table lists the commands that are used to control the popular Agilent Z5623A H08 Test Set. These commands can be entered into the GPIB Interface control.

<table>
<thead>
<tr>
<th>Connection Path</th>
<th>Test Set Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reflection to Port 1</td>
<td>refl_01</td>
</tr>
<tr>
<td>Reflection to Port 2</td>
<td>refl_02</td>
</tr>
<tr>
<td>Reflection to Port 3</td>
<td>refl_03</td>
</tr>
<tr>
<td>Reflection to Port 4</td>
<td>refl_04</td>
</tr>
<tr>
<td>Reflection to Port 5</td>
<td>refl_05</td>
</tr>
<tr>
<td>Reflection to Port 6</td>
<td>refl_06</td>
</tr>
<tr>
<td>Reflection to Port 7</td>
<td>refl_07</td>
</tr>
<tr>
<td>Reflection to Port 8</td>
<td>refl_08</td>
</tr>
<tr>
<td>Transmission to Port 1</td>
<td>tran_01</td>
</tr>
<tr>
<td>Transmission to Port 2</td>
<td>tran_02</td>
</tr>
<tr>
<td>Transmission to Port 3</td>
<td>tran_03</td>
</tr>
<tr>
<td>Transmission to Port 4</td>
<td>tran_04</td>
</tr>
<tr>
<td>Transmission to Port 5</td>
<td>tran_05</td>
</tr>
<tr>
<td>Transmission to Port 6</td>
<td>tran_06</td>
</tr>
<tr>
<td>Transmission to Port 7</td>
<td>tran_07</td>
</tr>
<tr>
<td>Transmission to Port 8</td>
<td>tran_08</td>
</tr>
<tr>
<td>Reset</td>
<td>*rst</td>
</tr>
<tr>
<td>Reflection Termination</td>
<td>*r_term</td>
</tr>
<tr>
<td>Transmission Termination</td>
<td>*t_term</td>
</tr>
<tr>
<td>All Termination</td>
<td>*all_term</td>
</tr>
</tbody>
</table>
Auxiliary I/O Connector

General Description

This DB-25 male connector provides a variety of analog I/O, digital I/O, timing I/O, and supply lines. You can change the settings on the Auxiliary IO connector through SCPI and COM programming commands. The settings are NOT accessible through the front-panel keys or display menu.

**Note:** This connector does NOT exist on the PNA-X model.

**Note:** The AUX IO configuration settings REMAIN after an Instrument Preset and Hibernation. However, Preset will clear the DAC values. The settings will revert to their default settings ONLY after the PNA is restarted, or until they are changed by you. AUX IO settings are saved and recalled with Instrument State.

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ACOM</td>
<td>Ground reference for analog signals</td>
</tr>
<tr>
<td>2</td>
<td>Analog Out 2</td>
<td>-10 to +10Vdc output, 10mA max</td>
</tr>
<tr>
<td>3</td>
<td>Analog Out 1</td>
<td>-10 to +10Vdc output, 10mA max</td>
</tr>
<tr>
<td>4</td>
<td>no connect</td>
<td>for future enhancements</td>
</tr>
<tr>
<td>5</td>
<td>DCOM</td>
<td>Ground reference for digital signals</td>
</tr>
<tr>
<td>6</td>
<td>reserved</td>
<td>for future enhancements</td>
</tr>
<tr>
<td>7</td>
<td>reserved</td>
<td>for future enhancements</td>
</tr>
<tr>
<td>8</td>
<td>reserved</td>
<td>for future enhancements</td>
</tr>
<tr>
<td>9</td>
<td>+5V</td>
<td>+5Vdc output, 100mA max.</td>
</tr>
<tr>
<td>10</td>
<td>Pass/Fail Write Strobe</td>
<td>Indicates pass/fail line is valid (active low)</td>
</tr>
<tr>
<td>11</td>
<td>Sweep End</td>
<td>Indicates sweep is done (programmable modes)</td>
</tr>
<tr>
<td>12</td>
<td>Pass/Fail</td>
<td>Indicates pass/fail (programmable logic, modes and scope)</td>
</tr>
<tr>
<td>13</td>
<td>Output Port Write Strobe</td>
<td>Writes I/O port data (active low)</td>
</tr>
<tr>
<td>14</td>
<td>Analog In</td>
<td>-10 to +10VDC analog input</td>
</tr>
<tr>
<td>15</td>
<td>ACOM</td>
<td>Ground reference for analog signals</td>
</tr>
<tr>
<td>16</td>
<td>Power Button In</td>
<td>Grounding replicates front panel power button press</td>
</tr>
<tr>
<td></td>
<td><strong>DCOM</strong></td>
<td>Ground reference for digital signals</td>
</tr>
<tr>
<td>---</td>
<td>----------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>18</td>
<td><strong>Ready for Trigger</strong></td>
<td>Indicates ready for external trigger (active low)</td>
</tr>
<tr>
<td>19</td>
<td><strong>External Trigger In</strong></td>
<td>Measurement trigger input (programmable to be active high or low)</td>
</tr>
<tr>
<td>20</td>
<td><strong>Footswitch In</strong></td>
<td>Active low input latches a user-readable status bit.</td>
</tr>
<tr>
<td>21</td>
<td><strong>+22V</strong></td>
<td>+22Vdc output, 100mA max.</td>
</tr>
<tr>
<td>22</td>
<td><strong>In/Out port C0</strong></td>
<td>General purpose input / output</td>
</tr>
<tr>
<td>23</td>
<td><strong>In/Out port C1</strong></td>
<td>General purpose input / output</td>
</tr>
<tr>
<td>24</td>
<td><strong>In/Out port C2</strong></td>
<td>General purpose input / output</td>
</tr>
<tr>
<td>25</td>
<td><strong>In/Out port C3</strong></td>
<td>General purpose input / output</td>
</tr>
</tbody>
</table>

**ACOM (pins 1, 15)**

**Description**
Analog common (ground) - To be used with the Analog Out and Analog In lines.
ACOM and DCOM are connected to system ground at a star ground point inside the analyzer.

**Analog Out 1, 2 (pins 2, 3)**

**Description**
Two analog outputs programmable to +/-10V; I_{out}<10mA; Rout=100 ohms
12-bit DACs with voltage resolution of approximately 5mV/count.
The DACs are set to constant values using [SCPI](#) or [COM](#), and can be read using [SCPI](#) or [COM](#) commands.
Preset state for both pins is 0 volts.

**HW Details**
Looking into this output pin is a 100-ohm series resistor followed by two diodes tied to +/-15V for static protection, then the output or an op-amp.
The voltage output is provided by a 12-bit DAC with an op amp buffer.

Specifics:
- Maximum output current = 10mA
- Settling time = 3us

**Timing**
The DACs are set after the last data point is measured, during retrace. If the analyzer is in single sweep mode, the DACs are set as part of the presweep process, before the sweep is triggered.
DCOM (pins 5, 17)

*Description*

Digital common (ground).

Used with the digital input and output lines.

ACOM and DCOM are connected to system ground at a star ground point inside the analyzer.

---

**Pins 6, 7, 8**

*Description*

Reserved

---

+5V (pin 9)

*Description*

+5V nominal output (100mA max).

Protected by self-healing fuse:

---

**Pass/Fail Write Strobe (pin 10)**

*Description*

See [Handler IO connector](#).

---

**Sweep End (pin 11)**

*Description*

See [Handler IO connector](#).

---

**Pass/Fail (pin 12)**

*Description*

See [Handler IO connector](#).

---

**Output Port Write Strobe (pin 13)**

*Description*

See [Handler IO connector](#).

---

**Analog In (pin 14)**

*Description*

Analog input, +/-10V range, Rin=100k ohm

Bandwidth = 40kHz (2-pole lowpass filter).

This analog input may be read using the [SCPI](#) or [COM](#) commands.

**HW Details**

---

2539
Looking into this pin there is a 1k-ohm series resistor followed by a 100k-ohm resistor to ground, static protection diodes after the 1k resistor limit the signal to +/-15V, then a high impedance buffer and active filter limiting the bandwidth to 40kHz with a lowpass filter.

**Power Button In (pin 16)**

**Description**
Short this pin to ground to replicate a front panel power button key press.

**HW Details**
Looking into the pin there is a 215-ohm series resistor followed by a 10k pull-up to the 3V standby supply, static protection diodes to 0V/5V and then connects to the front panel power key circuit.

**CAUTION:** Because this line is internally pulled up to 3V, it should not be driven by a TTL driver.

**Timing**
Grounding this line for 1us to 2 seconds will simulate pressing the front panel power button.

Grounding this line for >4 seconds will perform a hard reset (similar to a personal computer) and is not recommended.

**Ready for Trigger (pin 18)**

**Description**
TTL output.
Active Low signal indicates that system is ready for an external trigger.
Remains High if system is not in External Trigger mode.
Goes High after an External Trigger is acknowledged.
Goes Low after the system has finished with its measurements, the source has been set up, and the next data point is ready to be measured.

**HW Details**
Looking into this pin there is a 215-ohm series resistor followed by a 10k pullup, diodes to 0V/5V for static protection, then the output of an "ABT" TTL buffer.

This line is enabled only when the analyzer is in External Trigger mode.
Refer to External Trigger In (following pin) for more information.

**Timing**
Refer to External Trigger In (following pin)

**External Trigger In (pin 19)**

**Description**
This input accepts level trigger signals (High / Low) on all PNA models, or edge trigger signals on some PNA models.

The external trigger configuration is set from the front panel, **SCPI** or **COM**.

For more information, see **External triggering**.
A single trigger is achieved by asserting the external trigger for a period from 1us to 50us. Continuous triggering is achieved by holding the external trigger in the "asserted" mode (either Low or High).

**HW Details**

Looking into this pin is a 215-ohm series resistor followed by a 4.64k pullup, 1000pF to ground and then a "FAST" TTL buffer input.

**Timing**

A level trigger width should be between 1us and 50us.

---

A Low level input such as shorting this line to ground using a footswitch (where the input stays low for >1us) will be latched.

The latched status may be read using the SCPi or COM commands.

Only one footswitch press can be latched (remembered) by the system.

Reading the latch status will reset it if Footswitch In has returned to a high level.

**HW Details**

Looking into this pin is a 215-ohm series resistor followed by a 4.64k pullup to 5V and 1000pF to ground. This line is an input to a "FAST" TTL buffer.

**Timing**

Footswitch In must be Low for at least 1us.

---

**+22V (pin 21)**

**Description**

+22V nominal output (100mA max).

Protected by self-healing fuse.

---

**In/Out Port C0-C3 (pins 22-25)**

**Description**

See Handler IO connector
External Test Set I/O Connector

General Description
This DB-25 female connector is used to control external test sets. The external test set bus consists of 13 multiplexed address and data lines, three control lines, and an open-collector interrupt line. The Test Set IO is not compatible with the 8753 test sets.

You can change the settings on the External Test Set IO connector through SCPI and COM programming commands. The settings are NOT accessible through the front-panel keys or display menu.

Notes:
- The External Test Set pin settings are NOT affected by Instrument State Save/Recall or Instrument Preset.
- At PNA Power Up and return from Hibernation, the External Test Set bus data lines, address lines, and control lines are set HIGH, and no strobe lines are pulsed.

Caution: Do not mistake this connector with a Parallel Printer port. A printer may be damaged if connected to this port.

Other External Device Control Topics

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SEL0</td>
<td>Test set select bit 0; tied to GND</td>
</tr>
<tr>
<td>2</td>
<td>Sweep Holdoff</td>
<td>TTL input - state may be read with SCPI or COM command</td>
</tr>
<tr>
<td>3</td>
<td>AD12</td>
<td>Address and latched data</td>
</tr>
<tr>
<td>4</td>
<td>AD10</td>
<td>Address and latched data</td>
</tr>
<tr>
<td>5</td>
<td>AD9</td>
<td>Address and latched data</td>
</tr>
<tr>
<td>6</td>
<td>AD8</td>
<td>Address and latched data</td>
</tr>
<tr>
<td>7</td>
<td>GND</td>
<td>0V</td>
</tr>
<tr>
<td>8</td>
<td>LAS</td>
<td>TTL output  Low = Address Strobe</td>
</tr>
<tr>
<td>No.</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>AD4 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>AD3 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>AD2 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>GND 0V</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Interrupt In TTL input - state may be read with a SCPI or COM command</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>No connect CAUTION: Older PNAs have +22v on this line; this will damage a printer.</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>SEL1 Test set select bit 1; tied to GND</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>SEL2 Test set select bit 2; tied to GND</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>AD11 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>SEL3 Test set select bit 3; tied to GND</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>AD7 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>AD6 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>AD5 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>AD0 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>AD1 Address and latched data</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>LDS TTL output - active low data strobe</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>RLW TTL output - high-read, low write</td>
<td></td>
</tr>
</tbody>
</table>

**SEL0-SEL3 (pins 1,15,16,18)**

**Description**
Selects addresses of test sets that are "daisy chained" to this port. The select code is set to zero at the PNA connector and is incremented by one as it goes through each successive external test set. Therefore, the first test set in the chain has address zero and so on, for up to 16 test sets.

**HW Details**
Connected to ground inside the PNA.

**Timing**
None

**Sweep Holdoff In (pin 2)**

**Description**
Input line used by the test set for holding off a sweep. Holding off a sweep is one way of introducing a delay that allows an external device to settle before the PNA starts taking data. You must write a program that will query the line and perform the delay. The program needs to query the line and keep PNA from sweeping while the line remains low. When a subsequent query detects that the line went high the program would then trigger the PNA to start the sweep.

Use either Single or External trigger mode to control the PNA sweep.

**HW Details**

This pin has a series 215-ohms resistor followed by 4.7k-ohm pull-up and then an "ABT" TTL buffered register.

**Timing**

This input is not latched by the PNA hardware. Therefore the input level must be held at the desired state by the test set until it’s read by your program.

---

**AD0-AD12 (pins 3-6, 9-11, 17, 19-23)**

**Description**

Thirteen lines are used to output data addresses or input / output data. Several SCPI and COM commands are available for reading and writing to these lines. You can choose to use commands where the PNA provides the appropriate timing signals needed for strobing the addresses and data. Or you can choose to control the timing signal directly. The timing signals are RLW, LAS and LDS. If you decide to do direct control refer to the corresponding SCPI and COM command details. Close attention to detail is needed to insure the desired results.

After a write command, lines AD0-AD12 are left in the state they were programmed. Default setting for Mode is Read / Input).

After a read command, lines AD0-AD12 are left in input mode. While in this mode an external test set attached to the IO is free to set the level on each line.

**HW Details**

Each of these I/O pins has a series 215-ohm resistor followed by 4.7k-ohm pull-up resistor.

Write/Read is implemented by an output tri-state TTL buffer / latch for latching and enabling write data in parallel with a TTL input buffer for reading.

**Timing**

Output Address and data setup and hold times are 1us minimum.

---

### Address & Data I/O Write

![Timing Diagram](image)

2544
### GND (pins 7, 12)

**Description**

Two ground pins used as ground references by the test set.

**HW Details**

Connected to digital ground.

**Timing**

None.

### LAS (Low Address Strobe) (pin 8)

**Description**

This line has two behaviors that are command dependent. Refer to the [SCPI](#) and [COM](#) commands for further details.

In one behavior LAS is one of the lines used by the PNA to provide appropriate timing for writing Address and Data to the Test Set. In this case LAS is controlled automatically by the PNA and is intended to be used as the strobe for the Address. When LAS is low, lines AD0 - AD12 represent the Address. LAS will return to its normally high state when the transaction is finished.

In the second behavior the PNA will NOT provide appropriate timing. In this case LAS is controlled directly by the user through a [SCPI](#) or [COM](#) command. When the transaction is finished LAS is left set to the state it was programmed to until another command changes it. (Default for LAS is TTL High).

**HW Details**

This output pin is driven by a TTL latched buffer with a series 215-ohm resistor followed by 2.15k-ohm pull-up.

**Timing**

Strobe length, setup and hold times are all 1us minimum.

See the description for AD0-AD12 for more timing information.

---

### Interrupt In (pin 13)

**Description**

Query this line with a [SCPI](#) or [COM](#) command.

**HW Details**
This line is a non-latched TTL input, has series 215-ohms followed by 4.64k-ohm pullup.

**Timing**
The Test Set must maintain at the desired TTL level until its read.

__(pin 14) No Connect (previously +22V)__

**WARNING**: Early versions of the PNA had +22v on this pin. Connecting a printer to this port will usually damage the printer.

**Description**
+22V, 100mA max. The 25-pin D connector is the same as a computer parallel printer port connector. Pin (14) corresponds to a printer's "autofeed" line. Connecting a printer to this port will damage the printer if +22v is present since printers requires less than 5V on all control lines.

**HW Details**
No connect

**Timing**
None

---

**LDS (Low Data Strobe) (pin 24)**

**Description**
This line has two behaviors that are command dependent. Refer to the External Test Set IO SCPI and COM commands for further details. (Default setting for LDS is TTL High)

In one behavior LDS is one of lines used by the PNA to provide appropriate timing for writing Address and Data to the Test Set. In this case LDS is controlled automatically by the PNA and is intended to be used as the strobe for the Data. When LDS is low, lines AD0 - AD12 represents Data. LDS will return to its normally high state when the transaction is finished.

In the second behavior the PNA will NOT provide appropriate timing. In this case LDS is controlled directly by the user through a SCPI or COM command. When the transaction is finished the LDS is left set to the state it was programmed to.

**HW Details**
This output pin is driven by a TTL latched buffer with a series 215-ohm resistor followed by 2.15k-ohm pull-up.

**Timing**
Strobe length, setup and hold times are all 1us minimum.

See the description for AD0-AD12 for more timing information.

---

**RLW (pin 25)**

**Description**
This line is the output for the Read Write signal. It has two behaviors that are command dependent. Refer to the External Test Set IO SCPI and COM commands for further details. (Default setting for RLW is TTL High)
In one behavior RWL is controlled automatically by the PNA during a Read Write operation. When RLW is low, lines AD0 - AD12 represent output Data. When RLW is high, the lines represent input Data.

In the second behavior the PNA does NOT provide the timing. The user must control it directly through the SCPI or COM command. In this case the line is left set to the state it was programmed to.

**HW Details**

This pin is a TTL latched output with a series 215-ohm resistor followed by 2.15k-ohm pull-up resistor.

**Timing**

Strobe length, setup and hold times are all 1us minimum.

See the description for **AD0-AD12** for more timing information.
Material Handler I/O Connector

This [rectangular 36-pin female connector](#) provides communication signals between the PNA and a material parts handler. You can change the settings on the Material Handler IO connector using [SCPI](#) and [COM](#) commands. The settings are NOT accessible through the front-panel keys or display menu.

- **Overview - Controlling a Material Handler**
- **Pin Assignments**
- **Pin Descriptions**
- **Timing Diagrams**
- **Input Output Electrical Characteristics**

**Note:** On early PNAs this connector is labeled "GPIO". It is covered to indicate that the connector is not functional.

**Overview - Controlling a Material Handler**

The PNA is capable of interacting with an external material handler or part handler. This allows the PNA to be used in an automated test environment, where devices to be tested are inserted into a test fixture by a part handler, and sorted into pass/fail bins by the handler after testing is complete. By connecting the part handler to the PNA Auxiliary or Material Handler I/O ports, the PNA and part handler can synchronize their activities in a way that makes automated testing possible.

**PNA and Part Handler Preparation**

1. **Define the measurements** you want to make.
2. **Define limits** for each of the measurements.
3. Configure the PNAs Material Handler port so that it is compatible with your part handler. This usually involves setting the [handler logic](#), [pass/fail logic](#), [pass/fail scope](#), and [pass/fail mode](#). These settings are made remotely using [SCPI](#) or [COM](#) commands.
4. Use a cable to connect the PNA to your part handler.
5. Put the PNA in [External Trigger](#) mode.
6. Load parts in handler per manufacturer instructions.

**Note:** The Material Handler configuration settings REMAIN after an [Instrument Preset](#) and [Hibernation](#). The settings will revert to their default settings ONLY after the PNA is restarted, or until they are changed by you. Material Handler settings are saved and recalled with [Instrument State](#).

**Flow Diagram**

The following diagram and descriptions summarizes the events that occur during automated testing. 'DUT' refers to Device Under Test.
0. (Optional). The PNA sends values out the Material Handler and/or Auxiliary I/O connectors to configure external instruments. The A,B,C, and D ports of the Material Handler can be used to control devices used in testing, such as step attenuators, part handlers, or even the DUT itself. Also, the DAC1 and DAC2 lines on the Aux I/O connector can be used to provide bias voltages for devices and instruments. If you wish to use the Material Handler or Aux connectors for testing, you will need to write a program to send values out the various lines and ports, as there is no activity on these lines by default.

1. The part handler receives a **Ready for Trigger** signal from the PNA. This indicates that the PNA is properly configured and ready to take a measurement.

2. The part handler sends an **External Trigger** signal to the PNA. This signals that the part handler has settled, and allows the PNA to begin taking measurements.

3. The PNA takes measurements on all triggerable channels.

4. The **Index line** on the material handler goes to a Low state, which means that all required data has been collected by the PNA.

5. The part handler removes the DUT from the test fixture, and inserts a new DUT into the fixture. This operation is often referred to as part handler indexing. The device just tested is staged (removed from the fixture and prepared for binning), and the next part to be tested is put into the fixture. The removed DUT cannot be assigned to a Pass/Fail bin yet, as the Pass/Fail status is not available.

6. The PNA sends the **Pass/Fail Status**.

7. The PNA sends the **Pass/Fail Strobe** meaning that the Pass/Fail status has been determined.
8. The part handler reads the Pass/Fail Status line.

9. The part handler bins the staged part based on the Pass/Fail Status.

10. The test process repeats at step 1, waiting for Ready for Trigger from the PNA.

**Material Handler IO Pin Assignments**

There are three different Handler IO pin assignment configurations depending on the PNA model:

- **Type 1** - All E835xA, E880xA, N338xA models. You can change the pinout configuration to Type 2 on these models. This requires opening the instrument and changing a connector internally. Refer to the procedure in the Service Guide, Chapter 7. You can download the Service Guide for your PNA model from our website: [http://na.tm.agilent.com/pna](http://na.tm.agilent.com/pna)  **Caution:** Changing this connection should be done by qualified service personnel.

- **Type 2** - E8362x, E8363x, E8364x and N5230A Opts 220, 225, 420, 425, 520, 525;

- **Type 3** - All N5242A (PNA-X) and N5230 Opts 020, 025, 120, 125, 140, 145, 146, 245, 240, 245, 246.

See PNA models and options.

**Shaded/bold** indicates changes from Type 1

**Note:** A slash (/) preceding the signal names indicates that the signal uses negative (active low) logic. A low pulse is a logical 1.

<table>
<thead>
<tr>
<th>Pin</th>
<th>Type 1</th>
<th>Type 2</th>
<th>Type 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ground</td>
<td>Ground</td>
<td>Ground</td>
</tr>
<tr>
<td>2</td>
<td>/ INPUT1</td>
<td>/ INPUT1</td>
<td>/ INPUT1</td>
</tr>
<tr>
<td>3</td>
<td>/ OUTPUT1</td>
<td>/ OUTPUT1</td>
<td>/ OUTPUT1</td>
</tr>
<tr>
<td>4</td>
<td>/ OUTPUT2</td>
<td>/ OUTPUT2</td>
<td>/ OUTPUT2</td>
</tr>
<tr>
<td>5</td>
<td>/ Output port A0</td>
<td>/ Output port A0</td>
<td>/ Output port A0</td>
</tr>
<tr>
<td>6</td>
<td>/ Output port A1</td>
<td>/ Output port A1</td>
<td>/ Output port A1</td>
</tr>
<tr>
<td>7</td>
<td>/ Output port A2</td>
<td>/ Output port A2</td>
<td>/ Output port A2</td>
</tr>
<tr>
<td>9</td>
<td>/ Output port A4</td>
<td>/ Output port A4</td>
<td>/ Output port A4</td>
</tr>
<tr>
<td>10</td>
<td>/ Output port A5</td>
<td>/ Output port A5</td>
<td>/ Output port A5</td>
</tr>
<tr>
<td>No.</td>
<td>Description</td>
<td>Description</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>---------------------------------------</td>
<td>---------------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>11</td>
<td>Output port A6</td>
<td>Output port A6</td>
<td>Output port A6</td>
</tr>
<tr>
<td>12</td>
<td>Output port A7</td>
<td>Output port A7</td>
<td>Output port A7</td>
</tr>
<tr>
<td>13</td>
<td>Output port B0</td>
<td>Output port B0</td>
<td>Output port B0</td>
</tr>
<tr>
<td>14</td>
<td>Output port B1</td>
<td>Output port B1</td>
<td>Output port B1</td>
</tr>
<tr>
<td>15</td>
<td>Output port B2</td>
<td>Output port B2</td>
<td>Output port B2</td>
</tr>
<tr>
<td>16</td>
<td>Output port B3</td>
<td>Output port B3</td>
<td>Output port B3</td>
</tr>
<tr>
<td>17</td>
<td>Output port B4</td>
<td>Output port B4</td>
<td>Output port B4</td>
</tr>
<tr>
<td>18</td>
<td>no connect</td>
<td>Output port B5</td>
<td>Ext. Trigger</td>
</tr>
<tr>
<td>19</td>
<td>Output port B5</td>
<td>Output port B6</td>
<td>Output port B5</td>
</tr>
<tr>
<td>20</td>
<td>Output port B6</td>
<td>Output port B7</td>
<td>Output port B6 -or Index Signal Learn more</td>
</tr>
<tr>
<td>21</td>
<td>Output port B7</td>
<td>In/Out port C0</td>
<td>Output port B7 or Ready for Trigger Learn more</td>
</tr>
<tr>
<td>22</td>
<td>In/Out port C0</td>
<td>In/Out port C1</td>
<td>In/Out port C0</td>
</tr>
<tr>
<td>23</td>
<td>In/Out port C1</td>
<td>In/Out port C2</td>
<td>In/Out port C1</td>
</tr>
<tr>
<td>24</td>
<td>In/Out port C2</td>
<td>In/Out port C3</td>
<td>In/Out port C2</td>
</tr>
<tr>
<td>25</td>
<td>In/Out port C3</td>
<td>In/Out port D0</td>
<td>In/Out port C3</td>
</tr>
<tr>
<td>26</td>
<td>In/Out port D0</td>
<td>In/Out port D1</td>
<td>In/Out port D0</td>
</tr>
<tr>
<td>27</td>
<td>In/Out port D1</td>
<td>In/Out port D2</td>
<td>In/Out port D1</td>
</tr>
<tr>
<td>28</td>
<td>In/Out port D2</td>
<td>In/Out port D3</td>
<td>In/Out port D2</td>
</tr>
<tr>
<td>29</td>
<td>In/Out port D3</td>
<td>Port C Status</td>
<td>In/Out port D3</td>
</tr>
<tr>
<td>30</td>
<td>Port C Status</td>
<td>Port D Status</td>
<td>Port C Status</td>
</tr>
<tr>
<td>31</td>
<td>Port D Status</td>
<td>Output Port Write Strobe</td>
<td>Port D Status</td>
</tr>
<tr>
<td>32</td>
<td>Output Port Write Strobe</td>
<td>no connect</td>
<td>Output Port Write Strobe</td>
</tr>
<tr>
<td>33</td>
<td>Pass/Fail</td>
<td>Pass/Fail</td>
<td>Pass/Fail</td>
</tr>
<tr>
<td>34</td>
<td>Sweep End</td>
<td>+5V</td>
<td>Sweep End</td>
</tr>
</tbody>
</table>
**Pin Descriptions**

**Input1**
When this Input line receives a Low pulse from the material handler, data is latched on the OUTPUT1 and OUTPUT2 lines. See OUTPUT1|2 Data Output Write Timing

**Note: Type 1 and Type 2 Behavior:** The Input line responds to a High (rising edge) pulse.

The Input Line activity can be read:

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:INPut?</td>
<td>get_Input1 Method</td>
</tr>
</tbody>
</table>

The current state of these latched TTL outputs may be set High or Low (Default setting) using the (non-user) SCPI put_Output (COM) commands.

The next state (following a negative edge on the INPUT1 line) may be pre-loaded to High or Low (Default setting) using the user commands.

For example, on the next negative pulse on the INPUT1 line, you want the OUTPUT1 line to go from 0 to 1. To do this:

```
CONT:HAND:OUTP1:DATA 0  'Force the OUTPUT1 line to 0
CONT:HAND:OUTP1:USER 1 'Set the OUTPUT1:USER buffer to 1, indicating the next state
```

**Output1, Output2**
See OUTPUT1|2 Data Output Write Timing

The current state of these latched TTL outputs may be set High or Low (Default setting) using the (non-user) SCPI put_Output (COM) commands.

Write User Data

<table>
<thead>
<tr>
<th></th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write User Data</td>
<td>CONT:HAND:OUTP&lt;pin&gt;:USER</td>
<td>put_Output Method</td>
</tr>
<tr>
<td>Read last value written</td>
<td>CONT:HAND:OUTP&lt;pin&gt;:USER</td>
<td>get_Output Method</td>
</tr>
<tr>
<td>Write non-user data</td>
<td>CONT:HAND:OUTP&lt;pin&gt;:DATA</td>
<td>put_Output Method</td>
</tr>
<tr>
<td>Read last value written</td>
<td>CONT:HAND:OUTP&lt;pin&gt;:DATA</td>
<td>get_Output Method</td>
</tr>
</tbody>
</table>
Output Ports A and B

These two general purpose, 8-bit output ports are used to write data to the material handler. When any line changes state, all output lines are latched to the I/O connector as the Output Write Strobe goes Low.

The default state for data is Low.

See Data Output Write Timing Diagram

Set Port Logic:

The logic for the data lines can be set to either: Positive (1 = High) or Negative (1 = Low). This setting affects all data ports. They cannot be set independently.

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:LOGic</td>
<td>PortLogic Property</td>
</tr>
</tbody>
</table>

Combine to read or write data to Port F:

Ports A and B can be virtually combined to write data to one 16-bit I/O port F.

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:F &lt;num&gt;</td>
<td>put Port (F)</td>
</tr>
</tbody>
</table>

Input/Output Ports C and D

These two general purpose 4-bit Input/Output ports are used to write data (Output) or read data (Input). These lines could be used to write to an external device such as a step attenuator.

When any line changes state, all output lines are latched to the I/O connector as the Output Write Strobe goes Low. See Data Output Write Timing

The four lines of Port C are connected internally to the Auxiliary IO connector.

Set Input | Output Mode:

Each port may be independently defined as Output or Input.

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:C:MODE</td>
<td>PortMode Property</td>
</tr>
<tr>
<td>CONTrol:HANDler:D:MODE</td>
<td></td>
</tr>
</tbody>
</table>
## Set Port Logic:
The logic for the data lines can be set to either: Positive (1 = High) or Negative (1 = Low). This setting affects all data ports. They cannot be set independently.

**SCPI**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:LOGic</td>
</tr>
</tbody>
</table>

**COM**

<table>
<thead>
<tr>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>PortLogic Property</td>
</tr>
</tbody>
</table>

## Read or write data:
Ports C and D can be virtually combined to read or write data to one 8-bit I/O port E. When combined, both C and D ports must be set to either INPUT or OUTPUT mode.

**SCPI**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:&lt;port&gt;[DATa]</td>
</tr>
</tbody>
</table>

**COM**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>get Port(x)</td>
</tr>
<tr>
<td>put Port(x)</td>
</tr>
</tbody>
</table>

## Port C Status, Port D Status
These two output lines indicate the Read / Write mode of the C and D ports.

- A Low level indicates that the associated port is in **INPUT** mode (read only).
- A High level indicates that the associated port is in **OUTPUT** mode (write only).

These logic of these status outputs cannot be changed.

See [Input/Output Ports C and D](#) to learn how to set I/O Mode

See [Data Output Write Timing](#)

## Output Port Write Strobe
This Output line goes Low to write data from **Ports A and B** and **Ports C and D** when a change is detected on any of the data lines.

These logic of this strobe output cannot be changed.

This line is shared with Auxiliary IO connector.

See [Data Output Write Timing](#)
External Trigger
When trigger source is set to external, this Input line accepts a trigger signal from the material handler. This usually means that a part is in place and ready to be tested.
See Trigger Timing Diagram

Index
A Low signal on this Output line indicates to the material handler that the measurement is complete. This usually means that the handler can connect the next device. However, measurement data is not available until data is calculated. See Trigger Timing Diagram.

Set Function:
This line also serves as a data line. Set the function using the following commands:

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:INDex:STATe</td>
<td>IndexState</td>
</tr>
</tbody>
</table>

Ready for Trigger
When this output line goes low, it indicates to the material handler that the PNA is ready for a trigger signal.
See Trigger Timing Diagram
See Pass/Fail Timing Diagram

Set Function:
This line also serves as a data line. Set the function using the following commands:

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:RTRigger:STATe</td>
<td>ReadyForTriggerState</td>
</tr>
</tbody>
</table>
**Pass/Fail State**
This Output line indicates to the handler whether the limit test has passed or failed.

Pass/Fail state is valid only when the limit test function is ON and while Pass/Fail strobe line is Low. See Pass/Fail Timing Diagram

This line is shared with the Auxiliary IO connector.

**Set Pass / Fail Logic:**

- Positive Logic: High=Pass, Low=Fail. (Default setting)
- Negative Logic: High=Fail, Low=Pass.

```
SCPI                   COM
CONT:HANDler:PASSfail:LOGic
```

**PassFailLogic Property**

**Set Default Conditions:**

- **PASS**- the line stays in PASS state. When a device fails, then the line goes to fail after the Sweep End line is asserted.
- **FAIL**- the line stays in FAIL state. When a device passes, then the line goes to PASS state after the Sweep End line is asserted.
- **No Wait**- the line stays in PASS state. When a device fails, then the line goes to fail IMMEDIATELY. (Default setting)

```
SCPI                   COM
CONT:HANDler:PASSfail:MODE
```

**PassFailMode Property**

**Set Pass / Fail Scope:**

- **Channel scope**: The line resets to the default state after the measurements on a channel have completed.
- **Global scope**: The line resets to the default state after the measurements on all triggerable channels have completed. (Default setting)

```
SCPI                   COM
CONT:HANDler:PASSfail:SCOPe
```

**PassFailScope Property**
**Pass/Fail Write Strobe**
A Low pulse indicates that Pass/Fail line is valid and the Pass / Fail State is output to the material handler. This line is shared with the Auxiliary IO connector.

The Pass/Fail Strobe is fixed in duration and timing. However, when the strobe occurs depends on the Pass/Fail Mode and Pass/Fail Scope (Channel or Global) settings. See Pass/Fail State
See Pass/Fail Timing Diagram

**+5V**
+5V nominal output (100mA max).
Protected by self-healing fuse.

---

**Sweep End**
This output line indicates the status of the PNA sweep. The sweep includes sweeping the source and taking data.

- **Low** (falling edge) indicates that the specified sweep event has finished. This does NOT indicate that all calculations have finished.

- **High** indicates that the specified sweep event is active.

See Trigger Timing Diagram
This line is shared with the Auxiliary IO connector.

**Set Sweep Event Mode:**

- **Sweep**: indicates that a single source sweep has finished. (Default setting)

- **Channel**: indicates that a single channel has finished.

- **Global**: indicates that all enabled channels have finished.

<table>
<thead>
<tr>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTrol:HANDler:SWEepend</td>
<td>SweepEndMode Property</td>
</tr>
</tbody>
</table>
Timing Diagrams

Trigger Timing

- **T1** = 1 ms External Trigger pulse width
- **T2** > 10 ms Sweep End pulse width (both High and Low)

Pass / Fail Timing

- **T1** = 1 s Pulse width and response time of Pass / Fail Strobe
- **T2** > 10 s Ready for Trigger lag

All signals are active low.
Ports A-F Data Output Write Timing

\[ T_1 = 1 \text{ s} \quad \text{Write Strobe response time} \]
\[ T_2 = 1 \text{ s} \quad \text{Write Strobe pulse width} \]

OUTPUT1|2 Data Output Write Timing

The old state to new state transition can be either low to high (as shown) or high to low.

\[ T_1 = 0.6 \text{ s} \quad \text{Output1|2 response time} \]
\[ T_2 = 1 \text{ s} \quad \text{Input1 Strobe pulse width} \]

Input / Output Electrical Characteristics

All Material Handler I/O Input and Output lines are TTL compatible.

Input and Input/Output lines

Lines carrying information IN (or bidirectional) to the PNA from the material handler.

<table>
<thead>
<tr>
<th>Maximum Input Voltages:</th>
<th>0.5 V to 5.5 V</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTL High level:</td>
<td>2.0 V to 5.0 V</td>
</tr>
<tr>
<td>TTL Low level:</td>
<td>0 V to 0.5 V</td>
</tr>
</tbody>
</table>
PNA Input and Input/Output Circuit Diagram

![Circuit Diagram](image)

To Handler I/O Port

**Note:** The INPUT1 line does NOT have the 10K pullup resistor.

**Output Lines**

Lines carrying information OUT of the PNA to the material handler.

<table>
<thead>
<tr>
<th></th>
<th>Maximum Output Current:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Output Current</strong></td>
<td>-10 mA to 10 mA</td>
</tr>
<tr>
<td>TTL High level</td>
<td>-5 mA</td>
</tr>
<tr>
<td>TTL Low level</td>
<td>3 mA</td>
</tr>
<tr>
<td><strong>Output Voltage</strong></td>
<td></td>
</tr>
<tr>
<td>TTL High level</td>
<td>2.0 V to 3.3 V</td>
</tr>
<tr>
<td>TTL Low level</td>
<td>0 V to 0.8 V</td>
</tr>
</tbody>
</table>

PNA Output Circuit Diagram

![Circuit Diagram](image)

To Handler I/O Port

Last modified:

8-Feb-2008    Clarified Types
Nov 3, 2006    Added link from pin 20
New with the PNA-X, the 15 pin D connector provides access to Pulse Modulators and Generators.

- See IF Path Configuration block diagram, which includes the Pulse Modulators and Generators.
- See the Pulsed Application (Opt H08)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IFGateAIn</td>
<td>IF pulse gate input A (TTL)</td>
</tr>
<tr>
<td>2</td>
<td>IFGateBIn</td>
<td>IF pulse gate input B (TTL)</td>
</tr>
<tr>
<td>3</td>
<td>IFGateCIn</td>
<td>IF pulse gate input C (TTL)</td>
</tr>
<tr>
<td>4</td>
<td>IFGateDIn</td>
<td>IF pulse gate input D (TTL)</td>
</tr>
<tr>
<td>5</td>
<td>IFGateRIn</td>
<td>IF pulse gate input R (TTL)</td>
</tr>
<tr>
<td>6</td>
<td>DCOM</td>
<td>Ground</td>
</tr>
<tr>
<td>7</td>
<td>PulseSyncIn</td>
<td>Pulse gen. synchronization trigger input (TTL)</td>
</tr>
<tr>
<td>8</td>
<td>RFPulseModIn</td>
<td>RF source pulse modulation drive input (TTL)</td>
</tr>
<tr>
<td>9</td>
<td>DCOM</td>
<td>Ground</td>
</tr>
<tr>
<td>10</td>
<td>Pulse1Out</td>
<td>Hardwired pulse train output #1 (TTL)</td>
</tr>
<tr>
<td>11</td>
<td>Pulse2Out</td>
<td>Hardwired pulse train output #2 (TTL)</td>
</tr>
<tr>
<td>12</td>
<td>Pulse3Out</td>
<td>Hardwired pulse train output #3 (TTL)</td>
</tr>
<tr>
<td>13</td>
<td>Pulse4Out</td>
<td>Hardwired pulse train output #4 (TTL)</td>
</tr>
<tr>
<td>14</td>
<td>N.C.</td>
<td>No connect -- for future use</td>
</tr>
<tr>
<td>15</td>
<td>DCOM</td>
<td>Ground</td>
</tr>
</tbody>
</table>

See Pulse SCPI and COM commands
N1966A Pulse I/O Adapter

This D connector to RF adapter makes accessing the Pulse I/O connector more convenient.

Last Modified:

16-Jul-2007    Clarification
18-Jan-2007    MX New topic
**Power I/O Connector**

New on the PNA-X, the 9-pin D connector replaces much of the functionality of the AUX I/O connector on older PNA models. See PNA-X Rear Panel

![Power I/O Connector](image_url)

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+15V</td>
<td>+15V @ 400 mA</td>
</tr>
<tr>
<td>2</td>
<td>-15V</td>
<td>-15V @ 400 mA</td>
</tr>
<tr>
<td>3</td>
<td>AnalogOut1</td>
<td>Analog Output Voltage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Programmable +/-10V @100 mA out</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Nominally 0 ohms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.44mV typical resolution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1MHz BW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read and write voltage programmatically using:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- CONT:AUX:OUTP1:VOLT (SCPI - read and write)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- get OutputVoltage Method (COM - read)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- put OutputVoltage Method (COM - write)</td>
</tr>
<tr>
<td>4</td>
<td>AnalogOut2</td>
<td>Analog Output Voltage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Programmable +/-10V @100 mA out</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Nominally 0 ohms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.44mV typical resolution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1MHz BW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read and write voltage programmatically using:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- CONT:AUX:OUTP2:VOLT (SCPI - read and write)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- get OutputVoltage Method (COM - read)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- put OutputVoltage Method (COM - write)</td>
</tr>
<tr>
<td>5</td>
<td>ACOM</td>
<td>System ground</td>
</tr>
<tr>
<td>6</td>
<td>GndSense</td>
<td>Ground sense for Analog In &amp; Out</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>-----------------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>7</td>
<td>AnalogIn1</td>
<td>Connected with 51.1-ohm to ACOM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read voltage programmatically using:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>get InputVoltageEX Method</code> (COM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>CONT:AUX:OUTP3:VOLT</code> (SCPI)</td>
</tr>
<tr>
<td></td>
<td>Analog input:</td>
<td>+/-10V @ 1.22mV typical resolution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rin &gt;1 M-ohm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BW ~ 1 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ADC conversion time &lt; 1 us typical</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read voltage programmatically using:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>CONT:AUX:INP1:VOLT</code> (SCPI)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>get InputVoltageEX Method</code> (COM)</td>
</tr>
<tr>
<td>8</td>
<td>AnalogIn2</td>
<td>Connected with 51.1-ohm to ACOM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read voltage programmatically using:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>get InputVoltageEX Method</code> (COM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>CONT:AUX:INP2:VOLT</code> (SCPI)</td>
</tr>
<tr>
<td></td>
<td>Power Button</td>
<td>Open collector input</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Active low replicates power button key press.</td>
</tr>
</tbody>
</table>

Last Modified:

10-Jul-2007  Added COM commands
18-Jan-2007  MX New topic
## New Programming Commands

The following are new programming commands for **PNA release 8.0**  
See What's New

### Gain Compression Setup

<table>
<thead>
<tr>
<th>Feature</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of frequency points</td>
<td>SENS:GCS:SWE:FREQ:POIN</td>
<td>NumberOfFrequencyPoints</td>
</tr>
<tr>
<td>Number of power points</td>
<td>SENS:GCS:SWE:POW:POIN</td>
<td>NumberOfPowerPoints</td>
</tr>
<tr>
<td>Maximum number of points</td>
<td>None</td>
<td>MaximumNumberOfPoints</td>
</tr>
<tr>
<td>Total number of points</td>
<td>None</td>
<td>TotalNumberOfPoints</td>
</tr>
<tr>
<td>Acquisition mode</td>
<td>SENS:GCS:AMOD</td>
<td>AcquisitionMode</td>
</tr>
<tr>
<td>Smart tolerance</td>
<td>SENS:GCS:SMAR:TOL</td>
<td>SmartSweepTolerance</td>
</tr>
<tr>
<td>Smart Iterations</td>
<td>SENS:GCS:SMAR:MIT</td>
<td>SmartSweepMaximumIterations</td>
</tr>
<tr>
<td>Smart settling time</td>
<td>SENS:GCS:SMAR:STIM</td>
<td>SmartSweepSettlingTime</td>
</tr>
<tr>
<td>Smart show iterations</td>
<td>SENS:GCS:SMAR:SIT</td>
<td>SmartSweepShowIterations</td>
</tr>
<tr>
<td>Read compression failures</td>
<td>SENS:GCS:SFA?</td>
<td>SearchFailures</td>
</tr>
<tr>
<td>Write port map</td>
<td>SENS:GCS:PORTM</td>
<td>SetPortMap</td>
</tr>
<tr>
<td>Read Port Map (Input)</td>
<td>SENS:GCS:PORT</td>
<td>DeviceInputPort</td>
</tr>
<tr>
<td>Read Port Map (Output)</td>
<td>SENS:GCS:PORT</td>
<td>DeviceOutputPort</td>
</tr>
<tr>
<td>End of Sweep</td>
<td>SENS:GCS:EOS</td>
<td>EndOfSweepOperation</td>
</tr>
<tr>
<td>Linear input power</td>
<td>SENS:GCS:POW:LIN:INP:LEV</td>
<td>InputLinearPowerLevel</td>
</tr>
<tr>
<td>Reverse Power</td>
<td>SENS:GCS:POW:REV:LEV</td>
<td>ReverseLinearPowerLevel</td>
</tr>
<tr>
<td>Start power</td>
<td>SENS:GCS:POW:STAR:LEV</td>
<td>chan, Start Power</td>
</tr>
<tr>
<td>Stop power</td>
<td>SENS:GCS:POW:STOP:LEV</td>
<td>chan, Stop Power</td>
</tr>
<tr>
<td>Compression algorithm</td>
<td>SENS:GCS:COMP:ALG</td>
<td>CompressionAlgorithm</td>
</tr>
<tr>
<td>Compression Level</td>
<td>SENS:GCS:COMP:LEV</td>
<td>CompressionLevel</td>
</tr>
<tr>
<td>Backoff Level</td>
<td>SENS:GCS:COMP:BACK:LEV</td>
<td>CompressionBackoff</td>
</tr>
</tbody>
</table>
X Delta  
SENS:GCS:COMP:DELT:X  CompressionDeltaX

Y Delta  
SENS:GCS:COMP:DELT:Y  CompressionDeltaY

Interpolation  
SENS:GCS:COMP:INT  CompressionInterpolation

Safe Sweep enable  
SENS:GCS:SAFE:ENAB  SafeSweepEnable

Safe Sweep coarse  
SENS:GCS:SAFE:CPAD  SafeSweepCoarsePowerAdjustment

Safe Sweep fine  
SENS:GCS:SAFE:FPAD  SafeSweepFinePowerAdjustment

Safe Sweep threshold  
SENS:GCS:SAFE:FTHR  SafeSweepFineThreshold

Read all GCA data  
CALC:GCData:DATA  GetRaw2DData

Read real GCA data  
CALC:GCData:REAL  GetDataIm

Read imaginary GCA data  
CALC:GCData:IMAG  GetDataRe

### Noise Figure Setup

**Create Noise figure meas**  
Calc:Cust:Def  CreateCustomMeasurementEx

Sets the number of impedance states to use  
SENS:NOIS:IMP:COUN  ImpedanceStates

Noise averaging ON and OFF  
SENS:NOIS:AVER:STAT  NoiseAverageState

Set averaging of noise receiver.  
SENS:NOIS:AVER  NoiseAverageFactor

Set bandwidth of noise receiver.  
SENS:NOIS:BWID  NoiseBandwidth

Set gain state of noise receiver.  
SENS:NOIS:GAIN  NoiseGain

Sets noise tuner identifier  
SENS:NOIS:TUN:ID  NoiseTuner

Sets the port identifier of the ECal noise tuner that is connected to the PNA Source.  
SENS:NOIS:TUN:INP  NoiseTunerIn

Sets the port identifier of the ECal noise tuner that is connected to the DUT.  
SENS:NOIS:TUN:OUTP  NoiseTunerOut

2566
Set the excess noise source ON or OFF.

CONTROL:NOISe:SOURce or NoiseSourceState
OUTPut:MANual:NOISe[:STATe]

Set mechanical switches
SENSe:PATH:CONF:ELEM PathConfiguration

Sets the default setting for the Noise Tuner switch.
SYST:PREF:ITEM:SWIT:DEF Port1NoiseTunerSwitchPresetsToExternal

---

### Noise Figure Cal

Create Noise Cal object

CreateCustomCalEx

Set Noise Calibration method

SENS:NOIS:CAL:METH CalMethod

Noise source ENR filename

SENS:NOIS:ENR:FIL ENRFile

Set noise source Cal Kit type

SENS:NOIS:SOUR:CKIT NoiseSourceCalKitType

Set ambient temperature

SENS:NOIS:TEMP:AMB AmbientTemperature

Sets noise source connector type

SENS:NOIS:SOUR:CONN NoiseSourceConnectorType

Set Noise source temperature

SENS:CORR:TCOL:USER:VAL NoiseSourceCold

---

### Noise Figure ENR File Data Management

Set ENR calibration data.

SENS:CORR:ENR:CAL:TABL:DATA PutENRData

Read ENR calibration data.


Get/set ID of ENR table.

SENS:CORR:ENR:CAL:TABL:ID:DATA ENRID

Get/set serial number of noise source.

SENS:CORR:ENR:CAL:TABL:SERial:DATA ENRSN

Load ENR table from file.

MMEMory:LOAD:ENR LoadENRFile

Save ENR table to file.

MMEMory:STORe:ENR SaveENRFile
### Custom Cal Window

<table>
<thead>
<tr>
<th>Action</th>
<th>Command/Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turn ON</td>
<td>OFF Custom Cal window.</td>
<td>SENS:CORR:COLL:DISP:WIND DisplayNAWindowDuringCal Acquisition</td>
</tr>
<tr>
<td>Show NO Custom Cal windows.</td>
<td>SENS:CORR:COLL:DISP:WIND:AOFF DisplayOnlyCalWindowDuringCal Acquisition</td>
<td></td>
</tr>
<tr>
<td>Specify channel to sweep before Cal acquisition.</td>
<td>SENS:CORR:COLL:SWE:CHAN AllowChannelToSweepDuringCal Acquisition</td>
<td></td>
</tr>
<tr>
<td>Sweep NO channel before Cal acquisition.</td>
<td>SENS:CORR:COLL:SWE:CHAN:AOFF SweepOnlyCalChannelDuringCal Acquisition</td>
<td></td>
</tr>
<tr>
<td>Preview sweep before remote Cal acquisition.</td>
<td>SENS:CORR:COLL:GUID:PACQuire SetupMeasurementsForStep</td>
<td></td>
</tr>
</tbody>
</table>

### Miscellaneous

<table>
<thead>
<tr>
<th>Action</th>
<th>Command/Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Trigger sweep mode</td>
<td>SENS:SWE:TRIG:MODE Trigger Mode</td>
<td></td>
</tr>
<tr>
<td>Copy a Cal Set</td>
<td>SENS:CORR:CSET:COPY Copy a Cal Set</td>
<td></td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release 7.50** See What's New

### USB Power Sensors

<table>
<thead>
<tr>
<th>Specification</th>
<th>Command/Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specifies the type of power sensor to be used.</td>
<td>SYST:COMM:PSEN Path</td>
<td></td>
</tr>
<tr>
<td>Specifies the location of the power sensor to be used.</td>
<td>SYST:COMM:PSEN Locator</td>
<td></td>
</tr>
<tr>
<td>Returns the ID string of connected USB power meters / sensors.</td>
<td>SYST:COMM:USB:PMET:CAT? USBPowerMeterCatalog</td>
<td></td>
</tr>
</tbody>
</table>
### Miscellaneous

**Single trigger**

SENS:SWE:MODE SINGle

**Reads ADC voltages from Power I/O connector**

get InputVoltageEX Method

**Copies an existing Source Power Calibration to another channel.**

SYST:MACR:COPY:CHAN

ApplySourcePowerCorrectionTo

**Returns an array of Cal Set names being stored on the PNA.**

EnumerateCalSets

---

The following are new programming commands for PNA release 7.22 [See What's New](#)

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Embedded LO</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Embedded LO ON</td>
<td>OFF</td>
<td>SENS:MIX:ELO:STAT</td>
</tr>
<tr>
<td>Select tuning point</td>
<td>SENS:MIX:ELO:NORMalyze:POINT</td>
<td>NormalizePoint</td>
</tr>
<tr>
<td>Set tuning mode</td>
<td>SENS:MIX:ELO:TUNing:MODE</td>
<td>TuningMode</td>
</tr>
<tr>
<td>Set broadband sweep span</td>
<td>SENS:MIX:ELO:TUNing:SPAN</td>
<td>BroadbandTuningSpan</td>
</tr>
<tr>
<td>Set precise tuning tolerance</td>
<td>SENS:MIX:ELO:TUNing:TOL</td>
<td>PreciseTuningTolerance</td>
</tr>
<tr>
<td>Set precise tuning iterations</td>
<td>SENS:MIX:ELO:TUNing:ITER</td>
<td>MaxPreciseTuningIterations</td>
</tr>
<tr>
<td>LO delta frequency</td>
<td>SENS:MIX:ELO:LO:DELTA</td>
<td>LOFrequencyDelta</td>
</tr>
<tr>
<td>Resets tuning parameters</td>
<td>SENS:MIX:ELO:TUNing:RESet</td>
<td>ResetTuningParameters</td>
</tr>
<tr>
<td>Reset LO Delta frequency</td>
<td>SENS:MIX:ELO:LO:RESET</td>
<td>ResetLOFrequency</td>
</tr>
<tr>
<td><strong>Embedded LO Diagnostics</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clear current diagnostic information</td>
<td>SENS:MIX:ELO:DIAG:CLEAr</td>
<td>Clear</td>
</tr>
<tr>
<td>Get result of the last tuning sweeps.</td>
<td>SENS:MIX:ELO:DIAG:STATus?</td>
<td>StatusAsString</td>
</tr>
<tr>
<td>Get number of tuning sweeps.</td>
<td>SENS:MIX:ELO:DIAG:SWE:COUNt?</td>
<td>NumberOfSweeps</td>
</tr>
</tbody>
</table>
Was a marker was used for a tuning sweep?  
IsMarkerOn

Get the marker X-axis position.  
MarkerPosition

Get the marker annotation.  
MarkerAnnotation

Get the tuning sweep X axis annotation.  
XAxisAnnotation

Get the tuning sweep Y axis annotation.  
YAxisAnnotation

Get the Start sweep value.  
XAxisStart

Get the Stop sweep value.  
SENS:MIX:ELO:DIAG:SWE:X:STOP?  
XAxisStop

Returns the tuning sweep parameter name.  
SENS:MIX:ELO:DIAG:SWE:PAR?  
Parameter

Returns the tuning sweep title.  
SENS:MIX:ELO:DIAG:SWE:TITL?  
StepTitle

Returns the LO frequency delta from this tuning sweep  
LODeltaFound

---

**Enhanced Response Cal**

Set guided Cal method  
SENS:CORR:COLL:GUID:PATH:CMET  
PathCalMethod

Set guided Thru method  
SENS:CORR:COLL:GUID:PATH:TMET  
PathThruMethod

---

The following are new programming commands for **PNA release 7.21**  
See What's New

**Perform Isolation Cal**

Perform ECal Isolation  
SENS:CORR:COLL:ISOL:ECAL  
ECALIsolation

Increment Avg for ECal Isolation  
SENS:CORR:COLL:ISOL:AVER:INCR  
IsolationAveragingIncrement

Perform Guided (mech) Isolation  
SENS:CORR:COLL:GUID:ISOL:PATHs  
GetIsolationPaths Method

SetIsolationPaths Method

Increment Avg for Guided Isolation  
IsolationAveragingIncrement Property
## Miscellaneous Cal

Preference to Save Cal to selected Cal Set

**SENS:CORR:PREF:CSET:SAVE**  
RemoteCalStoragePreference

Perform a Reference Receiver cal as part of a Source Power cal.

**SOUR:POW:CORR:COLL:SAVE RREceiver**  
ApplyPowerCorrectionValuesEx

---

The following are new programming commands for **PNA release 7.20**  
[See What's New](#)

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RF and IF Path Configuration</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Catalog configuration names</td>
<td><strong>SENS:PATH:CONF:CAT?</strong></td>
<td>Configurations</td>
</tr>
<tr>
<td>Load a configuration</td>
<td><strong>SENS:PATH:CONF:SEL</strong></td>
<td>LoadConfiguration</td>
</tr>
<tr>
<td>Store a configuration</td>
<td><strong>SENS:PATH:CONF:STOR</strong></td>
<td>StoreConfiguration</td>
</tr>
<tr>
<td>Delete a configuration</td>
<td><strong>SENS:PATH:CONF:DEL</strong></td>
<td>DeleteConfiguration</td>
</tr>
<tr>
<td>Read the name of a configuration</td>
<td><strong>SENS:PATH:CONF:NAME?</strong></td>
<td>config.Name</td>
</tr>
<tr>
<td>Write descriptive text</td>
<td><strong>SENS:PATH:CONF:DTEX</strong></td>
<td>DescriptiveText</td>
</tr>
<tr>
<td>Catalog all elements</td>
<td><strong>SENS:PATH:CONF:ELEM:CAT?</strong></td>
<td>Elements</td>
</tr>
<tr>
<td>Catalog all settings</td>
<td><strong>SENS:PATH:CONF:ELEM:VAL:CAT?</strong></td>
<td>Values</td>
</tr>
<tr>
<td>Set element</td>
<td><strong>SENS:PATH:CONF:ELEM</strong></td>
<td>Element</td>
</tr>
<tr>
<td>Read name of current element</td>
<td><strong>SENS:PATH:CONF:ELEM</strong></td>
<td>element.Name</td>
</tr>
<tr>
<td>Set value for an element</td>
<td><strong>SENS:PATH:CONF:ELEM</strong></td>
<td>Value</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IF / DSP Settings</strong></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets ADC capture mode: auto or manual</td>
<td><strong>SENS:IF:FILT:CMOD</strong></td>
<td>ADCCaptureMode</td>
</tr>
<tr>
<td>Sets and returns method for specifying the way the IF Frequency is determined.</td>
<td><strong>SENS:IF:FREQ:AUTO</strong></td>
<td>IFFrequencyMode</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FREQ</td>
<td>Sets and returns the IF frequency.</td>
<td></td>
</tr>
<tr>
<td>IFFrequency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FilterErrors</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:AUTO</td>
<td>Sets digital filter mode.</td>
<td></td>
</tr>
<tr>
<td>FilterMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAGE&lt;n&gt;:COEF</td>
<td>Sets Stage1Coefficients</td>
<td></td>
</tr>
<tr>
<td>Stage1Coefficients</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAGE1:FREQ</td>
<td>Sets Stage1 NCO frequency</td>
<td></td>
</tr>
<tr>
<td>Stage1Frequency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAGE&lt;n&gt;:COEF</td>
<td>Returns the maximum value of any single stage1coefficient.</td>
<td></td>
</tr>
<tr>
<td>Stage1MaximumCoefficient</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAG&lt;n&gt;:COUNT?</td>
<td>Returns the maximum number of Stage1 coefficients</td>
<td></td>
</tr>
<tr>
<td>Stage1MaximumCoefficientCount</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAG&lt;n&gt;:COUNT?</td>
<td>Returns the minimum number of Stage1 coefficients</td>
<td></td>
</tr>
<tr>
<td>Stage1MinimumCoefficientCount</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAGE&lt;n&gt;:COEF</td>
<td>Sets Stage2Coefficients</td>
<td></td>
</tr>
<tr>
<td>Stage2Coefficients</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAGE&lt;n&gt;:COEF</td>
<td>Returns the maximum value of any single stage2coefficient.</td>
<td></td>
</tr>
<tr>
<td>Stage2MaximumCoefficient</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAG&lt;n&gt;:COUNT?</td>
<td>Returns the maximum number of Stage2 coefficients</td>
<td></td>
</tr>
<tr>
<td>Stage2MaximumCoefficientCount</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAG&lt;n&gt;:COUNT?</td>
<td>Returns the minimum number of Stage2 coefficients</td>
<td></td>
</tr>
<tr>
<td>Stage2MinimumCoefficientCount</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAGE3:TYPE</td>
<td>Sets and returns stage3 filter type</td>
<td></td>
</tr>
<tr>
<td>Stage3FilterType</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAGE3:CAT?</td>
<td>Returns the names of supported types of Stage3 filters.</td>
<td></td>
</tr>
<tr>
<td>Stage3FilterTypes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAGE3:PAR</td>
<td>Sets and returns the parameter value of the current filter type.</td>
<td></td>
</tr>
<tr>
<td>Stage3Parameter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SENS:IF:FILT:STAGE3:PAR</td>
<td>Returns maximum parameter value for the current filter type.</td>
<td></td>
</tr>
<tr>
<td>Stage3ParameterMaximum</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage3ParameterMinimum</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stage3Parameters</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Pulse Generator Settings

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turns the pulse output ON and OFF.</td>
<td>SENS:PULSe:STATe</td>
<td>State</td>
</tr>
<tr>
<td>Sets the pulse delay</td>
<td>SENS:PULS:DEL</td>
<td>Delay</td>
</tr>
<tr>
<td>Sets the pulse delay increment.</td>
<td>SENS:PULS:DINC</td>
<td>DelayIncrement</td>
</tr>
<tr>
<td>Sets the pulse-period (1/PRF) for ALL pulse generators.</td>
<td>SENS:PULS:PERiod</td>
<td>Period</td>
</tr>
<tr>
<td>Sets the pulse width</td>
<td>SENS:PULS:WIDT</td>
<td>Width</td>
</tr>
</tbody>
</table>

### Auxiliary Triggering

<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the number of the AuxTrig connector pair being used.</td>
<td></td>
<td>Not Applicable</td>
</tr>
<tr>
<td>Read number of aux connector pairs.</td>
<td>TRIGger:AUX:COUN?</td>
<td>AuxiliaryTriggerCount</td>
</tr>
<tr>
<td>Turns ON / OFF the trigger output</td>
<td>TRIG:CHAN:AUX</td>
<td>Enable</td>
</tr>
<tr>
<td>Turns handshake ON / OFF.</td>
<td>TRIG:CHAN:AUX:HAND</td>
<td>HandshakeEnable</td>
</tr>
<tr>
<td>Specifies the polarity of the trigger IN signal.</td>
<td>TRIG:CHAN:AUX:IPOL</td>
<td>TriggerInPolarity</td>
</tr>
<tr>
<td>Specifies the polarity of the trigger OUTPUT signal.</td>
<td>TRIG:CHAN:AUX:OPOL</td>
<td>TriggerOutPolarity</td>
</tr>
<tr>
<td>Specifies the type of aux Input trigger:Edge or Level.</td>
<td>TRIG:CHAN:AUX:TYPE</td>
<td>TriggerInType</td>
</tr>
<tr>
<td>Specifies the pulse width of the Output signal</td>
<td>TRIG:CHAN:AUX:DUR</td>
<td>TriggerOutDuration</td>
</tr>
<tr>
<td>Specifies how often a trigger output signal is sent:Point or Sweep.</td>
<td>TRIG:CHAN:AUX:INT</td>
<td>TriggerOutInterval</td>
</tr>
<tr>
<td>Specifies whether the aux trigger out signal is sent BEBefore or AFTer the acquisition.</td>
<td>TRIG:CHAN:AUX:POS</td>
<td>TriggerOutPosition</td>
</tr>
</tbody>
</table>
Sets the External Trigger OUT behavior to have either Global or Channel scope.

### Preferences

<table>
<thead>
<tr>
<th>Offset for Receiver Attenuator</th>
<th>SYST:ITEM:OFFS:RCV</th>
<th>OffsetReceiverAttenuator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offset for Source Attenuator</td>
<td>SYST:ITEM:OFFS:SRC</td>
<td>OffsetSourceAttenuator</td>
</tr>
<tr>
<td>Report source unleveled events as errors?</td>
<td>SYST:ERR:REP:SUNL</td>
<td>EnableSourceUnleveledEvents</td>
</tr>
</tbody>
</table>

### ALC Leveling

<table>
<thead>
<tr>
<th>Returns list of valid ALC Leveling Modes</th>
<th>SOUR:POW:ALC:MODE:CAT?</th>
<th>GetSupportedALCModes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set ALC Mode</td>
<td>SOUR:POW:ALC:MODE</td>
<td>ALCLevelingMode</td>
</tr>
</tbody>
</table>

### Specifying Source Ports

<table>
<thead>
<tr>
<th>Returns the number of source ports.</th>
<th>None</th>
<th>SourcePortCount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the string names of source ports.</td>
<td>SOURce:CAT?</td>
<td>SourcePortNames</td>
</tr>
<tr>
<td>Returns the source port number of the specified string port name.</td>
<td>None</td>
<td>GetPortNumber</td>
</tr>
</tbody>
</table>

2574
### Miscellaneous

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Touchscreen ON</td>
<td>Off</td>
<td>SYST:TOUCHscreen</td>
</tr>
<tr>
<td>Launches dialogs</td>
<td>None</td>
<td>app.LaunchDialog</td>
</tr>
<tr>
<td>Returns error term data (<strong>new behavior</strong>).</td>
<td>N/A</td>
<td>Get ErrorTermByString</td>
</tr>
<tr>
<td>Turns Trace Title ON and OFF</td>
<td>DISP:WIND:TRAC:TITL:STATE</td>
<td>TraceTitleState</td>
</tr>
<tr>
<td>Writes and reads trace title.</td>
<td>DISP:WIND:TRAC:TITL:DATA</td>
<td>TraceTitle</td>
</tr>
<tr>
<td>Makes PNA app visible or not</td>
<td>DISP:VISIBLE</td>
<td>Visible</td>
</tr>
<tr>
<td>Read voltages on the Power I/O connector.</td>
<td>CONT:AUX:INP:VOLT?</td>
<td>None</td>
</tr>
<tr>
<td>Outputs voltages on the Power I/O connector.</td>
<td>CONT:AUX:OUTP:VOLT</td>
<td>None</td>
</tr>
<tr>
<td>Segment sweep time</td>
<td></td>
<td>Sweep Time</td>
</tr>
</tbody>
</table>

---

The following are new programming commands for **PNA release 7.1**  
See What's New

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Frequency Offset Commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Freq Offset ON</td>
<td>Off</td>
<td>SENS:FOM</td>
</tr>
<tr>
<td>Read available ranges</td>
<td>SENS:FOM:CAT?</td>
<td></td>
</tr>
<tr>
<td>Read number of ranges</td>
<td>SENS:FOM:COUN?</td>
<td>RangeCount</td>
</tr>
<tr>
<td>X-Axis display range</td>
<td>SENS:FOM:DISP:SEL</td>
<td>DisplayRange</td>
</tr>
<tr>
<td>Read range name</td>
<td>SENS:FOM:RANG:NAME?</td>
<td>Name FOMRange</td>
</tr>
<tr>
<td>Read range number</td>
<td>SENS:FOM:RNUM?</td>
<td>rangeNumber</td>
</tr>
<tr>
<td>Set range coupling</td>
<td>SENS:FOM:RANG:COUP</td>
<td>Coupled</td>
</tr>
<tr>
<td>Set sweep type</td>
<td>SENS:FOM:RANG:SWE:TYPE</td>
<td>SweepType</td>
</tr>
<tr>
<td>Set CW freq</td>
<td>SENS:FOM:RANG:FREQ:CW</td>
<td>CWFrequency</td>
</tr>
<tr>
<td>Command</td>
<td>Command Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Set start freq</td>
<td>SENS:FOM:RANG:FREQ:STAR</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Set stop freq</td>
<td>SENS:FOM:RANG:FREQ:STOP</td>
<td>StopFrequency</td>
</tr>
<tr>
<td>Set offset value</td>
<td>SENS:FOM:RANG:FREQ:OFFS</td>
<td>Offset</td>
</tr>
<tr>
<td>Set divisor value</td>
<td>SENS:FOM:RANG:FREQ:DIV</td>
<td>Divisor</td>
</tr>
<tr>
<td>Set multiplier value</td>
<td>SENS:FOM:RANG:FREQ:MULT</td>
<td>Multiplier</td>
</tr>
</tbody>
</table>

**Freq. Offset Segment Sweep**

<table>
<thead>
<tr>
<th>Command</th>
<th>Command Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>OFF</td>
<td>SENS:FOM:RANG:SEGM</td>
</tr>
<tr>
<td>Add a segment</td>
<td>SENS:FOM:RANG:SEGM:ADD</td>
<td>FOM/Add</td>
</tr>
<tr>
<td>Delete a segment</td>
<td>SENS:FOM:RANG:SEGM:DEL</td>
<td>Remove</td>
</tr>
<tr>
<td>Count the segments</td>
<td>SENS:FOM:RANG:SEGM:COUNT?</td>
<td>Count</td>
</tr>
<tr>
<td>Center Frequency</td>
<td>SENS:FOM:RANG:SEGM:FREQ:CENT</td>
<td>CenterFrequency</td>
</tr>
<tr>
<td>Frequency Span</td>
<td>SENS:FOM:RANG:SEGM:FREQ:SPAN</td>
<td>FrequencySpan</td>
</tr>
<tr>
<td>Start Frequency</td>
<td>SENS:FOM:RANG:SEGM:FREQ:STAR</td>
<td>StartFrequency</td>
</tr>
<tr>
<td>Stop Frequency</td>
<td>SENS:FOM:RANG:SEGM:FREQ:STOP</td>
<td>Stop Frequency</td>
</tr>
<tr>
<td>Number of Points</td>
<td>SENS:FOM:RANG:SEGM:SWE:POIN</td>
<td>Number of Points</td>
</tr>
<tr>
<td>IF Bandwidth value</td>
<td>SENS:FOM:RANG:SEGM:BWID</td>
<td>IF Bandwidth</td>
</tr>
<tr>
<td>IF Bandwidth control</td>
<td>SENS:FOM:RANG:SEGM:BWID:CONT</td>
<td>IF BandwidthOption</td>
</tr>
<tr>
<td>Source Power value</td>
<td>SENS:FOM:RANG:SEGM:POW</td>
<td>Test Port Power</td>
</tr>
<tr>
<td>Sweep time value</td>
<td>SENS:FOM:RANG:SEGM:SWE:TIME</td>
<td>Sweep Time</td>
</tr>
<tr>
<td>Sweep time control</td>
<td>SENS:FOM:RANG:SEGM:SWE:TIME:CONT</td>
<td>TimeOption</td>
</tr>
</tbody>
</table>

**Miscellaneous**

<table>
<thead>
<tr>
<th>Command</th>
<th>Command Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Power (Auto</td>
<td>ON</td>
<td>OFF)</td>
</tr>
</tbody>
</table>

The following are new programming commands for **PNA release 6.2.** [See What's New](#)
<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Multiport</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perform Multiport restart</td>
<td>SYST:CONFigure</td>
<td>app.Configure</td>
</tr>
<tr>
<td>Returns list of supported testsets</td>
<td>tsets.TestsetCatalog</td>
<td></td>
</tr>
<tr>
<td>Returns the testset model</td>
<td>ts.Type</td>
<td></td>
</tr>
<tr>
<td>Sets and returns the logical port value</td>
<td>ts.SelectPort</td>
<td></td>
</tr>
<tr>
<td>Sets and returns a port mapping for a single port.</td>
<td>SENS:MULT:PORT:SEL</td>
<td></td>
</tr>
<tr>
<td>Sets and returns the display label</td>
<td>SENS:MULT:LAB</td>
<td></td>
</tr>
<tr>
<td>Create new parameter</td>
<td>CALC:PAR:DEF:EXT</td>
<td></td>
</tr>
<tr>
<td>Change parameter</td>
<td>CALC:PAR:MOD:EXT</td>
<td></td>
</tr>
<tr>
<td>Read the Test Set model for a Cal Set.</td>
<td>SENS:CORR:CSET:TSet:ALLPorts?</td>
<td>calset.OutputPorts</td>
</tr>
<tr>
<td>Read the Port Mapping for a Cal Set</td>
<td>SENS:CORR:CSET:TSET:TYPE?</td>
<td>calset.TestSetType</td>
</tr>
<tr>
<td><strong>Macros</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify Path and filename</td>
<td>SYSTem:SHORtcut:PATH</td>
<td></td>
</tr>
<tr>
<td>Specify Arguments</td>
<td>SYSTem:SHORtcut:ARGuments</td>
<td></td>
</tr>
<tr>
<td>Specify Title</td>
<td>SYSTem:SHORtcut:TITLe</td>
<td></td>
</tr>
<tr>
<td>Run a macro</td>
<td>SYSTem:SHORtcut:EXECute</td>
<td></td>
</tr>
<tr>
<td>Remove a macro</td>
<td>SYSTem:SHORtcut:DELete</td>
<td></td>
</tr>
<tr>
<td><strong>Set Mixer Fixed Input and Output</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set input to fixed or swept</td>
<td>SENS:MIX:INP:FREQ:MODE</td>
<td>mxr.InputRangeMode</td>
</tr>
<tr>
<td>Set output to fixed or swept</td>
<td>SENS:MIX:OUT:FREQ:MODE</td>
<td>mxr.OutPutRangeMode</td>
</tr>
<tr>
<td>Set input fixed frequency</td>
<td>mxr.InputFixedFrequency</td>
<td></td>
</tr>
<tr>
<td><strong>Receiver-only source power cal</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Select and Acquire receiver-only readings</td>
<td>SOUR:POW:CORR:COLL:ACQ:REC</td>
<td>AcquirePowerReadingsEx Method</td>
</tr>
</tbody>
</table>
### sNp Data

Reads SnP data for the specified ports.  
**CALC:DATA:SNP:PORTs?**  
**GetSnpDataWithSpecifiedPorts**

Saves SnP data for the specified ports.  
**CALC:DATA:SNP:PORTs:Save**  
**WriteSnpFileWithSpecifiedPorts**

---

The following are new programming commands for **PNA release 6.04**

<table>
<thead>
<tr>
<th>Description</th>
<th>SCPI</th>
<th>COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equation Editor</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Turn equation ON and OFF</td>
<td><strong>CALC:EQUation:STATe</strong></td>
<td></td>
</tr>
<tr>
<td>Set equation</td>
<td><strong>CALC:EQUation:TEXT</strong></td>
<td></td>
</tr>
<tr>
<td>Returns validity of the equation</td>
<td><strong>CALC:EQUation:VALid?</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Specify THRU adapter and delay for Unknown Thru and Adapter Removal</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specifies the use of a THRU adapter.</td>
<td><strong>SENS:CORR:COLL:GUID:ADAP:CREate?</strong></td>
<td></td>
</tr>
<tr>
<td>Returns the number of THRU adapters that have been created.</td>
<td><strong>SENS:CORR:COLL:GUID:ADAP:COUN?</strong></td>
<td></td>
</tr>
<tr>
<td>Specifies the adapter delay.</td>
<td><strong>SENS:CORR:COLL:GUID:ADAP:DELay</strong></td>
<td></td>
</tr>
<tr>
<td>Specifies the adapter description</td>
<td><strong>SENS:CORR:COLL:GUID:ADAP:DESC</strong></td>
<td></td>
</tr>
<tr>
<td>Specifies the port pairs for the THRU connection.</td>
<td><strong>SENS:CORR:COLL:GUID:ADAP:PATHs</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Macros</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify Path and filename</td>
<td><strong>SYSTem:SHORTcut:PATH</strong></td>
<td></td>
</tr>
<tr>
<td>Specify Arguments</td>
<td><strong>SYSTem:SHORTcut:ARGuments</strong></td>
<td></td>
</tr>
<tr>
<td>Specify Title</td>
<td><strong>SYSTem:SHORTcut:TITLe</strong></td>
<td></td>
</tr>
<tr>
<td>Run a macro</td>
<td><strong>SYSTem:SHORTcut:EXECute</strong></td>
<td></td>
</tr>
<tr>
<td>Remove a macro</td>
<td><strong>SYSTem:SHORTcut:DELete</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Miscellaneous</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
At the end of a power sweep, specifies whether to maintain source power at the start or stop power level.

```
SYST:PREF:ITEM:PSRT            PowerSweepRetracePowerMode
```

For single-band frequency or segment sweeps ONLY, specify whether to turn RF power ON or OFF during a retrace.

```
SYST:PREF:ITEM:RETR:POW        PowerOnDuringRetraceMode
```

Various print options

```
Hardcopy commands
```

**New SCPI Programming Examples:**

- Unguided Cal on Multiple Channels
- Triggering the PNA

---

Last modified:

<table>
<thead>
<tr>
<th>Date</th>
<th>Update Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>March 10, 2008</td>
<td>MX Updated for 7.2</td>
</tr>
<tr>
<td>SeptMarch 10, 2008</td>
<td>MQ Updated for 7.1</td>
</tr>
</tbody>
</table>
COM versus SCPI

There are two methods you can use to remotely control the PNA: COM and SCPI. The following topics can help you choose the method that best meets your needs:

- **Software Connection**
- **Physical Connection**
- **Selecting a Method**
- **Programming Languages**

**Software Connection**

**COM** uses a binary protocol, allowing you to directly invoke a PNA feature. This is more efficient than SCPI. For example, the following statement calls directly into the PNA, executing the routine GetIDString.

```
PNA.GetIDString()
```

**SCPI** is a text based instrument language. To retrieve the ID string, you would send the following text string to the PNA:

```
IbWrite( "*IDN?"
```

The PNA SCPI parser would first decode this text string to determine that the user has asked for the PNA to identify itself. Then the parser would call GetIDString().

**The Physical Connection**

**Internal Control**

With either COM or SCPI, the best throughput is attained by using the PNA's internal PC to execute your test code. However, if your test code uses too much system resources (CPU cycles and/or memory), this will slow the PNA's performance.

Using the SICL I/O Libraries, you can also connect to the PNA from a program running on the PNA.

**External Control**

You can control the PNA from a remote PC using either COM or SCPI.

**COM** - (Component Object Model) can be used to access any program like the PNA (835x.exe) or library (.dll) that exposes its features using a COM compliant object model. These programs or libraries are called "servers". Programs (like your remote program on your PC) that connect to and use the features of these servers are called "clients."

With COM, the server and the client do not need to reside on the same machine. DCOM, or distributed COM, is easy to configure and makes the location of the server transparent to the client. When you access the PNA from a remote computer, you are using DCOM. In this case, the mechanical transport is a LAN (local area network).

**SCPI** - Using a GPIB interface card in a remote computer, you can connect to the instrument using a GPIB cable. There are some constraints on the length of this cable and the number of instruments that can be daisy-chained.
Using the Agilent SICL I/O libraries, you can connect to the instrument over a LAN connection. (LAN or INTERNAL) You can send SCPI commands using COM with the `ScpiStringParser` object.

**Selecting a Method**

You should almost always choose COM for the following reasons:

- COM executes faster most of the time.
- COM is generally easier to use. The latest development tools embrace COM and know how to make your life easier with integrated development environments that show automation syntax as you type.
- As time goes on, more emphasis will be put on COM as the preferred programming paradigm.

But choosing a connection method depends on your situation. Here are some additional things to consider:

1. If you want to use the PNA to control other GPIB instruments, you may want to use COM as the means of talking to the PNA. In GPIB, the PNA can not be configured as both **System Controller** and **talker/listener**. Because the PNA does not support pass control mode, only one mode can be used at a time.

2. If you have legacy code written in SCPI for another network analyzer, you may be able to leverage that code to control the PNA. However, the PNA uses a different platform than previous Agilent Network Analyzers. Therefore, not all commands have a direct replacement. See the PNA Code Translator Application.

**Programming Languages**

You can program the PNA with either COM or SCPI using several languages. The most common include:

**Agilent VEE** - With this language you can send text based SCPI commands and also use automation. VEE 6.0 or later is recommended.

**Visual Basic** - This language has great support for automation objects and can be used to drive SCPI commands. The use of VISA drivers for your GPIB hardware interface will make the task of sending SCPI commands easier.

**C++** - This language can do it all. It is not as easy to use as the above two, but more flexible.
Remotely Specifying a Source Port

In the 'not-to-distant past', it was a simple task to specify a PNA source port. It was either port 1 or port 2. Now, the following reasons, it is not so simple:

- **Internal 2nd sources** are now offered on various PNA models. However, some source ports do not have an obvious port number. One example is the second source on the PNA-X 2-port model (option 224). Learn more about Internal Second Sources.

- **External sources** can now be controlled by the PNA as though they are internal sources. External sources do not have a source port number, but use String names as identifiers.
  - **For FCA ONLY**: Once configured using the Configuration dialog, an external source can be selected remotely and controlled by the PNA by specifying the LOName using SCPI or COM.
  - **All other uses for External sources**: The external source must be configured and selected manually. You can then save an Instrument State file, then recall that state file remotely to effectively select an external source.

- **Multiport test sets**…choose between ports 1 through port N, where N is the number of ports on the test set. You still use a port number, but what port number to use may not be easy to recognize because of port mapping. Learn more about Multiport test sets.

**Source Port String Names**

The PNA User Interface (UI) makes it easy to configure and select the sources and ports. Remotely however, string names are used now, in addition to port numbers, to specify a Source port.

**COM** - The existing COM commands which specify source ports as numbers, are still used. However, it may be necessary to learn the string name for the source port to be used, then learn the port number using the following commands. Port numbers are assigned dynamically depending on whether external sources are selected.

- **SourcePortNames Property**
- **GetPortNumber Method**
- **SourcePortCount Property**.

**SCPI** - All of the existing SCPI commands that specify a source port are extended to also allow the source port to be specified using string names. The following new command was created to list the available source port string names: Source:Cat?
Connecting the PNA to a PC

This document describes how to temporarily connect a PC to a PNA using a LAN cable. This is not necessary if your PNA is already connected to a network. This type of connection is for conveniently transferring large files, such as firmware, that may have already been downloaded and stored on the PC.

The PC can have any version of Windows (Windows 95 or newer.)

You will need the following:

- RJ-45 LAN crossover cable (or two normal cables with a suitable hub).
- Must be logged on the PNA with an Administrator User name and password.

Note: If your PC is on a domain, do not leave that domain by changing to a workgroup. This may prevent you from later rejoining your domain unless you involve your IT systems administrator. The following procedure will work regardless of whether or not you are on a domain and will not change any domain settings on the PC.

Procedure for All Operating Systems

1. Disconnect the PC from any existing LAN and connect it to the PNA using a crossover cable or hub. There is no need to turn it off to do this.

2. Find the current IP address of your PC. Open a DOS prompt (command window) and type `ipconfig`.

3. On the PNA, click File, then Exit to close the PNA application.


5. Right-click on Local Area Connection and select Properties.

6. Select Internet Protocol (TCP/IP) then click Properties.

7. Click Use the following IP address then enter an IP address that is ONE more or ONE less than that noted for the PC. Do not use a number that ends in 000. For this example, one could use either 10.0.0.101 or 10.0.0.99. We'll assume the use of 101.

8. For Subnet mask, enter 255.255.255.0. Click OK, then OK again

For Windows 95/98 systems, go to Win95/98 step 9 below.

9. On the PC, open Windows Explorer. Click Tools, then Map Network Drive. Note the drive letter shown (for this example we'll assume it is "X").

10. In Folder or Path type the IP address that was just entered for the PNA followed by C$ in the following format:\10.0.0.101\C$

11. In Connect As dialog, enter the PNA User name and Password. Unless it has been changed, this is either Administrator with NO password or PNA-Admin, Password Agilent. Click OK.
12. In Windows Explorer on the PC, you should now see a new drive letter entry (X) with the description of C$ on 10.0.0.101. The entire contents of the PNA C drive will now be available for reading or writing. Files can now be transferred by simply dragging them from one location to the other.

13. When you are done transferring files, disconnect the LAN cable connection between the two and reconnect your PC to the network if needed. If the PNA will never be connected to any network, the current network settings could remain. However it may be safer to reset the TCP/IP settings (changed in step 7) back to Obtain an IP address automatically.

If your PC currently has no IP address assigned

Follow steps 4 through 8 to add an IP address for the PC. The actual steps may vary slightly depending upon your operating system (NT4 uses Network Neighborhood and you must click on Protocols.) Assign the PC an IP address of 10.0.0.100. The last digits can be anything between 1 and 255, but do not alter the first 3 numbers (10.0.0.) When complete, reset the PC network configuration back to its previous settings.

Win95/98 Procedure for Accessing/Transferring Files

Because Windows 95/98 does not have the security features of NT-based operating systems (NT, Win2k, XP), the PC cannot access the drive on the PNA. To get around this limitation, any files that need to be transferred to/from the PC must be placed in a shared folder. The PNA can then read or write within this folder. This procedure will work for all versions of Windows.

Steps 1 through 8 are identical to the above and must be performed first.

9. On the PC, open Windows Explorer. Create a directory under the C drive named Shared. Right click the Shared folder name and select Sharing. Share the directory with full read/write permissions. If Sharing does not appear as a choice, then file sharing is not enabled. To enable file sharing,

   1. Right click Network Neighborhood, then click Properties.
   2. Click File and Print Sharing and enable give others access to my files
   3. Click OK, then OK again.
   4. Repeat the beginning of this step.

10. Copy the files to be transferred to the PNA to this shared folder.

11. On the PNA, open Windows Explorer. Click on Tools, Map Network Drive. Note the drive letter shown (for this example we?ll assume it is "X"). Uncheck "Reconnect at Logon" if it is currently checked.

9. Under the Folder entry, enter the IP address of the PC and the shared folder name in the following format:\10.0.0.100\Shared

10. The PNA should immediately connect to this folder and display its contents as drive "X". Files can now be read from, or written to, this shared directory (shown as Drive X.) Files can be transferred by simply dragging them from one location to the other.

11. When you are done transferring files, disconnect the LAN cable connection between the two and reconnect your PC to the network if needed. If the PNA will never be connected to any network, its current network settings could remain, however it is probably safer to reset the TCP/IP settings (changed in step 7) back to
Obtain an IP address automatically.
Easy versus Secure Configuration

When upgrading Firmware on the PNA, you encounter a **Choose Configuration** dialog box. This is used to determine the level of security set for the DCOM interface on the PNA. For more detailed information on the security settings for the DCOM interface, including a procedure for making these settings manually, see [Configure for COM-DCOM Programming](#).

Comparison of the "Easy and More Secure" settings are as follows:

**Easy Connection:**

- No configuration of the PNA required for remote access to connect.
- Anyone on the local subnet can access the PNA remotely.
- People from other NT domains can connect to the PNA.

**More Secure:**

- Requires creating users on the PNA or adding the PNA to a domain
- An administrator of the PNA can specify users or groups that are allowed remote access to the PNA application
Changing Network Client

If your PC network uses Novell NetWare servers, a change must be made to the PNA setup before it can operate on your network. If you are unsure, ask your local IT department.

**Note:** Do NOT Uninstall "Client for Microsoft Networks". This will prevent proper operation of the PNA.

To remove "Client for Microsoft Networks" (Remove is different from Uninstall):

1. From the PNA Desktop, right-click **My Network Places**
2. Click **Properties**
3. Right-click **Local Area Connection**
4. Click **Properties**
5. Click (remove the check from) **Client for Microsoft Networks**

To install "Client Service for NetWare".

1. Click **Install**
2. In **Select Network Component Type**, make sure **Client** is selected
3. Click **Add**
4. In **Select Network Client**, make sure **Client Service for NetWare** is selected
5. Click **OK**.
Troubleshooting the PNA

By running a few checks, you can identify if the analyzer is at fault. Before calling Agilent Technologies or returning the instrument for service, please make the following checks.

- **Check the Basics**
- **PNA Application Terminates Unexpectedly**
- **Check Error Terms**
- **Check the Service Guide**

### Other Support Topics

**Check the Basics**

A problem can often be solved by repeating the procedure you were following when the problem occurred. Before calling Agilent Technologies or returning the instrument for service, please make the following checks:

**Note:** Problems with the PNA application (slow or terminates unexpectedly) can be caused by a faulty Hard Disk Drive (HDD). For more information, see Preventing PNA Hard Drive Problems and The PNA HDD Recovery Process.

1. Is there power at the power socket? Is the instrument plugged in?
2. Is the instrument turned on? Check to see if the front panel line switch and at least one of the LED rings around the test ports glows green. This indicates the power supply is on.
3. If you are experiencing difficulty with the front-panel keypad or peripherals, the USB bus may be overloaded. Remove the USB devices, restart the PNA, and reconnect the USB devices. See Power-up.
4. If other equipment, cables, and connectors are being used with the instrument, make sure they are connected properly and operating correctly.
5. Review the procedure for the measurement being performed when the problem appeared. Are all the settings correct?
6. If the instrument is not functioning as expected, return the unit to a known state by pressing the **Preset** key.
7. Is the measurement being performed, and the results that are expected, within the specifications and capabilities of the instrument?
8. If the problem is thought to be due to firmware, check to see if the instrument has the latest firmware before starting the troubleshooting procedure.
9. Check that the measurement calibration is valid. See Accurate Measurement Calibrations for more information.
10. If the necessary test equipment is available, perform the operator's check and system verification in Chapter

2588

11. **Phase lock lost message** - This usually occurs when there is not enough source power to phase lock the PNA. It can occur during an errant FCA setup or Source Power Calibration. It can also occur if one of the front panel reference channel loops is not connected. Otherwise, this indicates a hardware problem.

---

**PNA Application Terminates Unexpectedly**

If an unexpected and irrecoverable error occurs, Agilent would like to know about it. The PNA attempts to save pertinent information about the state of the system. The PNA does NOT send this information to Agilent.

We respect the privacy of our customers. However, access to information that helps us improve the PNA is a benefit to both Agilent and you. Please take the time to contact us or email the saved information to na_support@agilent.com.

The following procedure shows how to do this:

1. A message box immediately appears on the screen containing the location of a directory. Please record this message. If you miss the message, you can find the directory location using the Windows Event Log: On the PNA, click Start, Settings, Control Panel, Administrative Tools, Event Viewer. Double-click the top line (most recent event). The location of the directory is seen in the Description.

2. A dialog box may appear on the screen (shown below) allowing you to add comments to help us replicate the crash.

3. Find the directory (described in Step 1) which contains the following files:

   - 835x.dmp which is the 835x.exe capturing the context in which the program crashed.
   - 835x.xml which reports some very basic information (exception code, OS version, and the list of modules loaded at the time of the crash and their respective version numbers).
   - 835xCrashLog.txt: The text file with your comments (described in Step 2), if submitted.

4. If your PNA is not connected to LAN or is not configured to send email, copy the files to a PC. Then, please email the files to na_support@agilent.com
Check Error Terms

If you print the error terms at set intervals (weekly, monthly, and so forth), you can compare current error terms to these records. A stable, repeatable system should generate repeatable error terms over long time intervals, for example, six months. If a subtle failure or mild performance problem is suspected, the magnitude of the error terms should be compared against values generated previously with the same instrument and calibration kit. See the procedure for monitoring error terms.

- A long-term trend often reflects drift, connector and cable wear, or gradual degradation, indicating the need for further investigation and preventative maintenance. Yet, the system may still conform to specifications. The cure is often as simple as cleaning and gaging connectors or inspecting cables.

- A sudden shift in error terms reflects a sudden shift in systematic errors, and may indicate the need for further troubleshooting.

Consider the following while troubleshooting:

- All parts of the system, including cables and calibration devices, can contribute to systematic errors and impact the error terms.

- Connectors must be clean and gauged, and within specification for error term analysis to be meaningful. See the Chapter 2 in the PNA Service Guide for information on cleaning and gaging connectors.

  - Avoid unnecessary bending and flexing of the cables following measurement calibration, thus minimizing cable instability errors.

  - Use good connection techniques during the measurement calibration. The connector interface must be repeatable. See the PNA Service Guide for information on connection techniques.

- It is often worthwhile to perform the procedure twice (using two distinct measurement calibrations) to establish the degree of repeatability. If the results do not seem repeatable, check all connectors and cables.
• Use error-term analysis to troubleshoot minor, subtle performance problems. See Chapter 3, "Troubleshooting," in the PNA Service Guide if a blatant failure or gross measurement error is evident.

**Check the Service Guide**


---

Last modified:

10/16/06    Added phase lock lost
PNA Error Messages

- 500 - 750  Calibrate
- 770 - 1000  Hardware
- 1000 - 1200  Measure
- 1281 - 1535  Parser
- 1536 - 1650  Display
- 1700 - 2000  Channel
- 2048 - 2200  General
- Standard SCPI Errors

Note: The EventID's listed below are provided for COM programming. For more information, see Working with PNA Events.

For more information on PNA error messages (see Error Messages).

Cal Errors

Message: 512
"A secondary parameter (power, IFBW, sweeptime, step mode) of the calibrated state has changed."
Severity: Informational

Further explanation: The calibration is questionable when any of these secondary parameters change after the calibration is performed.

Suggestions: If you require an accurate measurement with the new settings, repeat the calibration.

EventID: 68020200 (hex)

Message: 513
"Calibration cannot be completed until you have measured all the necessary standards for your selected Cal Type."
Severity: Informational

Further explanation: You probably received this message because you attempted to turn correction on without first measuring all of the calibration standards

Suggestions: Finish measuring the cal standards

EventID: 68020201 (hex)

Message: 514
"Calibration set has been recalled using a file previously saved on an analyzer that had a different hardware configuration."
Severity: Informational

Further explanation:

Suggestions:

EventID: 68020202 (hex)

**Message: 515**

"Calibration is required before correction can be turned on. Channel number is <x>, Measurement is <x>."

Severity: Informational

Further explanation: There are no error correction terms to apply for the specified channel and measurement.

Suggestions: Perform or recall a calibration

EventID: 68020203 (hex)

**Message: 516**

"Critical parameters in your current instrument state do not match the parameters for the calibration set, therefore correction has been turned off. The critical instrument state parameters are sweep type, start frequency, frequency span, and number of points."

Severity: Informational

Further explanation: None

Suggestions: You can either recalibrate using the new settings or change back to the original setting that was used when the calibration was performed.

EventID: 68020204 (hex)

**Message: 517**

"Interpolation is turned off and you have changed the stimulus settings of the original calibration, so correction has been turned off."

Severity: Informational

Further explanation: The most accurate calibration is maintained only when the original stimulus settings are used.

Suggestions: If reduced accuracy is OK, set interpolation ON to allow stimulus setting changes.

EventID: 68020205 (hex)

**Message: 518**

"Interpolation is turned off and you have selected correction ON. Correction has been restored with the previous stimulus settings."

Severity: Informational

Further explanation: None

Suggestions: None

EventID: 68020206 (hex)

**Message: 519**

"Stimulus settings for your current instrument state exceeded the parameters of the original calibration, so
correction has been turned off."

**Severity:** Informational

**Further explanation:** Correction data outside the stimulus settings does not exist.

**Suggestions:** Perform a broadband calibration, with increased numbers of points with interpolation ON, to maintain calibration over the widest possible stimulus frequency settings.

**EventID:** 68020207 (hex)

**Message: 520**

"Cal Type is set to NONE for Channel <x>, Measurement <x>; please select Calibration menu or press Cal hard key."

**Severity:** Informational

**Further explanation:** A cal operation can not proceed until a calibration exists or the cal type is selected. This error can occur if the calibration can not be found. Also this error can happen if a calibration type is not specified before attempting to programatically execute cal acquisitions.

**Suggestions** To find a calibration, select a Cal Set that contains the calibration needed for the current measurements. OR specify the cal type before beginning a calibration procedure.

**EventID:** 68020208 (hex)

**Message: 521**

"The measurement you set up does not have a corresponding calibration type, so correction has been turned off or is not permitted."

**Severity:** Informational

**Further explanation:** The calibration for the channel may apply only to certain S-Parameters. For example, a 1-Port calibration for S11 can not be applied to a 1-Port calibration applied to S22.

**Suggestions** Select a calibration type, such as full 2-Port cal, that can be applied to all the measurements to be selected.

**EventID:** 68020209 (hex)

**Message: 522**

"The calibration type you selected cannot be set up."

**Severity:** Informational

**Further explanation:** "Please use the SCPI command ROUTe:PATH:DEFine:PORT <num>,<num> for full 2 port type port assignment."

**Suggestions:**

**EventID:** 6802020A (hex)

**Message: 523**

"The calibration path you selected cannot be set up because it is not valid for the current measurement."

**Severity:** Informational

**Further explanation:** "Please use the SCPI command ROUTe:PATH:DEFine:PORT <num>,<num> for full 2 port type port assignment related to your current measurement."
Suggestions:
EventID: 6802020B (hex)

Message: 524
"The source power calibration is complete."
Severity: Informational
Further explanation:
Suggestions:
EventID: 6802020C (hex)

Message: 525
"You have specified more than 7 standards for one or more calibration classes."
Severity: Informational
Further explanation: These have been truncated to 7 selections.
EventID: 6802020D (hex)

Message: 526
"No user calibration found for this channel."
Severity: Informational
Further explanation: A cal operation can not proceed until a calibration exists.
Suggestions: To find a calibration, you can select a Cal Set that contains the calibration needed for the current measurement.
EventID: 6802020E (hex)

Message: 527
"You do not need to acquire this standard for this calibration type."
Severity: Informational
Further explanation: This error can happen as a result of PROGRAMMATICALLY requesting the measurement of an un-needed calibration standard during a calibration procedure.
Suggestions: Check the specified cal type or eliminate the request for the measurement of the standard.
EventID: 6802020F (hex)

Message: 528
"Could not configure the Electronic Calibration system. Check to see if the module is plugged into the proper connector."
Severity: Informational
Further explanation: During an ECal operation, communication could not be established with the ECal module. The calibration will not be initiated until the presence of the ECal module is verified.
Suggestions: Verify the USB cable is connected properly. Disconnect and re-connect the cable to ensure the analyzer recognizes the module.
EventID: 68020210 (hex)

Message: 529
"DATA OUT OF RANGE: Design Limits Exceeded"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020211 (hex)

Message: 530
"EXECUTION ERROR: Could not open ECal module memory backup file"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020212 (hex)

Message: 531
"EXECUTION ERROR: Access to ECal module memory backup file was denied"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020213 (hex)

Message: 532
"EXECUTION ERROR: Failure in writing to ECal module memory backup file"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020214 (hex)

Message: 533
"EXECUTION ERROR: Failure in reading from ECal module memory backup file"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020215 (hex)

Message: 534
"EXECUTION ERROR: Array index out of range"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020216 (hex)

Message: 535
"EXECUTION ERROR: Arrays wrong rank"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020217 (hex)

Message: 536
"EXECUTION ERROR: CPU"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020218 (hex)

Message: 537
"EXECUTION ERROR: Cannot ERASE module"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020219 (hex)

Message: 538
"EXECUTION ERROR: Cannot WRITE module"
Severity: Error
Further explanation:
Suggestions:
EventID: E802021A (hex)

Message: 539
"EXECUTION ERROR: Entry Not Found"
Severity: Error
Further explanation:
Suggestions:
EventID: E802021B (hex)
**Message: 540**
"EXECUTION ERROR: Invalid command while system is busy"

**Severity:** Error

**Further explanation:**

**Suggestions:**

**EventID:** E802021C (hex)

**Message: 541**
"Electronic Cal: Unable to orient ECal module. Please ensure the module is connected to the necessary measurement ports."

**Severity:** Error

**Further explanation:** There is no RF connection to the ECal module during a calibration step. An ECal orientation measurement has been attempted but the signal was not found.

**Suggestions:** Connect the ECal module RF connections to ports specified for the calibration step. The ECal module typically requires at least -18dBm for measurements. If your measurement requires the power level to be less than that, clear the **Do orientation** checkbox to bypass the automatic detection step.

**EventID:** E802021D (hex)

**Message: 542**
"EXECUTION ERROR: NO SPACE for NEW CAL, DELETE A CAL"

**Severity:** Error

**Further explanation:**

**Suggestions:**

**EventID:** E802021E (hex)

**Message: 543**
"EXECUTION ERROR: No More Room"

**Severity:** Error

**Further explanation:**

**Suggestions:**

**EventID:** E802021F (hex)

**Message: 544**
"EXECUTION ERROR: Other array error"

**Severity:** Error

**Further explanation:**

**Suggestions:**

**EventID:** E8020220 (hex)

**Message: 545**
"EXECUTION ERROR: Ranks not equal"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020221 (hex)

Message: 546
"EXECUTION ERROR: Too few CONSTANT ranks"
Severity: Error
EventID: E8020222 (hex)

Message: 547
"EXECUTION ERROR: Too few VARYing ranks"
Severity: Error
EventID: E8020223 (hex)

Message: 548
"EXECUTION ERROR: Unknown error"
Severity: Error
EventID: E8020224 (hex)

Message: 549
"EXECUTION ERROR: ecaldrvr.dll bug or invalid module #"
Severity: Error
EventID: E8020225 (hex)

Message: 550
"EXECUTION ERROR: unexpected error code from ecal driver"
Severity: Error
EventID: E8020226 (hex)

Message: 551
"EXECUTION ERROR: unexpected internal driver error"
Severity: Error
EventID: E8020227 (hex)

Message: 552
"HARDWARE ERROR: Can't access ECal Interface Module"
Severity: Error
EventID: E8020228 (hex)
Message: 553
"HARDWARE ERROR: Can't release LPT port, reboot"
Severity: Error
EventID: E8020229 (hex)

Message: 554
"HARDWARE ERROR: VNA Error"
Severity: Error
EventID: E802022A (hex)

Message: 555
"HARDWARE ERROR: not enough data read from ECal module"
Severity: Error
EventID: E802022B (hex)

Message: 556
"OPERATION ABORTED BY HOST COMPUTER"
Severity: Error
EventID: E802022C (hex)

Message: 557
"OPERATION ABORTED BY USER"
Severity: Error
EventID: E802022D (hex)

Message: 558
"OUT OF MEMORY"
Severity: Error
EventID: E802022E (hex)

Message: 559
"QUERY INTERRUPTED:Message(s Abandoned"
Severity: Error
EventID: E802022F (hex)

Message: 560
"QUERY UNTERMINATED: INCOMPLETE PROGRAM Message"
Severity: Error
Further explanation:
Suggestions:
EventID: E8020230 (hex)

**Message: 561**
"QUERY UTERMINATED: NOTHING TO SAY"
Severity: Error

Further explanation:

Suggestions:
EventID: E8020231 (hex)

**Message: 562**
"QUEUE OVERFLOW"
Severity: Error
EventID: E8020232 (hex)

**Message: 563**
"SETTINGS CONFLICT: ADDITIONAL STANDARDS ARE NEEDED"
Severity: Error
EventID: E8020233 (hex)

**Message: 564**
"SETTINGS CONFLICT: Adapter Cal is NOT possible"
Severity: Error
EventID: E8020234 (hex)

**Message: 565**
"SETTINGS CONFLICT: COMMAND OUT OF SEQUENCE"
Severity: Error
EventID: E8020235 (hex)

**Message: 566**
"SETTINGS CONFLICT: Cal STOPPED - VNA SETUP CHANGED"
Severity: Error
EventID: E8020236 (hex)

**Message: 567**
"SETTINGS CONFLICT: Calibration is NOT in progress"
Severity: Error
EventID: E8020237 (hex)
Message: 568
"SETTINGS CONFLICT: Can't find specified GPIB board"
Severity: Error
EventID: E8020238 (hex)

Message: 569
"SETTINGS CONFLICT: Can't find/load gpib32.dll"
Severity: Error
EventID: E8020239 (hex)

Message: 570
"SETTINGS CONFLICT: Can't find/load sicl32.dll"
Severity: Error
EventID: E802023A (hex)

Message: 571
"SETTINGS CONFLICT: Can't initialize VNA (bad address?)"
Severity: Error
EventID: E802023B (hex)

Message: 572
"SETTINGS CONFLICT: Can't load LPT port driver or USB driver DLL"
Severity: Error
EventID: E802023C (hex)

Message: 573
"SETTINGS CONFLICT: Invalid Calibration Sweep Mode."
Severity: Error
EventID: E802023D (hex)

Message: 574
"SETTINGS CONFLICT: Invalid Calibration Type"
Severity: Error
EventID: E802023E (hex)

Message: 575
"SETTINGS CONFLICT: Invalid Calibration"
Severity: Error
EventID: E802023F (hex)
Message: 576
"SETTINGS CONFLICT: Invalid GPIB board number specified"
Severity: Error
EventID: E8020240 (hex)

Message: 577
"SETTINGS CONFLICT: Invalid GPIB board type specified"
Severity: Error
EventID: E8020241 (hex)

Message: 578
"SETTINGS CONFLICT: Invalid Module Status"
Severity: Error
EventID: E8020242 (hex)

Message: 579
"SETTINGS CONFLICT: Invalid States"
Severity: Error
EventID: E8020243 (hex)

Message: 580
"SETTINGS CONFLICT: LPT port must be between 1 and 4"
Severity: Error
EventID: E8020244 (hex)

Message: 581
"Could not configure the Electronic Calibration system. Check to see if the module is properly connected."
Severity: Error
EventID: E8020245 (hex)

Message: 582
"SETTINGS CONFLICT: Specified LPT port does not exist"
Severity: Error
EventID: E8020246 (hex)

Message: 583
"SETTINGS CONFLICT: Use frequency domain for cal"
Severity: Error
EventID: E8020247 (hex)
Message: 584
"SETTINGS CONFLICT: Use step sweep type for cal."
Severity: Error
EventID: E8020248 (hex)

Message: 585
"SETTINGS CONFLICT: VNA address must be between 0 and 30"
Severity: Error
EventID: E8020249 (hex)

Message: 586
"SETTINGS CONFLICT: Wrong LPT port driver or USB driver DLL"
Severity: Error
EventID: E802024A (hex)

Message: 587
"SYNTAX ERROR: ECAL:DELAY command must have 2 numbers"
Severity: Error
EventID: E802024B (hex)

Message: 588
"SYNTAX ERROR: INCORRECT SYNTAX"
Severity: Error
EventID: E802024C (hex)

Message: 589
"SYNTAX ERROR: UNKNOWN COMMAND"
Severity: Error
EventID: E802024D (hex)

Message: 590
"Wrong port of module in RF path"
Severity: Error
EventID: E802024E (hex)

Message: 591
"User characterization not found in module"
Severity: Error
EventID: E802024F (hex)
Message: 592
Severity: Informational
"No source power calibration found for the channel and source port of the current measurement."
Further explanation: You tried to turn on source power cal but there is no source power cal data.
Suggestions: Perform a source power calibration
EventID: 68020250 (hex)

Message: 593
Severity: Informational
"A source power calibration sweep was not performed, so there is no correction for the channel and source port of the current measurement."
Further explanation: You tried to turn on source power cal but there is incomplete source cal data.
Suggestions: Perform a complete source power calibration
EventID: 68020251 (hex)

Message: 594
Severity: Informational
"A new trace could not be added to the active window for viewing the source power cal sweep, because it would have exceeded the limit on number of traces/window. Please remove a trace from the window before proceeding with source power cal."
Further explanation: The source power cal attempts to add a data trace to the active window. The active window already contains four traces.
Suggestions: Make the active window contain less than four traces.
EventID: 68020252 (hex)

Message: 595
Severity: Informational
"A new measurement could not be added for performing the source power cal sweep, because the limit on number of measurements has been reached. Please remove a measurement before proceeding with source power cal."
Further explanation: The source power cal attempts to add a measurement. The PNA already has the maximum number of measurements.
Suggestions: Delete a measurement.
EventID: 68020253 (hex)

Message: 596
Severity: Informational
"The calibration power value associated with the source power calibration of Port %1 on Channel %2 was changed with the calibration on. The calibration was not turned off, but the power value might no longer represent the calibration."
Further explanation: The source power cal accuracy is questionable.
Suggestions: If high accuracy is required, perform another source power calibration.
EventID: 68020254 (hex)

**Message: 597**

**Severity:** Informational

- Message that is passed from the power meter driver for a source power calibration.

**Further explanation:** This error is generated by the power meter driver and passed through the PNA.

EventID: 68020255 (hex)

**Message: 598**

"During the acquisition of the sliding load standard, the slide was not properly moved to perform a circle fit. The standard's raw impedance was used to determine the directivity for one or more points."

**Severity:** Informational

**Further Explanation:** To accurately characterize the standard, the sliding load must be move sufficiently to ensure enough samples around the complex circle or Smith Chart. Under-sampling will cause an inaccurate result.

**Suggestions:** For best results when using a sliding load, be sure to use multiple slide positions that cover the full range of movement from front to back of the slot.

EventID: 68020256 (hex)

**Message: 599**

"This feature requires an unused channel, but could not find one. Please free up a channel and try again."

**Severity:** Informational

**Further Explanation:** You attempted to view an item within a calset. However, the calset viewer requires that the result be displayed in a channel that is not currently in use. All the channels are currently used. The view can not display the requested item.

**Suggestions:** You must delete at least one channel that is currently in use.

EventID: 68020257 (hex)

**Message: 600**

"Interpolation of the original calibration is not allowed since it was performed using Segment Sweep. Correction has been turned off."

**Severity:** Informational

EventID: 68020258 (hex)

**Message: 601**

"Cal preferences saved. Cal preference settings can be changed from the 'Cal Preferences' drop down Cal menu."

**Severity:** Informational

**Further explanation:** See Save Preference

EventID: 68020259 (hex)

**Message: 608**

"CalType not set."
Severity: Error  

Further explanation: A cal operation can not proceed until a calibration exists or the proper cal type is selected.

Suggestions: This error can happen if the calibration can't be found. To find a calibration, you can select a Cal Set that contains the calibration needed for the current measurements. This error can also happen if a calibration type is not specified before attempting to programmatically execute cal acquisitions. Specify the cal type before beginning a calibration procedure.

EventID: E8020260 (hex)

Message: 609

"The Calibration feature requested is not implemented."

Further explanation: The specified cal type can be one of many choices. For example, response calibrations require single standards, 1-Port calibrations require 3 standards, and 2-Port calibrations require up to 12 standards.

Suggestions: Be sure to measure only the standards needed for the specified cal type.

EventID: E8020261 (hex)

Message: 610

"The Calibration Class Acquisition requested is not valid for the selected Calibration Type. Please select a different acquisition or a different Calibration Type."

EventID: E8020262 (hex)

Message: 611

"The Calibration Standard data required for the selected caltype was not found."

Severity: Error

Further explanation: An unsuccessful attempt was made to retrieve a specified standard from the raw measurement buffer. The buffer should contain the raw measurements of cal standards stored during a calibration procedure.

Suggestions: Be sure the requested standard is required for the current cal type. Not all standards are needed for all cal types.

EventID: E8020263 (hex)

Message: 612

" The Error Term data required for the selected caltype was not found."

Severity: Error

Further explanation: An unsuccessful attempt was made to retrieve a specified error term from the error correction buffer. The buffer should contain the error correction arrays for the current calibration.

Suggestions: Be sure the requested error term is required for the current cal type. Not all error terms are needed for all cal types.

EventID: E8020264 (hex)

Message: 613

The Calibration data set was not found.

Severity: Error
Further explanation: An unsuccessful attempt to access a cal set has been made. This may indicate a calset has been deleted or has been corrupted.

Suggestions: Try again or select another cal set. If the cal set appears in the cal set list, it may need to be deleted.

EventID: E8020265 (hex)

Message: 614
"The specified measurement does not have a calibration valid for Confidence Check. Please select a different measurement, or recall or perform a different Calibration Type."

Severity: Error

Further explanation: The measurement choice is prevented so that calibration will not be turned off. Not all cal types support all measurements. For example, an 1-Port cal on S11 can not be used to calibrate an S12 measurement. When a measurement is selected that does not have a calibration which can be applied, an informational message is displayed and calibration is turned off.

Suggestions: Use a full 2-Port calibration to be compatible with any S-Parameter.

EventID: E8020266 (hex)

Message: 615
" New calset created."

Severity: Informational message.

Further explanation: The newly created cal set will be automatically named and time stamped. If this is the beginning of a calibration procedure, the cal set will not be stored to memory until the calibration has completed successfully. The new cal set will be deleted if the calibration is canceled or does not otherwise complete successfully.

Suggestions: Informational

EventID: 68020267

Message: 617
The calset file: <x> appears to be corrupted and cannot be removed. Exit the application, remove the file, and restart.

Severity: Error

Suggestions: The cal set file is stored in the application home directory C:/Program Files/Agilent/Network Analyzer/PNACalSets.dat. Remove this file, then restart the application.

EventID: E8020269 (hex)

Message: 634
"The calset file: <x> load failed."

Severity: Error


Suggestions: Try restarting the application. If the failure persists, you may have to delete the cal set data file and restart the application. The cal set file is stored in the application home directory C:/Program Files/Agilent/Network Analyzer/PNACalSets.dat. Remove this file, then restart the application.
EventID: E802027A (hex)

**Message:** 635

"The calset file: <x> save failed."

**Severity:** Error

**Further explanation:** The file operation detected an error. The save operation was aborted.

**Suggestions:** Retry.

EventID: E802027B (hex)

**Message:** 636

"A calset was deleted."

**Severity:** Informational

**Further explanation:** One of the calsets has been successfully deleted from the collection of calsets available. This can happen as the result of a user request or intentional operation.

**Suggestions:** None

EventID: 6802027C (hex)

**Message:** 637

"The version of the calset file: <x> is not compatible with the current instrument."

**Severity:** Error

**Further explanation:** A versioning error can prevent a calset from being used. This can happen as a result of instrument firmware upgrades.

**Suggestions:** If the versioning error is the result of firmware upgrade, you will have to re-install the old version of firmware to re-use the calset file. Or you can re-create the calsets with the current version of firmware.

The cal set file is stored in the application home directory C:/Program Files/Agilent/Network Analyzer/PNACalSets.dat. Remove this file, then restart the application.

EventID: E802027D (hex)

**Message:** 638

"Incompatible CalSets found: <x> of <y> stored calsets have been loaded."

**Severity:** Error

**Further explanation:** Errors were found on some of the calsets stored in the calset file. The errors may have been caused by versioning issues that may have corrupted the various calset keys.

**Suggestions:** Use the calset viewer to look at the contents of calset files. Delete the files that are corrupted.

EventID: 6802027E (hex)

**Message:** 639

"The Calset file: <x> was not found. A new file has been created."

**Severity:** Informational

**Further explanation:** The calset file should be stored on the hard drive. When the application is started, a search is done and the file is loaded if it can be found. If the file is not found, the analyzer will create a new file and display..."
this message.

**Suggestions:** None
**EventID:** E802027F (hex)

**Message:** 640
"The Calset specified is currently in use."

**Severity:** Error

**Further explanation:** This may indicate a conflict between multiple calset users attempting calibration tasks.

**Suggestions:** Save the instrument state. Preset the analyzer and recall the instrument state. This may abort any processes that may be in progress.
**EventID:** E8020280 (hex)

**Message:** 641
"The calset specified has not been opened."

**Severity:** Error

**Further explanation:** Multiple users may be attempting to access the calset.

**Suggestions:** Close multiple calset users so that only one user will access the calset.
**EventID:** E8020281 (hex)

**Message:** 642
"The maximum number of cal sets has been reached. Delete old or unused cal sets before attempting to create new ones."

**Severity:** Error

**Suggestions:** You may also delete the calsets data file.

The cal set file is stored in the application home directory. C:/Program Files/Agilent/Network_Analyzer/PNACalSets.dat. Remove this file, then restart the application.
**EventID:** E8020282 (hex)

**Message:** 643
The requested power loss table segment was not found.

**Severity:** Error
**EventID:** E8020283 (hex)

**Message:** 644
"A valid calibration is required before correction can be turned on."

**Severity:** Error

**Further explanation:** This usually indicates a calibration procedure has not run to completion or that the selected measurement does not have a valid calibration available from within the currently selected cal set.

**Suggestions:** To find a calibration, you can select a Cal Set that contains the calibration needed for the current measurements. This error can happen if a calibration type is not specified before attempting to programmatically execute cal acquisitions. Specify the cal type before beginning a calibration procedure.
EventID: E8020284 (hex)

**Message: 645**
The cal data for <x> is incompatible and was not restored. Please recalibrate.

**Severity:** Warning

**Further explanation:** None

**Suggestions:** None

EventID: A8020285 (hex)

**Message: 646**
"CalSet not loaded, version is too new."

**Severity:** Error

**Further explanation:** An old version of firmware is attempting to run with a new calset version. The version is incompatible.

**Suggestions:** The calset can be removed. You may also delete the calsets data file if you are migrating between various firmware revisions often and you would like to avoid this error. The cal set file is stored in the application home directory. C:/Program Files/Agilent/Network Analyzer/PNACalSets.dat. Remove this file, then restart the application.

EventID: E8020286 (hex)

**Message: 647**
"Custom cal type not found."

**Severity:** Error

**Further explanation:**

**Suggestions:**

EventID: E8020287 (hex)

**Message: 648**
"Custom correction algorithm defers to the client for interpolation."

**Severity:** Informational

EventID: 68020288 (hex)

**Message: 649**
"Custom cal dll threw an exception."

**Severity:** Error

EventID: E8020289 (hex)

**Message: 650**
"Could not load the ecal.dll library"

**Severity:** Error
EventID: E802028A (hex)

Message: 656
"The argument specified is not a valid cal type."
Severity: Error
EventID: E8020290 (hex)

Message: 657
"The function found existing interpolated data"
Severity: Informational
EventID: 68020291 (hex)

Message: 658
"The function computed new interpolation values."
Severity: Informational
EventID: 68020292 (hex)

Message: 659
"The source power measurement failed."
Severity: Error
Suggestions: Please check GPIB, power meter settings and sensor connections.
EventID: E8020293 (hex)

Message: 660
"Duplicate session found. Close session and retry."
Severity: Error
EventID: E8020294 (hex)

Message: 661
"The session does not exist. Open the session and try again."
Severity: Error
Further explanation:
EventID: E8020295 (hex)

Message: 662
"Attempt to launch a custom calibration failed."
Severity: Error
Further explanation:
EventID: E8020296 (hex)
Message: 663
"Request to measure a cal standard failed."

Severity: Error

Further explanation: Please ensure you are requesting to measure standards which are defined for this calibration.

EventID: E8020297 (hex)

Message: 664
"Since Electronic Calibration Kit is selected, Mechanical Cal Kit parameter cannot be changed."

Severity: Error

Further explanation:

EventID: E8020298 (hex)

Message: 665
"Frequencies of the active channel are below minimum or above maximum frequencies of the ECal module factory characterization."

Suggestions: Change the channel frequencies, or select another ECal module.

Severity: Error

EventID: E8020299 (hex)

Message: 666
"Calset chosen for characterizing the ECal Module Ports %1 does not contain a calibration for PNA Ports %2."

Severity: Error

Suggestions: Go back to select another calset or to perform another cal.

EventID: E802029A (hex)

Message: 667
"ECal module only has sufficient memory remaining to store a maximum of %1 points in User Characterization %2."

Severity: Error

Suggestions: Decrease your number of points, or choose to overwrite another user characterization.

EventID: E802029B (hex)

Message: 668
Input values are non-monotonic. Cannot interpolate.

Severity: Error

EventID: E802029C (hex)

Message: 669
Interpolation target is out of range. Cannot interpolate.

Severity: Error
EventID: E802029D (hex)

**Message: 670**
Guided Calibration Error: <>
**Severity**: Error
**EventID**: E802029E (hex)

**Message: 671**
The first call to the guided calibration interface must be Initialize.
**Severity**: Error
**EventID**: E802029F (hex)

**Message: 672**
The selected thru cal method was not recognized.
**Severity**: Error
**EventID**: E80202A0 (hex)

**Message: 673**
Could not generate the error terms.
**Severity**: Error
**EventID**: E80202A1 (hex)

**Message: 674**
Guided calibration must be performed on the active channel
**Severity**: Error
**EventID**: E80202A2 (hex)

**Message: 675**
You can not start using calibration steps until you have successfully called `generate steps`.
**Severity**: Error
**EventID**: E80202A3 (hex)

**Message: 676**
The step number given is out of range. Step numbers should be between 1 and the number of steps. 0 is not a valid step number.
**Severity**: Error
**EventID**: E80202A4 (hex)

**Message: 677**
A calset was selected for channel: <n> without restoring stimulus.
**Severity**: Informational
**EventID:** 680202A5 (hex)

**Message:** 678
A calset was selected for channel: <n> restoring stimulus.
**Severity:** Informational

**EventID:** 680202A6 (hex)

**Message:** 679
The selected calset stimulus could not be applied to the channel.
**Severity:** Informational

**EventID:** 680202A7 (hex)

**Message:** 680
You attempted to measure power at a frequency outside the frequency range defined for the specified power sensor. Select another sensor or adjust the range for this sensor.
**Severity:** Error

**EventID:** E80202A8 (hex)

**Message:** 681
Specified frequency is outside the frequency ranges currently defined for the power meter's sensors.
**Severity:** Error

**EventID:** E80202A9 (hex)

**Message:** 682
Additional Calibration Standards need to be acquired in order to calibrate over the entire frequency range currently being measured.
**Severity:** Informational

**EventID:** 680202AA (hex)

**Message:** 683
The PNA failed to convert cal kits for use by unguided calibrations. The recommended action is to restore Cal Kit defaults.
**Severity:** Error

**EventID:** E80202AB (hex)

**Message:** 684
The PNA failed to convert cal kits for use by unguided calibrations. CalKit defaults have been restored.
**Severity:** Error

**EventID:** E80202AC (hex)

**Message:** 685
Power meter is reserved by a source power cal acquisition already in progress.

**Severity**: Error  
**EventID**: E80202AD (hex)

**Message: 686**  
Source power calibration has not been performed or uploaded for the specified channel and source port.

**Severity**: Error  
**EventID**: E80202AE (hex)

**Message: 687**  
Source power calibration data array size for the specified channel and source port does not match it's associated stimulus number of points.

**Severity**: Error  
**EventID**: E80202AF (hex)

**Message: 688**  
Source power calibration of Port <n> on Channel <n> was turned off because the correction array no longer exists.

**Severity**: Error  
**EventID**: E80202B0 (hex)

**Message: 689**  
This command can only be used on a measurement created with a specified calibration loadport.

**Severity**: Error  
**EventID**: E80202B1 (hex)

**Message: 690**  
Interpolation is turned off and you have changed the stimulus settings of the original calibration, so correction has been turned off.

**Severity**: Error  
**EventID**: E80202B2 (hex)

**Message: 691**  
Stimulus settings for your current instrument state exceeded the parameters of the original calibration, so correction has been turned off.

**Severity**: Error  
**EventID**: E80202B3 (hex)

**Message: 692**  
Fixturing: the requested S2P file cannot be read. Possible formatting problem.

**Severity**: Error  
**EventID**: E80202B4 (hex)
**Message: 693**
Fixturing: the requested S2P file cannot be opened.
**Severity:** Error
**EventID:** E80202B5 (hex)

**Message: 694**
Fixturing: the requested S2P file cannot be interpolated. This is usually because the frequency range in the file is a subset of the current channel frequency range.
**Severity:** Error
**EventID:** E80202B6 (hex)

**Message: 695**
Cal Registers can only be used by one channel: the channel conveyed in the name of the cal register. The name cannot be changed.
**Severity:** Error
Further explanation: See [Cal Registers](#)
**EventID:** E80202B7 (hex)

**Message: 696**
Fixturing: cannot be enabled with Response Calibrations and has been turned off.
**Severity:** Error
**EventID:** E80202B8 (hex)

**Message: 697**
The selected calibration cannot be performed for this measurement.
**Severity:** Error
**EventID:** E80202B9 (hex)

**Message: 698**
Fitting: RemoveAllConnectors() should be called prior to calling AddConnector after a fit has been attempted.
**Severity:** Error
**EventID:** E80202BA (hex)

**Message: 699**
An attempt was made to acquire calibration data before the system was properly initialized.
**Severity:** Error
**EventID:** E80202BB (hex)

**Message: 700**
Use IGuidedCalibration for multiport calibration types.
Severity: Error
EventID: E80202BC (hex)

Message: 701
Guided calibration requires number of thru measurement paths be at least equal to the number of calibration ports minus 1.

Severity: Error
EventID: E80202BD (hex)

Message: 702
A thru path was specified that includes a port which the calibration was not specified to include.

Severity: Error
EventID: E80202BE (hex)

Message: 703
One or more of the ports to be calibrated was not found in the set of specified thru paths.

Severity: Error
EventID: E80202BF (hex)

Hardware Errors

Message: 770
Input power too high. Source power is off.

Severity: Warning
EventID: A8030302 (hex)

Message: 771
Source power restored.

Severity: Informational
EventID: 68030303 (hex)

Message: 772
"The spampnp.sys driver is not working. Check system hardware. ! Data will be simulated. !"

Severity: Error

Further explanation: The Network Analyzer application cannot locate the DSP board. Hardware or a driver may be malfunctioning. This is also common when attempting to run the Network Analyzer on a workstation.

EventID: E8030304 (hex)

Message: 773
"Instrument Serial Bus Not Working."
Severity: Error

Further explanation: The instrument EEPROM appears to contain either all ones or all zeros. A serial bus hardware failure prevents reading the EEPROM.

EventID: E8030305 (hex)

Message: 848
"Phase lock lost"

Severity: Error

Further explanation: The instrument source was not able to lock properly. This can be the result of broken hardware, poor calibration, or bad EEPROM values.

Suggestions: Perform source calibration. Click System / Service / Adjustments / Source Calibration

EventID: E8030350 (hex)

Message: 849
Phaselock restored.

Severity: Success

EventID: 0x28030351 (hex)

Message: 850
"Unknown hardware error."

Severity: Error

Further explanation: Hardware malfunctioned prevents communication with the DSP.

EventID: E8030352 (hex)

Message: 851
DSP communication lost.

Severity: Error

EventID: E8030353 (hex)

Message: 852
RF power off.

Severity: Error

EventID: E8030354 (hex)

Message: 853
RF power on.

Severity: Success

EventID: 28030355 (hex)

Message: 854
Hardware OK.

Message: 855
"Source unleveled."

Severity: Error

Further explanation: The source was unable to properly level at the requested power. The indicated power may not be accurate.

Suggestions: Try a different power level. Recalibrate source, if problem persists.

EventID: E8030357 (hex)

Message: 856
Source leveled.

Severity: Success

EventID: 28030358 (hex)

Message: 857
Input overloaded.

Severity: Error

EventID: E8030359 (hex)

Message: 858
Input no longer overloaded.

Severity: Success

EventID: 2803035A (hex)

Message: 859
"Yig calibration failed."

Severity: Error

Further explanation: Internal self-calibration of YIG oscillator tuning failed.

EventID: E803035B (hex)

Message: 860
Yig calibrated.

Severity: Success

EventID: 2803035C (hex)

Message: 861
"Analog ramp calibration failed."
Severity: Error
Further explanation: Internal analog sweep ramp calibration has failed.
EventID: E803035D (hex)

Message: 862
Analog ramp calibrated.
Severity: Success
EventID: 2803035E (hex)

Message: 863
Source temperature high.
Severity: Error
EventID: E803035F (hex)

Message: 864
Source temperature OK.
Severity: Success
EventID: 28030360 (hex)

Message: 865
"EEPROM write failed."
Severity: Error
Further explanation: Attempt to store calibration data to EEPROM has failed. There is a possible hardware failure.
EventID: E8030361 (hex)

Message: 866
EEPROM write succeeded.
Severity: Success
EventID: 28030362 (hex)

Message: 867
Attempted I/O write while port set to read only.
Severity: Error
Further explanation: Attempt to write to an I/O data port while the port set to input/read only.
Suggestions: Set data port to write/output before attempting to write to port.
EventID: E8030363 (hex)

Message: 868
" Attempted I/O read from write only port.
Severity: Error
Further explanation: Attempt to read from an I/O data port while the port set to output/write only.

Suggestions: Set data port to read/input before attempting to read from port.

EventID: E8030364 (hex)

Message: 869
Invalid hardware element identifier.
Severity: Error
EventID: E8030365 (hex)

Message: 870
Invalid gain level setting.
Severity: Error
EventID: E8030366 (hex)

Message: 871
Device driver was unable to allocate enough memory. Please try rebooting.
Severity: Error
EventID: E8030367 (hex)

Message: 872
DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 1
Severity: Error
EventID: E8030368 (hex)

Message: 873
DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 2
Severity: Error
EventID: E8030369 (hex)

Message: 874
DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 3
Severity: Error
EventID: E803036A (hex)

Message: 875
DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 4
Severity: Error
EventID: E803036B (hex)

Message: 876
DSP Error. Please Contact Agilent Support. Technical Information: DSP Type 5

Severity: Error
EventID: E803036C (hex)

Message: 910
The trigger connection argument was not recognized as valid by the firmware.
Severity: Error
EventID: 0xE803038E (hex)

Message: 911
The trigger connection specified does not support this trigger behavior
Severity: Error
EventID: E803038F (hex)

Message: 912
The trigger behavior specified was not recognized as valid by the firmware.
Severity: Error
EventID: E8030390 (hex)

Message: 913
The trigger connection specified does not physically exist on this network analyzer
Severity: Error
EventID: E8030391 (hex)

Message: 914
Cannot set "Accept Trigger Before Armed", since this hardware configuration does not support edge triggering.
Severity: Error
EventID: E8030392 (hex)

Message: 915
Cannot set "Trigger Output Enabled", since this hardware configuration does not support BNC2.
Severity: Error
EventID: E8030393 (hex)

Message: 916
Exceeded maximum trigger delay.
Severity: Error
EventID: E8030394 (hex)

Message: 917
Exceeded minimum trigger delay.

**Severity:** Error

**EventID:** E8030395 (hex)

---

### Measure Errors

**Message:** 1024

If you are going to display or otherwise use a memory trace, you must first store a data trace to memory.

**Severity:** Warning

**EventID:** A8040400 (hex)

**Message:** 1025

"The measurement failed to shut down properly. The application is in a corrupt state and should be shut down and restarted."

**Severity:** Error

**Further explanation:** This message is displayed if the PNA application becomes corrupt. If you continue to get this error, please call customer service

**EventID:** E8040401 (hex)

**Message:** 1026

The measurement failed to shut down properly. The update thread failed to exit properly.

**Severity:** Warning

**EventID:** A8040402 (hex)

**Message:** 1027

"Group Delay format with CW Time or Power sweeps produces invalid data."

**Severity:** Warning

**Further explanation:** Group Delay format is incompatible with single-frequency sweeps. Invalid data is produced.

**Suggestions:** Ignore the data or choose a different format or sweep type.

**EventID:** A8040403 (hex)

**Message:** 1028

**Severity:** Informational

"MSG_LIMIT_FAILED"

**Further explanation:** Limit line test failed.

**EventID:** 68040404 (hex)

**Message:** 1029

**Severity:** Informational
"MSG_LIMIT_PASSED"

**Further explanation:** Limit line test passed.

**EventID:** 68040405 (hex)

**Message: 1030**

"Exceeded the maximum number of measurements allowed."

**Severity:** Warning

**Further explanation:** See *Traces, Channels, and Windows on the PNA* for learn about maximum measurements.

**EventID:** A8040406 (hex)

**Message: 1031**

"Network Analyzer Internal Error. Unexpected error in AddNewMeasurement."

**Severity:** Warning

**Further explanation:** If you continue to get this message, contact product support.

**EventID:** A8040407 (hex)

**Message: 1032**

"No measurement was found to perform the selected operation. Operation not completed."

**Severity:** Warning

**Further explanation:** None

**Suggestions:** Create a measurement before performing this operation.

**EventID:** A8040408 (hex)

**Message: 1033**

The Markers All Off command failed.

**Severity:** Warning

**EventID:** A8040409 (hex)

**Message: 1034**

"A memory trace has not been saved for the selected trace. Save a memory trace before attempting trace math operations."

**Severity:** Warning

**Further explanation:** Must have a memory trace when trying to do Trace Math,

**EventID:** A804040A (hex)

**Message: 1035**

"MSG_SET_AVERAGE_COMPLETE"

**Severity:** Informational

**Further explanation:** Informational for COM programming. Averaging factor has been reached.

**EventID:** 6804040B (hex)
Message: 1036
"MSG_CLEAR_AVERAGE_COMPLETE"
Further explanation: Informational for COM programming. Averaging factor has NOT been reached.
EventID: 6804040C (hex)

Message: 1037
"Time Domain transform requires at least 3 input points. The transform has been deactivated."
Severity: Informational
Further explanation: None
Suggestions: Increase the number of points.
EventID: 6804040D (hex)

Message: 1038
Smoothing requires a scalar format, and has been deactivated.
Severity: Informational
EventID: 6804040E (hex)

Message: 1039
A receiver power calibration in this instrument state file cannot be recalled into this firmware version.
Severity: Warning
EventID: A804040F (hex)

Message: 1104
"Exceeded limit on number of measurements."
Severity: Error
Further explanation: See Traces, Channels, and Windows on the PNA for measurement limits.
EventID: E8040450 (hex)

Message: 1105
"Parameter not valid."
Severity: Error
Further explanation: A measurement parameter that was entered programmatically is not valid.
EventID: E8040451 (hex)

Message: 1106
"Measurement not found."
Severity: Error
Further explanation: Any of these could be the cause:
Trying to calibrate but already have maximum measurements.
Trying to do a confidence check but there is not a measurement.
Trying to create, activate, or alter a measurement through COM that has been deleted through the front panel.
Trying to use a trace name through programming that is not unique.

**EventID:** E8040452 (hex)

**Message:** 1107
"No valid memory trace."

**Severity:** Error

**Further explanation:** Must have a memory trace when trying to do Trace Math,

**Suggestions:** Store a memory trace.

**EventID:** E8040453 (hex)

**Message:** 1108
"The reference marker was not found."

**Severity:** Error

**Further explanation:** Attempted to create a delta marker without first creating a reference marker (COM only).

**EventID:** E8040454 (hex)

**Message:** 1109
"Data and Memory traces are no longer compatible. Trace Math has been turned off."

**Severity:** Error

**Further explanation:** Warning - channel setting has changed while doing trace math.

**Suggestions:** Store another memory trace and turn trace math back on.

**EventID:** A8040455 (hex)

**Message:** 1110
"Data and Memory traces are not compatible. For valid trace math operations, memory and data traces must have similar measurement conditions."

**Severity:** Error

**Further explanation:** Tried to do trace math without compatible data and memory traces.

**Suggestions:** Store another memory trace.

**EventID:** E8040456 (hex)

**Message:** 1111
"Marker Bandwidth not found."

**Severity:** Error

**Further explanation:** Could not find a portion of trace that meets the specified bandwidth criteria.

**EventID:** E8040457 (hex)

**Message:** 1112
"The peak was not found."

**Severity:** Error

**Further explanation:** Could not find portion of trace that meets peak criteria.

**Suggestions:** See Marker Peak criteria.

**EventID:** E8040458 (hex)

**Message:** 1113

"The target search value was not found."

**Severity:** Error

**Further explanation:** Could not find interpolated data point that meets search value.

**EventID:** E8040459 (hex)

**Message:** 1114

"Reflection measurement, such as S11, must supply an auxiliary port to disambiguate 2-port measurements on multiport instruments."

**Severity:** Error

**Further explanation:**

**EventID:** E804045A (hex)

**Message:** 1115

"The receiver power calibration has been turned off because the type of measurement or source port has changed, so the calibration is no longer valid."

**Severity:** Warning

**Further explanation:**

**EventID:** A804045B (hex)

**Message:** 1116

"Receiver power cal requires the active measurement to be of unratioed power."

**Severity:** Warning

**Further explanation:**

**EventID:** A804045C (hex)

**Message:** 1117

"There is currently no source power calibration associated with the channel and source port of the active measurement. A source power cal should be performed or recalled before performing a receiver power calibration."

**Severity:** Warning

**Further explanation:**

**EventID:** A804045D (hex)
**Message: 1118**
"The attempted operation can only be performed on a standard measurement type."

**Severity:** Error

**Further explanation:**

**EventID:** E804045E (hex)

**Message: 1119**
"The custom measurement cannot be loaded because it is not compatible with the Network Analyzer hardware."

**Severity:** Error

**Further explanation:**

**Suggestions:**

**EventID:** E804045F (hex)

**Message: 1120**
"The custom measurement cannot be loaded because it is not compatible with the Network Analyzer software."

**Severity:** Error

**Further explanation:**

**EventID:** E8040460 (hex)

**Message: 1121**
"The custom measurement load operation failed for an unspecified reason."

**Severity:** Error

**Further explanation:**

**EventID:** E8040461 (hex)

**Message: 1122**
"The custom measurement data processing has generated an unhandled exception, and will be terminated. The PNA software may be in an unstable state and it is recommended that the PNA software be shutdown and restarted."

**Severity:** Error

**Further explanation:**

**EventID:** E8040462 (hex)

**Message: 1123**
"The attempted operation can only be performed on a custom measurement type."

**Severity:** Error

**Further explanation:**

**EventID:** E8040463 (hex)

**Message: 1124**
"The requested custom measurement is not available."
**Severity**: Error

**Further explanation:**
**EventID**: E8040464 (hex)

**Message: 1125**
"The requested custom algorithm was not found."
**Severity**: Error

**Further explanation:**
**EventID**: E8040465 (hex)

**Message: 1126**
"Normalization cannot be turned on because the measurement does not have a valid divisor buffer."
**Severity**: Error

**Further explanation:**
**EventID**: E8040466 (hex)

**Message: 1127**
"The Raw Data requested by the measurement could not be provided."
**Severity**: Warning

**Further explanation:**
**EventID**: A8040467 (hex)

**Message: 1128**
"The selected Sweep Type does not allow Transform and Gating. Transform and Gating disabled."
**Severity**: Error

**Further explanation:**
**EventID**: E8040468 (hex)

**Message: 1129**
Memory trace can not be applied to this measurement
**Severity**: Error

**EventID**: E8040469 (hex)

**Message: 1130**
Normalization can not be applied to this measurement
**Severity**: Error

**EventID**: E804046A (hex)

**Message: 1131**
The data provided has an invalid number of points. It could not be stored

**Severity:** Error

**EventID:** E804046B (hex)

**Message:** 1132

The measurement stored in the save/recall state has an invalid version. It could not be loaded

**Severity:** Error

**EventID:** E804046C (hex)

**Message:** 1133

This data format argument for this operation must be "naDataFormat_Polar"

**Severity:** Error

**EventID:** E804046D (hex)

**Message:** 1134

This data format argument for this operation must be a scalar data format

**Severity:** Error

**EventID:** E804046E (hex)

**Message:** 1135

The memory trace is not valid for the current measurement setup.

**Severity:** Error

**EventID:** E804046F (hex)

**Message:** 1136

This measurement is incompatible with existing measurements in this channel. Choose another channel.

**Severity:** Error

**EventID:** E8040470 (hex)

**Message:** 1137

Port extension correction is not available for offset frequency measurements. Port extension correction has been disabled.

**Severity:** Error

**EventID:** E8040471 (hex)

**Message:** 1138

Physical port number assignments for logical port mappings must be unique.

**Severity:** Error

**EventID:** E8040472 (hex)
Message: 1281
"You have sent a read command to the analyzer without first requesting data with an appropriate output command. The analyzer has no data in the output queue to satisfy the request."
Severity: Error
EventID: 68050501 (hex)

Message: 1282
"You must remove the active controller from the bus or the controller must relinquish the bus before the analyzer can assume the system controller mode."
Severity: Error
EventID: E8050502 (hex)

Message: 1283
"The analyzer did not receive a complete data transmission. This is usually caused by an interruption of the bus transaction."
Severity: Error
EventID: E8050503 (hex)

Message: 1284
"The instrument status byte has changed."
Severity: Informational
EventID: 68050504 (hex)

Message: 1285
"The SCPI command received has caused error number %1: "%2"."
Severity: Informational
EventID: 68050505 (hex)

Message: 1286
"The INET LAN server has been started as process number %1."
Severity: Informational
EventID: 68050506 (hex)

Message: 1360
"Execution of the SCPI command has failed"
Severity: Error
EventID: E8050550 (hex)

Message: 1361
"The INET/LAN device is not accessible."

**Severity:** Error

**EventID:** E8050551 (hex)

**Message:** 1362

"The INET/LAN driver was not found."

**Severity:** Error

**EventID:** E8050552 (hex)

**Message:** 1363

"The INET/LAN driver was not found."

**Severity:** Error

**EventID:** E8050553 (hex)

**Message:** 1364

"The INET/LAN device is unable to acquire the necessary resources."

**Severity:** Error

**EventID:** E8050554 (hex)

**Message:** 1365

"The INET/LAN device generated a generic system error."

**Severity:** Error

**EventID:** E8050555 (hex)

**Message:** 1366

"Invalid address for the INET/LAN device."

**Severity:** Error

**EventID:** E8050556 (hex)

**Message:** 1367

"The INET I/O library was not found."

**Severity:** Error

**EventID:** E8050557 (hex)

**Message:** 1368

"An error occurred in the INET system."

**Severity:** Error

**EventID:** E8050558 (hex)

**Message:** 1369
"Access to the INET/LAN driver was denied."

**Severity:** Error  
**EventID:** E8050559 (hex)

**Message:** 1370

"Could not load error system message dll."

**Severity:** Error  
**EventID:** E805055A (hex)

**Message:** 1371

"ErrorSystemMessage.dll does not export the right function."

**Severity:** Error  
**EventID:** E805055B (hex)

**Message:** 1372

"Custom scpi library was not able to be knitted"

**Severity:** Error  
**EventID:** E805055C (hex)

**Message:** 1373

"Could not knit the scpi error messages from the ErrorSystemMessage lib"

**Severity:** Error  
**EventID:** E805055D (hex)

**Message:** 1374

Command is obsolete with this software version.

**Severity:** Error  
**EventID:** E808055E (hex)

**Message:** 1375

CALC measurement selection set to none. Use **Calc:Par:Sel**

**Severity:** Error  
**EventID:** E808055F (hex)

**Message:** 1535

"Parser got command: %1."

**Severity:** Informational  
**EventID:** 680505FF (hex)

**Display Errors 1536 - 1621**
Message: 1536
"Exceeded the maximum of 4 traces in each window. The trace for <x> will not be added to window <x>.”

Severity: Warning

Further explanation: None

Suggestions: Create the trace in another window. See the PNA window limits.

EventID: A8060600 (hex)

Message: 1537
"Exceeded the maximum of 16 data windows. New window will not be created."

Severity: Warning

Further explanation: None

Suggestions: Create the trace in an existing window. See the PNA window limits.

EventID: A8060601 (hex)

Message: 1538
"No Data Windows are present. Unable to complete operation."

Severity: Warning

Further explanation: Your remote SCPI operation tried to create a new measurement while there were no windows present

Suggestions: Create a new window before creating the measurement. See example Create a measurement using SCPI

EventID: A8060602 (hex)

Message: 1539
"No data traces are present in the selected window. Operation not completed."

Severity: Warning

Further explanation: None

EventID: A8060603 (hex)

Message: 1540
"Cannot complete request to arrange existing measurements in <x> windows due to the limit of <x> traces per window."

Severity: Informational

Further explanation: The arrange window feature cannot put the existing traces into the number of windows you requested because only 4 traces per window are allowed. See Arranging Existing Measurements

Suggestions: Either create more windows or delete some traces.

EventID: 68060604 (hex)

Message: 1541
"Unable to establish a connection with the specified printer."

**Severity:** Warning

**Further explanation:** None

**Suggestions:** Refer to Printer Help

**EventID:** A8060605 (hex)

**Message:** 1542

"Printout canceled."

**Severity:** Informational

**EventID:** 68060606 (hex)

**Message:** 1616

"Window not found."

**Severity:** Error

**Further explanation:** A window was specified in your program which does not exist.

**Suggestions:** Query the name of your window before specifying.

**EventID:** E8060650 (hex)

**Message:** 1617

"Duplicate window ID specified."

**Severity:** Error

**Further explanation:** None

**EventID:** E8060651 (hex)

**Message:** 1618

"Exceeded limit on number of windows."

**Severity:** Error

**Further explanation:** There is a limit of 4 windows per screen.

**EventID:** E8060652 (hex)

**Message:** 1619

"Exceeded limit on number of traces/window."

**Severity:** Error

**Further explanation:** There is a limit of 4 traces per window. See the [Traces, Channels, and Windows on the PNA](https://example.com).

**Suggestions:** Create the trace in another window

**EventID:** E8060653 (hex)

**Message:** 1620

"Trace not found."
Severity: Error
Further explanation: Your program tried to communicate with a non-existing trace.
Suggestions: Query the trace ID before writing to it.
EventID: E8060654 (hex)

Message: 1621
"The operating system does not recognize this printer."
Severity: Warning
EventID: A8060655 (hex)

Message: 1622
Duplicate trace ID specified.
Severity: Error
EventID: E8060656 (hex)

Channel Errors 1792 -1878

Message: 1792
"Sweep Complete."
Severity: Informational
Further explanation: None
Suggestions: None
EventID: 68070700 (hex)

Message: 1793
"All triggerable acquisitions have completed."
Severity: Informational
Further explanation: None
EventID: 68070701 (hex)

Message: 1794
"The last trigger produced an aborted sweep."
Severity: Informational
Further explanation: None
EventID: 68070702 (hex)

Message: 1795
"The segment list must be adjusted to have at least one active segment with more than 0 points to use segment sweep."
Severity: Informational

Further explanation: You attempted to change **Sweep type** to Segment sweep, but there is either no segments defined or no sweep points in the defined segments

Suggestions: Define at least one segment with at least one measurement point. See Segment sweep for more information

EventID: 68070703 (hex)

**Message: 1796**

"MSG_SET_CHANNEL_DIRTY"

Severity: Informational

Further explanation: This informational message occurs when a channel setting has changed but the channel still has data that was taken with the previous setting. The following CLEAR message occurs when new channel data is taken.

EventID: 68070704 (hex)

**Message: 1797**

"MSG_CLEAR_CHANNEL_DIRTY"

Severity: Informational

Further explanation: The previous SET message occurs when a channel setting has changed but the channel still has data that was taken with the previous setting. This CLEAR message occurs when new channel data is taken.

EventID: 68070705 (hex)

**Message: 1798**

Sweep time has changed from Auto to Manual mode. If desired to return to Auto mode, enter sweep time value of 0.

Severity: Informational

EventID: 68070706 (hex)

**Message: 1799**

"Set Sweep Completed"

Severity: Informational

Further explanation: This event occurs when a sweep and it's associated sweep calculations finish. This is typically when all sweeps on a channel complete.

EventID: 68070707 (hex)

**Message: 1800**

"Clear Sweep Completed"

Severity: Informational

Further explanation: This event occurs immediately after the SET SWEEP COMPLETED event. These two events set and clear the "Sweep Completed" bit (bit 4) on the SCPI Device Status register.

EventID: 68070708 (hex)
Message: 1801
"All Sweeps Completed and Processed"
Severity: Informational
Further explanation: This event occurs when all of the sweeps and sweep calculations are complete for a channel.
EventID: 68070709 (hex)

Message: 1802
Low Pass: Frequency limits have been changed.
Severity: Informational
EventID: 6807070A (hex)

Message: 1803
Low Pass: Number of points have been changed.
Severity: Informational
EventID: 6807070B (hex)

Message: 1804
Low Pass: Frequency limits and number of points have been changed.
Severity: Informational
EventID: 6807070C (hex)

Message: 1805
"Channel created"
Severity: Informational
EventID: 6807070D (hex)

Message: 1806
"Channel deleted"
Severity: Informational
EventID: 6807070E (hex)

Message: 1872
"Channel not found."
Severity: Error
Further explanation: A non-existent channel is being referenced under program control.
Suggestions: Query the channel number, then refer to it by number.
EventID: E8070750 (hex)
**Message: 1873**
"The requested sweep segment was not found."
**Severity:** Error
**Further explanation:** A non-existent sweep segment is being referenced under program control.
**EventID:** E8070751 (hex)

**Message: 1874**
"The sweep segment list is empty."
**Severity:** Error
**Further explanation:** Segment Sweep cannot be specified unless there is at least one defined segment. This error will only occur under remote control.
**EventID:** E8070752 (hex)

**Message: 1875**
"The number of points in active sweep segment list segments is 0."
**Severity:** Error
**Further explanation:** Segment Sweep cannot be specified unless there is at least data point specified in a segment. This error will only occur under remote control.
**EventID:** E8070753 (hex)

**Message: 1876**
"The specified source attenuator is not valid."
**Severity:** Error
**Further explanation:** You tried to set the Attenuator property on the Channel object on a PNA that doesn't have a source attenuator.
**EventID:** E8070754 (hex)

**Message: 1877**
"Log Frequency sweep cannot be selected with the current Number of Points. Please reduce Number of Points."
**Severity:** Error
**Further explanation:** The maximum number of points that can be used for Log sweep is 401.
**EventID:** E8070755 (hex)

**Message: 1878**
"The requested Number of Points is greater than can be selected for Log Frequency sweep."
**Severity:** Error
**Further explanation:** The maximum number of points that can be used for Log sweep is 401.
**EventID:** E8070756 (hex)

**Message: 1879**
"Response frequencies exceeded instrument range so Frequency Offset has been turned off."

**Severity:** Error

**Further explanation:** This error is returned whenever the instrument detects that the stimulus sweep setup and Frequency Offset settings result in computed response frequencies that exceed instrument limits. When this occurs, the instrument automatically turns off Frequency Offset to avoid the out-of-range conditions.

**Suggestions:** When this condition has occurred, change settings for either the stimulus frequencies or Frequency Offset so that the Response frequencies are within instrument bounds. Once this is done, Frequency Offset can once again be turned on.

**EventID:** E8070757 (hex)

**Message:** 1880

The total number of points for all the given segments exceeds the maximum number of points supported. The segments were not changed.

**Severity:** Error

**EventID:** E8070758 (hex)

**Message:** 1881

This instance of the Channels object was not used to place the channels in Hold, so no channels were resumed.

**Severity:** Error

**EventID:** E8070759 (hex)

**Message:** 1882

The port number was outside the range of allowed port numbers.

**Severity:** Error

**EventID:** E807075A (hex)

**Message:** 1883

More ports than are present are required for this operation.

**Severity:** Error

**EventID:** E807075B (hex)

---

**General Errors**

**Message:** 2048

"The function you requested requires a capability provided by an option to the standard analyzer. That option is not currently installed."

**Severity:** Error

**Further explanation:** None

**Suggestions:** To view the options on your analyzer, click Help / About Network Analyzer. For more information see PNA Options
EventID: 68080800 (hex)

**Message: 2049**
"The feature you requested is not available on the current instrument."

**Severity:** Error

**Further explanation:** None

EventID: 68080801 (hex)

**Message: 2050**
"The feature you requested is incompatible with the current instrument state."

**Severity:** Error

**Further explanation:** None

**Suggestions:** None

EventID: 68080802 (hex)

**Message: 2051**
"File<x> has been saved."

**Severity:** Informational

**Further explanation:** None

EventID: 68080803 (hex)

**Message: 2052**
"Attempt to save <x> failed."

**Severity:** Error

**Further explanation:** None

**Suggestions:** If using a floppy disk, ensure it is inside the drive and the disk is not full. Check the filename for special characters.

EventID: E8080804 (hex)

**Message: 2053**
"Attempt to recall file failed because <x> was not found."

**Severity:** Error

**Further explanation:** None

EventID: E8080805 (hex)

**Message: 2054**
"<x> has a bad header."

**Severity:** Error

**Further explanation:** None

**Suggestions:** Recopy the file and / or delete the file.
EventID: E8080806 (hex)

**Message: 2056**
"Request to enter hibernate state."

**Further explanation:** None

EventID: 68080808 (hex)

**Message: 2057**
"Power up from automatic hibernate state. Program received PBT_APMRESUMEAUTOMATIC Message."

**Further explanation:** None

EventID: 68080809 (hex)

**Message: 2058**
"Power up from suspend hibernate state. Program received PBT_APMRESUMESUSPEND Message."

**Further explanation:** None

EventID: 6808080A (hex)

**Message: 2059**
"Power up from suspend hibernate state. Program received PBT_APMRESUMECRITICAL Message."

**Severity:** Warning

**Further explanation:** None

EventID: A808080B (hex)

**Message: 2060**
"Power up from unknown hibernate state UI recovery called. Program received no PBT_Message within the time allotted and is attempting recovery."

**Severity:** Warning

**Further explanation:** None

EventID: A808080C (hex)

**Message: 2061**
"<x> already exists. File is being overwritten."

**Further explanation:** Used only for remote applications

EventID: 6808080D (hex)

**Message: 2062**
"File has not been saved."

**Severity:** Error

**Further explanation:** Used only for remote applications

EventID: E808080E (hex)
Message: 2063
"File <x> has been recalled."

Further explanation: Used only for remote applications
EventID: 6808080F (hex)

Message: 2064
"State version in <x> is considered obsolete by this version of this code."

Severity: Error
Further explanation: You attempted to recall a file that is no longer valid.
Suggestions: You must recreate the file manually.
EventID: E8080810 (hex)

Message: 2065
"State version in <x> is newer than the latest version supported by this code."

Severity: Error
Further explanation: You attempted to recall a file that was created by a later version of the PNA application.
Suggestions: You must recreate the file manually.
EventID: E8080811 (hex)

Message: 2066
"Error occurred while reading file <x>"

Severity: Error
Further explanation: The file may be corrupt.
Suggestions: Try to recreate the file.
EventID: E8080812 (hex)

Message: 2067
"Windows shell error: <x>"

Severity: Error
Further explanation: None
EventID: E8080813 (hex)

Message: 2068
Send message timed out returning: <x>.

Severity: Error
Further explanation: None
EventID: E8080814 (hex)

Message: 2069
"Changing GPIB mode to System Controller."

**Severity**: Informational

**Further explanation**: None

**EventID**: 68080815 (hex)

**Message**: 2070

"Changing GPIB mode to Talker Listener."

**Severity**: Informational

**Further explanation**: None

**EventID**: 68080816 (hex)

**Message**: 2071

"The Network Analyzer can not be put in GPIB System Controller mode until the GPIB status is Local. Stop any remote GPIB programs which may be using the Network analyzer, press the Macro/Local key and try again."

**Severity**: Informational

**Further explanation**: See [LCL and RMT Operation](#)

**Suggestions**: Press the Macro/Local key and try again.

**EventID**: 68080817 (hex)

**Message**: 2120

"This method can not be invoked through a late-bound COM call."

**Severity**: Error

**Further explanation**: None

**Suggestions**: Use the alternate method described in the [COM programming documentation](#)

**EventID**: E8080878 (hex)

**Message**: 2128

"The specified format is invalid."

**Severity**: Error

**Further explanation**: None

**EventID**: E8080850 (hex)

**Message**: 2129

"WINNT exception caught by Automation layer."

**Severity**: Error

**Further explanation**: None

**EventID**: E8080851 (hex)

**Message**: 2130

"Bad port specification."
Severity: Error
Further explanation: None
EventID: E8080852 (hex)

Message: 2131
"Failed to find a printer."
Severity: Error
Further explanation: None
Suggestions: See Connecting to a Printer
EventID: E8080853 (hex)

Message: 2132
"Manual trigger ignored."
Severity: Error
Further explanation: None
EventID: E8080854 (hex)

Message: 2133
"Attempt to set trigger failed."
Severity: Error
Further explanation: None
EventID: E8080855 (hex)

Message: 2134
"Macro execution failed."
Severity: Error
Further explanation: None
EventID: E8080856 (hex)

Message: 2135
"Specified macro definition is incomplete."
Severity: Error
Further explanation:
EventID: E8080857 (hex)

Message: 2137
"Block data length error."
Severity: Error
Further explanation: See Getting Data from the Analyzer
EventID: E8080859 (hex)

**Message: 2139**
"Requested data not found."

**Severity:** Error

**Further explanation:** None

EventID: E808085B (hex)

**Message: 2142**
"The parameter supplied was out of range, so was limited to a value in range before being applied to the instrument."

**Severity:** Success

**Further explanation:** None

**Suggestions:** View range limits before sending programming commands.

EventID: 2808085E (hex)

**Message: 2143**
The parameter supplied was out of range, so was limited to a value in range before being applied to the instrument.

**Severity:** Error

EventID: E808085F (hex)

**Message: 2144**
"Request failed. The required license was not found."

**Severity:** Error

**Further explanation:** None

EventID: E8080860 (hex)

**Message: 2145**
"A remote call to the front panel has returned hresult <x>"

**Severity:** Error

**Further explanation:** This may indicate a problem with the front panel

**Suggestions:** Contact Technical support

EventID: E8080861 (hex)

**Message: 2146**
The recall operation failed.

**Severity:** Error

**Further explanation:**

EventID: E8080862 (hex)
**Message: 2147**
Attempt to save file failed.
**Severity:** Error
**Further explanation:**
**EventID:** E8080863 (hex)

**Message: 2148**
Recall attempt failed because file was not found.
**Severity:** Error
**Further explanation:**
**EventID:** E8080864 (hex)

**Message: 2149**
Recall file has a bad header.
**Severity:** Error
**Further explanation:**
**EventID:** E8080865 (hex)

**Message: 2150**
Recall file version is obsolete and no longer compatible with this instrument.
**Severity:** Error
**Further explanation:**
**EventID:** E8080866 (hex)

**Message: 2151**
The recall file contains an istate version newer than this instrument. A remote call to the front panel has returned hresult %1
**Severity:** Error
**Further explanation:**
**EventID:** E8080867 (hex)

**Message 2152**
"Front Panel <x>
**Severity:** Error
**Further explanation:** None
**EventID:** E8080868 (hex)

**Message 2153**
"Front Panel message"
**Severity:** Informational
Further explanation: None
EventID: 68080869 (hex)

**Message 2154**
"Power Service <x>"

Severity: Error

Further explanation: There is more than 1 instance of powerservice running. There should only be one running. This might happen after running install shield - especially when upgrading the CPU board.

Suggestions: Try rebooting. If this persists, please call Customer Support.
EventID: E808086A (hex)

**Message 2155**
"Power Service <x>"

Severity: Informational

Further explanation: None
EventID: 6808086B (hex)

**Message 2156**
"The Agilent Technologies GPIB driver can not be loaded or unloaded."

Severity: Error

Further explanation: None

Suggestions: If the problem persists, from the PNA desktop, right-click on My Computer. Click Properties, Click Hardware Tab. Click Device Manager Button. Expand GPIB Devices. Right-click and click Uninstall all GPIB interfaces devices. Reboot the PNA.
EventID: E808086C (hex)

**Message 2157**
"The National Instruments GPIB driver can not be loaded or unloaded."

Severity: Error

Further explanation: None

Suggestions: If the problem persists, from the PNA desktop, right-click on My Computer. Click Properties, Click Hardware Tab. Click Device Manager Button. Expand GPIB Devices. Right-click and click Uninstall all GPIB interfaces devices. Reboot the PNA.
EventID: E808086D (hex)

**Message 2158**
"The Agilent GPIB driver is loaded but it can not start its parser."

Severity: Error

Further explanation: None
EventID: E808086E (hex)
**Message: 2159**
The front panel is in remote mode.

**Severity:** Warning

**EventID:** A808086F (hex)

**Message: 2160**
The Registry Key specified could not be found.

**Severity:** Error

**EventID:** E8080870 (hex)

**Message: 2161**
An overcurrent condition has been detected on a probe plugged into the front panel.

**Severity:** Warning

**EventID:** A8080871 (hex)

**Message: 2162**
The operation timed out.

**Severity:** Error

**EventID:** E8080872 (hex)

**Message 2163**
"The Network Analyzer executed a preset."

**Severity:** Informational

**Further explanation:** None

**EventID:** 68080873 (hex)

**Message 2164**
"Access to file denied."

**Severity:** Error

**Further explanation:** This means that the system can not open an output file for writing. Most likely because the file is write protected.

**Suggestions:** Pick another file name or file directory, check floppy disk hard disk write access.

**EventID:** E8080874 (hex)

**Message 2165**
"File type is structured storage."

**Severity:** Informational

**Further explanation:** None

**EventID:** 68080875 (hex)
**Message 2166**
"The trigger operation failed."
**Severity:** Error
**Further explanation:** None
**EventID:** E8080876 (hex)

**Message 2167**
"Argument out of range error."
**Severity:** Error
**Further explanation:** None
**Suggestions:** None
**EventID:** E8080877 (hex)

**Message: 2169**
The given COM object is not a custom application
**Severity:** Error
**EventID:** E8080879 (hex)

**Message: 2170**
The eventID supplied was not recognized as a valid PNA eventID
**Severity:** Error
**EventID:** E808087A (hex)

**Message: 2171**
The operation was canceled.
**Severity:** Error
**EventID:** E808087B (hex)

**Message: 2172**
High security level cannot be disabled directly. Only an instrument preset or recall of lower security instrument state will reset this security level.
**Severity:** Error
**EventID:** E808087C (hex)

**Message: 2173**
Local lockout mode is on. The PNA application will not accept input from front panel, keyboard or mouse until this mode is turned off from a remote interface.
**Severity:** Error
**EventID:** E808087D (hex)
Message: 2174
The SnP request is not valid for the selected measurement.
Severity: Error
EventID: E808087E (hex)

Message: 2175
Preset is not supported while this dialog or wizard is open. Close the dialog or wizard and then try again.
Severity: Error
EventID: E808087F (hex)

Message: 2176
The function you requested requires a capability provided by an option to the standard analyzer. That option is not currently installed.
Severity: Error
EventID: E8080880 (hex)

Message: 2177
Catastrophic error. Crash dump recorded at <n>
Severity: Error
EventID: E8080881 (hex)

Message: 2178
<n>
Severity: Error
EventID: E8080882 (hex)

Message: 2179
Failed to open gen.lic.
Severity: Error
EventID: E8080883 (hex)

Last modified:

  Nov. 29, 2006  Updated bookmarks
About Error Messages

PNA errors and Operating System errors are displayed and logged in an error file. You can choose how to display PNA errors, or choose to not display PNA errors at all.

Error Preferences

View Error Log
List of PNA Errors
SCPI Errors

Other Support topics

Error Preferences

By default, error messages appear on the PNA screen for a brief period. You can choose to have them stay on the screen until you click an OK button, or have them not appear at all. When they stay on the screen, a Help button is available to provide further assistance.

How to select Error Preferences

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>For PNA-L and E836x models</td>
<td>No programming commands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Help</td>
</tr>
<tr>
<td></td>
<td>2. then Error Messages</td>
</tr>
<tr>
<td></td>
<td>3. then Error Preferences</td>
</tr>
<tr>
<td>For PNA-X and 'C' models</td>
<td>No programming commands</td>
</tr>
<tr>
<td>1. Press SYSTEM</td>
<td>1. Click Help</td>
</tr>
<tr>
<td>2. then [Help]</td>
<td>2. then Error Messages</td>
</tr>
<tr>
<td>3. then [Error Messages]</td>
<td>3. then Error Preferences</td>
</tr>
<tr>
<td>4. then [Error Preferences]</td>
<td></td>
</tr>
</tbody>
</table>
Error Preferences dialog box help

**Enable Messages**  Check to display all PNA error messages as they occur. Clear to suppress the display of PNA error messages. You can still view them in the error log.

**Calibration Error Message Windows**

- **Timed Popups** Displays error messages on the screen for a duration of time proportional to the length of the message. You can then view the message in the error log and get further assistance.
- **Confirmation Dialog boxes** Displays error messages in a standard dialog box. You then choose OK or Cancel to close the dialog box, or press Help to get further information on the error message.

View Error Log

The PNA Error Log is a list of all events that have occurred. (Events are used in programming the PNA using COM.) PNA errors is a subset of PNA events. Only events with severity codes of ERROR are displayed on the PNA screen as they occur. From the error log, you can access further help with an error by selecting the error and clicking Help.

### How to view the Error Log

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For PNA-L and E836x models</strong></td>
<td><strong>No programming commands</strong></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Help</td>
</tr>
<tr>
<td>2. then Error Messages</td>
<td>2. then View Error Log</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td><strong>No programming commands</strong></td>
</tr>
<tr>
<td>1. Press SYSTEM</td>
<td>1. Click Utility</td>
</tr>
<tr>
<td>2. then [Help]</td>
<td>2. then System</td>
</tr>
</tbody>
</table>
3. then [Error]

4. then Error
**Error Log** dialog box help

**Network analyzer errors only**  Select to view only PNA errors. Clear to view all errors that occur on all applications of the computer.

**Description**  Error message that appears on the PNA screen.

**A**  - Event ID  Error message number

**B**  - Date the Error occurred

**C**  - Time the Error occurred

**D**  - Severity Code  All events have one of the following severity codes:

- SUCcess - the operation completed successfully
- INFormational - events that occur without impact on the measurement integrity
- WARning - events that occur with potential impact on measurement integrity
- ERRor - events that occur with serious impact on measurement integrity

**E**  - Application in which the error occurred.

**OK**  Closes the Dialog box

**Help**  Provides further information on the selected Error message

To clear the Error Log:

1. From the **View** menu click **Minimize Application**
2. On the desktop, select **Start, Settings, Control Panel**
3. On the Control Panel, click **Administrative Tools**
4. On the Administrative Tools window, click **Event Viewer**
5. On the Event Viewer window, right-click **Application Log**
6. Select **Clear all Events**
7. If you want to save a file with the contents of the Event Log, click **Yes**. Otherwise, click **No**

To restore the PNA application, click on the PNA Analyzer taskbar button at the bottom of the screen

Last Modified:

1-Jan-2007   MX Added UI
Analyzer Accessories

- Coax Mechanical Calibration Kits
- Waveguide Mechanical Calibration Kits
- Electronic Calibration (ECal)
- Mechanical Verification Kits
- Adapter and Accessory Kits
- Test Port Cables
- USB Peripherals
- Connector Care and Cleaning Supplies
- ESD Protection

Other Support topics

For product and order information:

- Call 1-800-452-4844 (8am-8pm EST)
- Visit www.agilent.com/find/accessories
  Use the search function to locate information about a particular accessory or view the entire RF and Microwave Test Accessories Catalog.

Accessories are available in these connector types:

- 50 ohm Type-N
- 75 ohm Type-N
- 3.5 mm
- 7 mm (APC-7)
- 7-16
- 2.92 mm
- 2.4 mm
- 1.85 mm
- 1 mm
Test port cables and a calibration kit are necessary for a complete measurement system. A verification kit is used to verify corrected system performance. See the connector type for each PNA model

### Coax Mechanical Calibration Kits

<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Upper Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>85032B</td>
<td>Type-N (50 Ohm)</td>
<td>6 GHz</td>
</tr>
<tr>
<td>85032F</td>
<td>Type-N (50 Ohm)</td>
<td>9 GHz</td>
</tr>
<tr>
<td>85054B</td>
<td>Type-N (50 Ohm)</td>
<td>18 GHz</td>
</tr>
<tr>
<td>85036E</td>
<td>Type-N (75 Ohm)</td>
<td>3 GHz</td>
</tr>
<tr>
<td>85050B</td>
<td>7 mm</td>
<td>18 GHz</td>
</tr>
<tr>
<td>85033D</td>
<td>3.5 mm</td>
<td>6 GHz</td>
</tr>
<tr>
<td>85038A</td>
<td>7-16</td>
<td>7.5 GHz</td>
</tr>
<tr>
<td>85033E</td>
<td>3.5 mm</td>
<td>9 GHz</td>
</tr>
<tr>
<td>85052B</td>
<td>3.5 mm</td>
<td>26.5 GHz</td>
</tr>
<tr>
<td>85052C</td>
<td>3.5 mm TRL</td>
<td>26.5 GHz</td>
</tr>
<tr>
<td>85056K</td>
<td>2.92 mm</td>
<td>50 GHz</td>
</tr>
<tr>
<td>85056A</td>
<td>2.4 mm</td>
<td>50 GHz</td>
</tr>
<tr>
<td>85058B/E (data-based)</td>
<td>1.85 mm</td>
<td>67 GHz</td>
</tr>
<tr>
<td>85059A (data-based)</td>
<td>1.00 mm</td>
<td>110 GHz</td>
</tr>
</tbody>
</table>

### Waveguide Mechanical Calibration Kits
<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>X11644A</td>
<td>WR-90</td>
<td>8.2-12.4 GHz</td>
</tr>
<tr>
<td>P11644A</td>
<td>WR-62</td>
<td>12.4-18 GHz</td>
</tr>
<tr>
<td>K11644A</td>
<td>WR-42</td>
<td>18-26.5 GHz</td>
</tr>
<tr>
<td>R11644A</td>
<td>WR-28</td>
<td>26.5-40 GHz</td>
</tr>
<tr>
<td>Q11644A</td>
<td>WR-22</td>
<td>33-50 GHz</td>
</tr>
<tr>
<td>U11644A</td>
<td>WR-19</td>
<td>40-60 GHz</td>
</tr>
<tr>
<td>V11644A</td>
<td>WR-15</td>
<td>50-75 GHz</td>
</tr>
</tbody>
</table>

**Electronic Calibration (ECal)**

<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RF Two-Port</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>85091C</td>
<td>7 mm (APC-7)</td>
<td>300 kHz-9 GHz</td>
</tr>
<tr>
<td>85092C</td>
<td>Type-N (50 ohm)</td>
<td>300 kHz-9 GHz</td>
</tr>
<tr>
<td>Port B available with 3.5 mm or 7-16A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>85093C</td>
<td>3.5 mm</td>
<td>300 kHz-9 GHz</td>
</tr>
<tr>
<td>Port B available with Type-N (50 ohm) or 7-16A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>85096C</td>
<td>Type-N (75 ohm)</td>
<td>300 kHz-3 GHz</td>
</tr>
<tr>
<td>85098C</td>
<td>7-16A</td>
<td>300 kHz-7.5 GHz</td>
</tr>
<tr>
<td>Port B available with Type-N (50 ohm) or 3.5 mm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>85099C</td>
<td>Type-F</td>
<td>300 kHz-3 GHz</td>
</tr>
</tbody>
</table>

**RF Four-Port**

<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>N4431B Option 010</td>
<td>3.5mm (f) (four-port)</td>
<td>300 kHz-13.5 GHz</td>
</tr>
</tbody>
</table>
N4431B Option 020  |  Type-N (f) (four-port)  |  300 kHz-13.5 GHz
N4432A Option 020  |  Type-N (f) (four-port)  |  300 kHz-18 GHz (available Feb. 2006)
N4432A Option 030  |  APC 7 (four-port)  |  300 kHz-18 GHz (available Feb. 2006)
N4433A Option 010  |  3.5mm (f) (four-port)  |  300 kHz-20 GHz (available Feb. 2006)

**Microwave Two-Port**

<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>N4690B</td>
<td>Type-N (50 ohm)</td>
<td>300 kHz-18 GHz</td>
</tr>
<tr>
<td>N4691B</td>
<td>3.5 mm</td>
<td>300 kHz-26.5 GHz</td>
</tr>
<tr>
<td>N4692A</td>
<td>2.92 mm</td>
<td>10 MHz-40 GHz</td>
</tr>
<tr>
<td>N4693A</td>
<td>2.4 mm</td>
<td>10 MHz-50 GHz</td>
</tr>
<tr>
<td>N4694A</td>
<td>1.85 mm</td>
<td>10 MHz-67 GHz</td>
</tr>
<tr>
<td>N4696BA</td>
<td>7 mm</td>
<td>300 kHz-18 GHz</td>
</tr>
</tbody>
</table>

a Limits ECal module high frequency to 7.5 GHz.

**Verification Kits**

<table>
<thead>
<tr>
<th>Model</th>
<th>Connector Type</th>
<th>Frequency Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>85055A</td>
<td>Type-N (50 Ohm)</td>
<td>300 kHz-9 GHz</td>
</tr>
<tr>
<td>85053B</td>
<td>3.5 mm</td>
<td>300 kHz-26.5 GHz</td>
</tr>
<tr>
<td>85057B</td>
<td>2.4 mm</td>
<td>.045-50 GHz</td>
</tr>
<tr>
<td>R11645A</td>
<td>WR-28</td>
<td>26.5-40 GHz</td>
</tr>
<tr>
<td>Q11645A</td>
<td>WR-22</td>
<td>33-50 GHz</td>
</tr>
</tbody>
</table>

**Adapters and Accessory Kits**
<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>11878A</td>
<td>Type-N to 3.5 mm Adapter Kit</td>
</tr>
<tr>
<td>11525A</td>
<td>Type-N (m) to 7 mm (APC-7)</td>
</tr>
<tr>
<td>11853A</td>
<td>Type-N Accessory Kit</td>
</tr>
<tr>
<td>11900B</td>
<td>2.4 mm (f) to 2.4 mm (f)</td>
</tr>
<tr>
<td>11900C</td>
<td>2.4 mm (f) to 2.4 mm (m)</td>
</tr>
<tr>
<td>85130G</td>
<td>Test Port Adapter Set, 2.4 mm (f) to 2.4 mm (m,f)</td>
</tr>
<tr>
<td>11901B</td>
<td>2.4 mm (f) to 3.5 mm (f)</td>
</tr>
<tr>
<td>11901D</td>
<td>2.4 mm (f) to 3.5 mm (m)</td>
</tr>
<tr>
<td>85130F</td>
<td>Test Port Adapter Set, 2.4 mm (f) to 3.5 mm (m,f)</td>
</tr>
<tr>
<td>11902B</td>
<td>2.4 mm (f) to 7 mm (APC-7)</td>
</tr>
<tr>
<td>11920A</td>
<td>1 mm (m) to 1 mm (m)</td>
</tr>
<tr>
<td>11920B</td>
<td>1 mm (f) to 1 mm (f)</td>
</tr>
<tr>
<td>11920C</td>
<td>1 mm (m) to 1 mm (f)</td>
</tr>
<tr>
<td>11921A</td>
<td>1 mm (m) to 1.85 mm (m)</td>
</tr>
<tr>
<td>11921B</td>
<td>1 mm (f) to 1.85 mm (f)</td>
</tr>
<tr>
<td>11921C</td>
<td>1 mm (m) to 1.85 mm (f)</td>
</tr>
<tr>
<td>11921D</td>
<td>1 mm (f) to 1.85 mm (m)</td>
</tr>
<tr>
<td>11922A</td>
<td>1 mm (m) to 2.4 mm (m)</td>
</tr>
<tr>
<td>11922B</td>
<td>1 mm (f) to 2.4 mm (f)</td>
</tr>
<tr>
<td>11922C</td>
<td>1 mm (m) to 2.4 mm (f)</td>
</tr>
<tr>
<td>11922D</td>
<td>1 mm (f) to 2.4 mm (m)</td>
</tr>
</tbody>
</table>
## Test Port Cables

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N4697E</td>
<td>1.85 mm (f) to 1.85 mm (rugged f) flexible (single)</td>
</tr>
<tr>
<td>N4697F</td>
<td>1.85 mm (rugged f, f) to 1.85 mm (rugged m, rugged f) flexible (set)</td>
</tr>
<tr>
<td>N6315A</td>
<td>Type-N (m) to Type-N (f), 16 in. (single)</td>
</tr>
<tr>
<td>N6314A</td>
<td>Type-N (m) to Type-N (m), 24 in. (single)</td>
</tr>
<tr>
<td>85133D</td>
<td>2.4 mm (f) to 2.4 mm (m,f) semi-rigid (set)</td>
</tr>
<tr>
<td>85133F</td>
<td>2.4 mm (f) to 2.4 mm (m,f) flexible (set)</td>
</tr>
<tr>
<td>85134D</td>
<td>2.4 mm (f) to 3.5 mm (m,f) semi-rigid (set)</td>
</tr>
<tr>
<td>85134F</td>
<td>2.4 mm (f) to 3.5 mm (m,f) flexible (set)</td>
</tr>
</tbody>
</table>

## USB Peripherals

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N4688A</td>
<td><strong>CD RW drive</strong> - with USB cable.</td>
</tr>
<tr>
<td>N4689A</td>
<td><strong>USB 4-port hub</strong> - for connecting additional USB peripherals.</td>
</tr>
<tr>
<td>82357A</td>
<td><strong>USB/GPIB Interface</strong> - for controlling GPIB devices through USB. Learn more about <a href="#">using the 82357A with the PNA</a></td>
</tr>
</tbody>
</table>

## Connector Care and Cleaning Supplies
<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9301-1243</td>
<td>Lint-Free Swabs (small), 100 ct.</td>
</tr>
<tr>
<td>8500-5344</td>
<td>IPA 99.5% alcohol, 30 ml. bottle</td>
</tr>
<tr>
<td>8500-6659</td>
<td>Compressed Air, 235 ml. can</td>
</tr>
<tr>
<td>5021-7607</td>
<td>Type-N Contact Removal Tool</td>
</tr>
<tr>
<td>1401-0225</td>
<td>Standard End-Cap, Type-N (m)</td>
</tr>
<tr>
<td>1401-0248</td>
<td>ESD Safe End-Cap, Type-N (m)</td>
</tr>
<tr>
<td>1401-0225</td>
<td>Standard End-Cap, Type-N (f)</td>
</tr>
<tr>
<td>1401-0247</td>
<td>ESD Safe End-Cap, Type-N (f)</td>
</tr>
</tbody>
</table>

**ESD Supplies**

<table>
<thead>
<tr>
<th>Part Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9300-1367</td>
<td>Adjustable antistatic wrist strap</td>
</tr>
<tr>
<td>9300-0980</td>
<td>Antistatic wrist strap grounding cord (5 foot)</td>
</tr>
<tr>
<td>9300-0797</td>
<td>Static control table mat (2 foot x 4 foot) with earth ground wire</td>
</tr>
<tr>
<td>9300-1126</td>
<td>ESD heel strap</td>
</tr>
</tbody>
</table>
82357A USB / GPIB Interface

The Agilent 82357A is an adapter that creates a GPIB Interface from one of your unused PNA USB ports.

- **Applications**
- **Installing**
- **Configuring**
- **Connecting**
- **Communicating with other Equipment**

**Applications**
The 82357A can be used for the following PNA applications:

- **Frequency Converter Application** - The 82357A is included with the Frequency Converter Application (option 083). External sources MUST be connected to this Interface if controlling the PNA using an external PC. See connecting diagram below. To learn more, see Configure an external LO source.

**Note**: If the PNA is **hibernated** during an FCA measurement involving an external source under FCA control, and then the PNA is restarted, a VISA error message will appear stating "VI_ERROR_INV_OBJECT." To correct this problem, the 82357 USB/GPIB interface must be reinitialized after hibernation. This is done by clicking on the Accept button in the interface initialization dialog box. The green READY light on the interface will illuminate.

- **PNA Controller** - The 82357A can be used by the PNA to control other GPIB devices. This frees the default GPIB interface to perform other GPIB operations, such as control the PNA from an external PC.

- **Source Power Cal** - The 82357A can be used to run a source power calibration.

**Installing the 82357A USB/GPIB Interface**

1. Download and install firmware PNA revision 3.0 or greater. To check the revision of your PNA firmware, click Help then About Network Analyzer.

2. Upgrade to the latest Agilent IO libraries from the CDROM that was shipped with the 82357A. If not available, download them from www.Agilent.com (search for 82357A)

**Configure the 82357A USB/GPIB Interface**

When the 82357A is connected to the PNA USB, the following dialog box appears:
Normally, you do NOT need to edit these settings. The 82357A USB/GPIB Interface is configured automatically as the next unused VISA interface. This is usually **GPIB2** unless you have already configured it for another purpose.

If the VISA Interface Name appears as GPIB0 or GPIB1, these Interfaces must be returned to their default settings for the 82357A to work properly with the PNA. See Configure for VISA / SICL to learn how.

### Connecting the 82357A USB/GPIB Interface

The following diagram illustrates how to connect GPIB test equipment using the USB/GPIB Interface.

- Plug the USB/GPIB Interface into any unused PNA USB port.
- The default GPIB Interface and USB/GPIB Interface should never be connected together.

![Diagram](image)

### Communicating with Equipment Connected to the USB/GPIB Interface

- The Frequency Converter Application will automatically find and communicate with test equipment that is connected to the USB/GPIB Interface.
- Source power calibration will automatically find and communicate with the power meter that is connected to the USB/GPIB Interface.
- To control other devices through your own program using the 82357A, you must include the new GPIB Interface number when addressing the devices.
Firmware Upgrade

PNA firmware upgrades are available to you at no cost in a self-extracting Install Shield file. The upgrade includes the PNA application, Online help, and Service Utilities. Note: The file is at least 50 MB.

The following options are available for you to upgrade your PNA application:

- **Auto-Check** and **AgileUpdate**: If your PNA is connected to the Internet, these utilities will automatically check for, download, and install, the new firmware and associated files when the PNA application is started. You will be prompted before this occurs.

- **Website Access**: If your PNA is NOT connected to the Internet, but you have a PC that is, you can download the PNA firmware and associated files to a storage medium.

To manually check the version of firmware on the PNA, click Help, then About Network Analyzer.

**Note: After a firmware upgrade...**

- Custom Cal Kits must be imported. Learn more
- If a different desktop icon named "Network Analyzer" exists, the shortcut to the PNA application will assume the same icon. Right-click on the desktop, then click **Refresh**.

### Other Support Topics

#### Auto-Check

With Internet access to your PNA, Auto-Check automatically and regularly checks the Internet for new PNA firmware revisions. If a new revision is found, a notification message prompts you to run the **AgileUpdate** utility, which then performs the actual download.

Without Internet access to your PNA, Auto-Check provides a reminder prompt at the selected intervals.

Auto-Check is run only when the PNA application is started. Once the PNA application is running, it will not check for updates again until it is restarted.

When Auto-Check runs, it checks the following conditions:

- Is there an active connection to the Internet?
- Is the Auto-Check utility enabled?
- Is it time to check for new firmware?
- Does new firmware exist?

If all of these conditions are true, Auto-Check shows the following dialog box.

If all of these conditions are NOT true, or to change these settings at any time, click **System, Service**, then **AgileUpdate**. From within AgileUpdate, click **AutoCheck**. These preferences are stored in the PNA registry. Future
firmware upgrades will not change these settings.

**PNA Auto-Check** dialog box help

**Enable**  When the PNA application is started, Auto-Check will search the PNA website for firmware updates at the selected time interval.

**Disable**  When the PNA application is started, Auto-Check will NOT search the PNA website for firmware updates.

**Time Interval**  Select the time interval Auto-Check is to search for firmware updates.

**Accept**  Starts update process.

**Ignore**  No further action is taken until the selected time interval has elapsed.

**Remind Me Later:**  This window is displayed again after 1-20 days depending upon the time interval selected.

**AgileUpdate**

**Note:** You must have administrative privileges on the analyzer to run this utility. See [Set Up Analyzer Users](#).

**How to start AgileUpdate**

Connect the PNA to the Internet. A LAN connection is recommended because a firmware download can take many hours using a modem.

**Using front-panel HARDKEY [softkey] buttons**

<table>
<thead>
<tr>
<th>For <strong>PNA-L</strong> and <strong>E836x</strong> models</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>No programming commands</td>
</tr>
<tr>
<td>2. then <strong>Service</strong></td>
<td></td>
</tr>
<tr>
<td>3. then <strong>AgileUpdate</strong></td>
<td></td>
</tr>
</tbody>
</table>
### For PNA-X and 'C' models

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Press System</td>
</tr>
<tr>
<td>2.</td>
<td>then [Service]</td>
</tr>
<tr>
<td>3.</td>
<td>then [AgileUpdate]</td>
</tr>
</tbody>
</table>

No programming commands

### AgileUpdate

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Click Utility</td>
</tr>
<tr>
<td>2.</td>
<td>then System</td>
</tr>
<tr>
<td>3.</td>
<td>then Service</td>
</tr>
<tr>
<td>4.</td>
<td>then AgileUpdate</td>
</tr>
</tbody>
</table>

1. Click **Check for Updates**.
2. If updates exist, click **Download & Install**.

![AgileUpdate screenshot](image_url)
**AgileUpdate** dialog box help

**Note:** Your privacy is important to Agilent. AgileUpdate does NOT send ANY information from the PNA to the server. It only downloads from the server to the PNA.

**Restart**  Click to restart from the beginning.

**Configure**  Click to launch the [Configure dialog box](#).

**Clean-up**  Click to delete all but the two most recent install shield packages from the PNA hard drive.

**Firmware History**  Available after clicking **Check for Updates**.

**Auto-Check**  Launches the [Auto-Check](#) dialog box.

**Item / Application**  Lists the items available for download at the firmware website.

- Click on items with ![i](#) to read more information about the download.
- Items in **RED** should be downloaded and installed individually.
- Multi-language help includes all help files except English.

**Note:** The firmware includes the help file. Therefore, only the firmware checkbox will be selected if a new version for both the firmware and the help file are available.

**Select Source**

**Default Website**  The Agilent site that contains upgrade FW.

**Other Specified URL**  Click if you were instructed to get firmware from a different website.

**Install from File**  Click if you have already downloaded the InstallShield package and want AgileUpdate to install it for you.

**Special Access Code...**  Type in the code if you were given one from Agilent Technical Support. Otherwise, leave blank.

**Make Latest Firmware Available...**  Select this checkbox if you want to download the latest firmware, even if it is not new.

**Check for Updates**  Click to look for firmware updates at the Agilent website. If there are newer versions, the files will be listed.

**Download and Install**  When updates are found, this selection becomes available. Some files may be pre-checked. Be sure the corresponding boxes are checked for the files you want to download. Then click to download and install the update.

**Download Only**  Click to download the files to the analyzer hard disk and install the files at a later time. At that time, click **Install from File**.
Configuration dialog box help

**Note:** If AgileUpdate will not connect, try to access ANY Internet website. Contact your local IT department if necessary.

**Proxy Setting**
- **No Proxy or Default Proxy**  Click if you use a LAN connection. AgileUpdate will automatically use the proxy specified in Internet Explorer.
- **Use specified Proxy / Port**  Click to enter the proxy name and port. The format is: proxyName:portNumber. (The proxy port number is typically 8088).

**Internet timeout**  If you are using an automatic dial-up Internet connection you may need to increase the timeout.

**Current Connection Status**  Shows the current status of the PNA connection to the Internet.

**Note:** These settings are NOT saved; they must be re-entered each time AgileUpdate is run.

---

**Agilent Website Access**

If you cannot access the Internet directly with your PNA, you can use an external PC with Internet access to download the file from the Agilent website. You can then transfer the file from your PC to your analyzer over a LAN or other means.

2. Click on the firmware to be downloaded.
3. Save the program to disk (hard drive of your PC).
4. Transfer the file from your PC to your PNA using LAN, CD, or USB Pen drive.
5. Double-click the file on the PNA.
**Warning**: You can save the upgrade file to your PC, but do not attempt to install the PNA application on your PC. It will alter system settings and can result in system crashes.
PNA Configurations and Options

Included with each PNA is a mouse, keyboard. This topic presents standard PNA models and the available options and upgrades.

- **PNA Models**
  - **PNA-X Series** (N5242A)
  - **PNA L Series** (N230A and N5230C)
  - **Microwave Models** (E836xA/B/C)
  - **RF Models** (Discontinued)
  - **mmWave Model** (N5250A)

- **Common Options and Upgrade Kits**

- **Warranty Period**

To view the options that are installed on your analyzer, click **Help** then **About Network Analyzer**

A documentation CD-ROM is no longer included in each PNA shipment (Feb.05).

---

**New PNA 'C' Models**

In Spring 2008, new PNA 'C' models were introduced that will eventually replace the PNA-L and Microwave 'A' and 'B' models. The new 'C' models have the PNA-X User Interface capabilities, including touchscreen and softkeys. In addition:

- The 'C' models have a bigger 8.4" display screen.
- The 'C' models share the same Specifications as their 'A' and 'B' model counterparts.
- Upgrades from 'A' and 'B' to 'C' models are available.

**Note:** The term 'Legacy' that is used in this help file refers to the PNA models that were introduced before the PNA-X (N5230A, E836xA/B, and all discontinued models).

---

**Other Support Topics**

---

**PNA-X Series  Model N5242A**

**Base Options**

**Note:** To see if your PNA has a Reference Receiver for each Test Port, scroll down to see the number of test ports and number of reference receivers. If they are equal, then there is a reference receiver for each test port.
Click the model or option number to see a block diagram.

Click the Option number to see the block diagram.

Click the Connector type to see the connector specifications.

<table>
<thead>
<tr>
<th>Option</th>
<th>Frequency Range</th>
<th>Requires</th>
<th>Test Ports</th>
<th>Reference Receivers</th>
<th>Sources</th>
<th>Test Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>10 MHz to 26.5 GHz</td>
<td>Opt 080</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>Includes six front-panel access loops</td>
</tr>
<tr>
<td>400</td>
<td>10 MHz to 26.5 GHz</td>
<td>None</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>Includes twelve front-panel access loops</td>
</tr>
</tbody>
</table>

Extended capabilities

<table>
<thead>
<tr>
<th>Option</th>
<th>Requires Options...</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>224</td>
<td>200, 219, and 080</td>
<td>Add internal 2nd source, combiner, and mechanical switches.</td>
</tr>
<tr>
<td>423</td>
<td>400, 419 and 080</td>
<td>Add combiner and mechanical switches.</td>
</tr>
</tbody>
</table>
| 219    | 200                 | Adds:  
  - 65 dB source attenuator (5dB steps)  
  - Bias-tees between each source and each test port  
  - 35 dB receiver attenuator (5dB steps). |
| 419    | 400 (080 recommended) | Adds:  
  - 65 dB source attenuator (5dB steps).  
  - Bias-tees between each source and each test port.  
  - 35 dB receiver attenuator (5dB steps). |
| 020    | None                | Add IF inputs on the rear panel for antenna and millimeter-wave. |
| 021    | None                | Add pulse modulator to internal Source1. Learn more. |
| 022    | None                | Add pulse modulator to internal Source2. Learn more. |
| 025    | None                | Add four internal pulse generators. Learn more. |
| 029    | Noise Figure Application | Adds hardware and firmware for high-accuracy noise figure measurements, utilizing Agilent’s unique source-correction technique. Learn more. |
### Gain Compression Application

Adds firmware for fast and accurate gain compression measurements.  
[Learn more.](#)

---

**PNA-L Series  Model N5230A and N5230C**  
See note

Click the [Option number](#) to see the block diagram.

Click the [Connector type](#) to see the connector specifications.

<table>
<thead>
<tr>
<th>Option</th>
<th>Frequency Range</th>
<th>Test Ports</th>
<th>Reference Receivers</th>
<th>Connector</th>
<th>Test Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>020</td>
<td>300 KHz* to 6 GHz</td>
<td>2</td>
<td>2</td>
<td>3.5 mm Male</td>
<td>Standard</td>
</tr>
<tr>
<td>025</td>
<td>300 KHz* to 6 GHz</td>
<td>2</td>
<td>2</td>
<td>3.5 mm Male</td>
<td>Configurable</td>
</tr>
<tr>
<td>120</td>
<td>300 KHz* to 13 GHz</td>
<td>2</td>
<td>2</td>
<td>3.5 mm Male</td>
<td>Standard</td>
</tr>
<tr>
<td>125</td>
<td>300 KHz* to 13 GHz</td>
<td>2</td>
<td>2</td>
<td>3.5 mm Male</td>
<td>Configurable</td>
</tr>
<tr>
<td>140</td>
<td>300 KHz* to 13.5 GHz</td>
<td>4</td>
<td>1</td>
<td>3.5 mm Male</td>
<td>Standard</td>
</tr>
<tr>
<td>145</td>
<td>300 KHz* to 13.5 GHz</td>
<td>4</td>
<td>1</td>
<td>3.5 mm Male</td>
<td>Configurable²</td>
</tr>
<tr>
<td>146</td>
<td>300 KHz* to 13.5 GHz</td>
<td>4</td>
<td>1</td>
<td>3.5 mm Male</td>
<td>Configurable³ + Second Source</td>
</tr>
<tr>
<td>220</td>
<td>10 MHz* to 20 GHz</td>
<td>2</td>
<td>2</td>
<td>3.5 mm Male</td>
<td>Standard</td>
</tr>
<tr>
<td>225</td>
<td>10 MHz* to 20 GHz</td>
<td>2</td>
<td>2</td>
<td>3.5 mm Male</td>
<td>Configurable</td>
</tr>
<tr>
<td>240</td>
<td>300 KHz* to 20 GHz</td>
<td>4</td>
<td>1</td>
<td>3.5 mm Male</td>
<td>Standard</td>
</tr>
<tr>
<td>245</td>
<td>300 KHz* to 20 GHz</td>
<td>4</td>
<td>1</td>
<td>3.5 mm Male</td>
<td>Configurable²</td>
</tr>
<tr>
<td>246</td>
<td>300 KHz* to 20 GHz</td>
<td>4</td>
<td>1</td>
<td>3.5 mm Male</td>
<td>Configurable³ + Second Source</td>
</tr>
<tr>
<td>420</td>
<td>10 MHz* to 40 GHz</td>
<td>2</td>
<td>2</td>
<td>2.4 mm Male</td>
<td>Standard</td>
</tr>
<tr>
<td>425</td>
<td>10 MHz* to 40 GHz</td>
<td>2</td>
<td>2</td>
<td>2.4 mm Male</td>
<td>Configurable</td>
</tr>
<tr>
<td>520</td>
<td>10 MHz* to 50 GHz</td>
<td>2</td>
<td>2</td>
<td>2.4 mm Male</td>
<td>Standard</td>
</tr>
<tr>
<td>525</td>
<td>10 MHz* to 50 GHz</td>
<td>2</td>
<td>2</td>
<td>2.4 mm Male</td>
<td>Configurable</td>
</tr>
</tbody>
</table>

* Typical specs apply
### Microwave Standard Models  (see options)

Click the **PNA model** to see the block diagram.

Click the **Connector type** to see the connector specifications.

<table>
<thead>
<tr>
<th>PNA Model</th>
<th>Frequency Range</th>
<th>Test Ports</th>
<th>Reference Receivers</th>
<th>Connector Type</th>
<th>Front Panel Jumpers</th>
</tr>
</thead>
<tbody>
<tr>
<td>E8361A</td>
<td>10 MHz* to 67 GHz (tunable to 70 GHz)**</td>
<td>2</td>
<td>2</td>
<td>1.85 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8361C</td>
<td></td>
<td></td>
<td></td>
<td>1.85 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8362A</td>
<td>45 MHz to 20 GHz</td>
<td>2</td>
<td>2</td>
<td>3.5 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8363A</td>
<td>45 MHz to 40 GHz</td>
<td>2</td>
<td>2</td>
<td>2.4 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8364A</td>
<td>45 MHz to 50 GHz</td>
<td>2</td>
<td>2</td>
<td>2.4 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8362B</td>
<td>10 MHz* to 20 GHz</td>
<td>2</td>
<td>2</td>
<td>3.5 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8362C</td>
<td></td>
<td></td>
<td></td>
<td>3.5 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8363B</td>
<td>10 MHz* to 40 GHz</td>
<td>2</td>
<td>2</td>
<td>2.4 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8363C</td>
<td></td>
<td></td>
<td></td>
<td>2.4 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8364B</td>
<td>10 MHz* to 50 GHz</td>
<td>2</td>
<td>2</td>
<td>2.4 mm Male</td>
<td>0</td>
</tr>
<tr>
<td>E8364C</td>
<td></td>
<td></td>
<td></td>
<td>2.4 mm Male</td>
<td>0</td>
</tr>
</tbody>
</table>

**Legend**

- **No longer produced**
- * Typical specs apply from 10 to 45 MHz
- ** Typical specs apply from 67 to 70 GHz

**New models**  [See note](#)

---

### RF Standard Models  (see options)

Click the **Connector type** to see the connector specifications.
### Millimeter Wave PNA

<table>
<thead>
<tr>
<th>PNA Model</th>
<th>Frequency Range</th>
<th>Ports</th>
<th>Connector Type</th>
<th>Reference Receivers</th>
<th>Front Panel Jumpers</th>
</tr>
</thead>
<tbody>
<tr>
<td>N5250A</td>
<td>10 MHz* to 110 GHz</td>
<td>2</td>
<td>1.0 mm</td>
<td>2</td>
<td>N5260A</td>
</tr>
<tr>
<td>N5250C</td>
<td>See note</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* *Typical specs* apply from 10 to 45 MHz

Test heads to 325 GHz are also available

Upgrade your existing E836xB with the following:

- **H11 option**
- N5260A Test Set
- Millimeter-Wave VNA Frequency Extension Modules from Oleson Microwave Labs Extension Modules

### Options and Upgrade Kits
The following options are installed at the time of purchase, and some are also available after the initial purchase of a PNA. To order an upgrade, contact your Agilent representative.

<table>
<thead>
<tr>
<th>Option</th>
<th>Supported Models</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>006</td>
<td>E8356A, E8801A, N3381A</td>
<td>Upgrade to 6 GHz frequency range. (No longer available) Adds 6 GHz operation. Includes installation at an Agilent service center. Instrument calibration (required after frequency upgrade) is available for an additional fee.</td>
</tr>
<tr>
<td>009</td>
<td>E8356/7A, E8801/2A, N3381/2A</td>
<td>Upgrade to 9 GHz frequency range. (No longer available) Adds 9 GHz operation. Includes installation at an Agilent service center. Instrument calibration (required after frequency upgrade) is available for an additional fee.</td>
</tr>
<tr>
<td>010</td>
<td>All</td>
<td><strong>Time-domain</strong> Adds time-domain capability to analyzer. The serial number of the analyzer must be specified when ordering this kit. Software upgrade. Learn more about Time Domain Learn how this option is enabled.</td>
</tr>
<tr>
<td>014</td>
<td>E8361A, E8362/63/64B</td>
<td><strong>Configurable test set</strong> Adds front panel access to the source output and coupler input on test ports 1 and 2. Adds front panel access directly to all receivers, including the reference receiver. Upgrade includes installation at an Agilent service center.</td>
</tr>
<tr>
<td>015</td>
<td>E8356/57/58A</td>
<td><strong>Configurable test set (No longer available)</strong> Adds front panel access to the source output and coupler input on test ports 1 and 2. Adds 35 dB step attenuators between the couplers and receivers. Upgrade includes installation at an Agilent service center.</td>
</tr>
<tr>
<td>016</td>
<td>E8361A, E8362/63/64B</td>
<td><strong>Receiver Step Attenuators</strong> Adds two step attenuators. (E8361A adds 50 dB step attenuators with 10 dB resolution; all other PNA models add 35 dB step attenuators with 5 dB resolution.) Each attenuator is inserted between a test port and its corresponding receiver. Requires option UNL.</td>
</tr>
<tr>
<td>022</td>
<td>E8361A, E8362/63/64B</td>
<td><strong>Extended Memory</strong> Adds more RAM for a total of 512MB. Learn more about PNA RAM</td>
</tr>
<tr>
<td>080</td>
<td>E8361A, E8362/63/64B, N5220A</td>
<td><strong>Frequency Offset</strong> Enables you to set the PNA source independently from where the receivers</td>
</tr>
<tr>
<td>Model</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>N5230A</td>
<td>N5242A</td>
<td></td>
</tr>
<tr>
<td>E8361A</td>
<td>E8362A/63A/64B</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reference Switch</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Adds an internal solid state switch in the R1 reference path for controlling an external reference mixer. Requires option 014 and opt 080</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See a block diagram which includes the reference switch.</td>
<td></td>
</tr>
<tr>
<td>E8361A</td>
<td>E8362B/63B/64B</td>
<td></td>
</tr>
<tr>
<td>N5242A</td>
<td>Frequency Converter Measurement Application</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Provides an intuitive user-interface for making extremely accurate conversion loss and absolute group delay measurements on mixers and converters. Exceptional amplitude and phase accuracy is achieved through two new calibration techniques: Scalar Mixer Calibration and Vector Mixer Calibration. The application also provides automatic control of all of Agilent's major signal sources. Requires option 014, 080, and 081.</td>
<td></td>
</tr>
<tr>
<td>E8361A</td>
<td>E8362B/63B/64B</td>
<td></td>
</tr>
<tr>
<td>N5242A</td>
<td>Embedded LO</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Provides the ability to measure frequency converters that have an embedded LO.</td>
<td></td>
</tr>
<tr>
<td>E8361A</td>
<td>E8362B/63B/64B</td>
<td></td>
</tr>
<tr>
<td>N5242A</td>
<td>CPU board upgrade (No longer available)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Replaces the 266 MHz CPU board with a 500 MHz CPU board.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Upgrade includes installation at an Agilent service center.</td>
<td></td>
</tr>
<tr>
<td>E8356/57/58A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E8380/02/03A</td>
<td>70 dB step attenuator (No longer available)</td>
<td></td>
</tr>
<tr>
<td>N3381/82/83A</td>
<td>Adds a single 70 dB step attenuator that is switched between the source and each output port to extend the output power to -85 dBm. <strong>Note:</strong> Two 70 dB step attenuators are standard equipment with PNA models E8365/57/58A.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Upgrade includes installation at an Agilent service center.</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Model</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>1E5</td>
<td>E8801/ 02/ 03A N3381/ 82/ 83A</td>
<td>High stability 10 MHz time base (No longer available)</td>
</tr>
<tr>
<td>550</td>
<td>E8361A E8362B/ 63B/ 64B N5230A</td>
<td>Add full 4 port capability and differential measurements - <a href="#">Learn more</a></td>
</tr>
<tr>
<td>551</td>
<td>E8361A E8362B/ 63B/ 64B N5230A</td>
<td>Add fully integrated measurements at all of the available test ports.</td>
</tr>
<tr>
<td>H08</td>
<td>E8361A E8362B/ 63B/ 64B N5242A</td>
<td>Narrowband Pulsed Measurement Application</td>
</tr>
</tbody>
</table>
| H11  | E8361A E8362B/ 63B/ 64B | IF Access | Provides rear panel access to the PNA IF paths for:  
- Extended frequency coverage to 325 GHz  
- Pulsed measurement capability  
- Advanced antenna measurements  
Requires options UNL, 014, 080, and 081.[Learn more](#) |
| UNL  | E8361A E8362/ 63/ 64A E8362/ 63/ 64B | Extended power range and bias tees | Adds two step attenuators and two bias tees. (E8361A adds 50 dB step attenuators; E8362/ 63/ 64A/B adds 60 dB step attenuators. All attenuators have a 10 dB resolution.) A step attenuator and bias-tee set is inserted between the source and test port 1 and another set between the source and test port 2. Upgrade includes installation at an Agilent service center. |

**Certification Options**
<table>
<thead>
<tr>
<th>Option</th>
<th>Supported Models</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1A7</td>
<td>E8361A, E8362B/63B/64B, N5250A</td>
<td>Complete set of measurement data which was acquired from testing your PNA to published specifications. Includes calibration label, calibration certificate, and data report. Conforms to ISO 9001.</td>
</tr>
<tr>
<td>UK6</td>
<td>E8361A, E8362B/63B/64B, N5250A</td>
<td>Complete set of measurement data which was acquired from testing your PNA to published specifications. Includes calibration label, ISO 17025 calibration certificate, data report, measurement uncertainties and guard bands on all specifications. Conforms to ISO17025 and ISO 9001.</td>
</tr>
</tbody>
</table>

**Documentation and Localization Options**

**Description**


- A documentation CD-ROM is no longer included with each PNA shipment (Feb.2005).

- To download a service guide for your PNA, or the latest version of PNA Help, visit [www.agilent.com/find/pna](http://www.agilent.com/find/pna), search for your PNA model, then click Library.

**PNA Warranty Period**

The actual warranty on your instrument depends on the date it was ordered as well as whether or not any warranty options were purchased at that time. To determine the exact warranty on your instrument, contact Agilent Technologies with the model and serial number of your instrument.

For online information about Agilent’s service and support products visit: [www.agilent.com/find/tm_services](http://www.agilent.com/find/tm_services).

Last modified:

- 25-Oct-2007 Added C models
- 5-Sep-2007 Added Embedded LO
- 9/26/06 MQ Added PNA-L 4port models
- 9/27/06 MX Added PNA-X
Option Enable Utility

The Option Enable utility allows you to perform the following activities on your PNA.

- Enable or remove software options and some hardware options.
- Recover option data if the hard drive or other data-containing assembly is replaced.
- Input or change a serial number.

The following items are discussed in this topic:

Keywords
- Running the Program
- Removing an Option
- Installing an Option
- Repairing and Recovering Option Data
- Installing or Changing the Serial Number

See Also
See PNA Configurations and Options

Keywords
To add certain options, you need a keyword that is provided by Agilent. There are two types of keywords:

- **Option Keywords** add a software option.
- **Model Keywords** may be required if you replace multiple assemblies.

Keywords are linked to the PNA Host ID, which is displayed on the Option Enable dialog box (below).

Temporary and Permanent Options
Any software option can also be installed on a temporary basis for a specified amount of time. This allows you to evaluate a specific feature or capability at no cost.

If the license key provided by Agilent has an expiration date, you must select the "temporary" option and enter the expiration date exactly as stated in the license statement. If you decide to make this option permanent, Agilent will provide a new keyword that converts the option to permanent status.

For either permanent or temporary software options, a provided keyword must be entered.

Running the Program
On the PNA, click **System**, point to **Service**, then click **Option Enable**.

1. To enable or remove an option, select it from the drop-down list of available options. If the desired option is not listed, select the last choice in the list, labeled **Enter Unlisted Option**.

2. Enter the 3-character option name and click **Enter**

If a software option was chosen, the following occurs.

- The **Remove** button will be enabled.
- The keyword entry area becomes visible.
- The permanent/temporary selection is enabled.

If a hardware option is selected, the following occurs.

- With the hardware option already installed, the **Remove** button is enabled.
- With the hardware option not installed, the **Enable** button is enabled.

**Removing an Option**

1. To remove an option, click **Remove**.

2. After the option is removed, restart the network analyzer application for the changes to take effect.

**Note:** Removal of a licensed option (such as Option 010, Time Domain) will permanently remove the license
keyword. If this option may be needed in the future, then record the license keyword before removing the option. Do this by copying the file "gen.lic" to another location (such as a floppy disk), or print it using notepad. The file, located at "C:\Program Files\Agilent\Network Analyzer" contains all the information needed to recreate the license.

**Installing an Option**

1. If the keyword entry area is visible, enter a keyword. (The keyword is not case sensitive.)
2. Click **Enable**.
3. After the option is installed, restart the network analyzer application for the changes to take effect.

**Note:** If a desired option is not visible, it may be because a prerequisite option has not yet been installed. For example, Option 083 will not be visible if Option 080 is not already present. See PNA options.

**Repairing and Recovering Option Data**

Use this part of the Option Enable Utility in the following situations:

- If the hard drive is replaced
- If the frequency reference assembly is replaced

This routine rebuilds the option information contained on the hard drive and frequency reference assembly (primary and backup).

1. Select **Repair** from the **Option Enable** menu bar.

**Note:** If you are unsure if this routine needs to be done, run it; no harm will result.

2. The model and serial number are displayed, along with four check boxes.
3. Select the boxes that apply.
4. Click **Begin Repair**. The routine checks all data files and performs any needed repairs. You may be asked to verify certain information and processes.
5. If the routine finds that the model number is incorrect or invalid, you will be asked to select the correct model number.
   - Along with this model number, a model keyword will be required. If this is not labeled on the analyzer, or is not otherwise known, contact Agilent
   - After you have entered the requested data, click **Change Model**. This process takes about 30 seconds.
6. When done, click **Exit Repair**.
7. If you do not need to install any other options, click **Exit**.
Installing or Changing the Serial Number

It may be necessary to install or change a serial number if certain assemblies are replaced.

1. To change the serial number, select Change Serial from the Option Enable menu bar. The current serial number will be displayed. If no serial number has previously been entered, the word "NONE" will be displayed.

2. Type the new serial number into the space provided, and click Change Serial. (The serial number is not case sensitive.)

Note: Use extreme care when entering the serial number; only one entry chance is allowed!

3. To change an incorrect serial number, a clear-code password is required. Contact Agilent to obtain this clear code and have the existing serial number available. Enter the clear code in the space provided, along with the new serial number, then click Change Serial.

Last Modified:

20-Sep-2007    Added Install note
**Instrument Calibration**

An instrument calibration is a process where the analyzer performance is measured to ensure that the analyzer operates within specifications. If any performance parameter does not conform to the published specifications, adjustments are made to bring the performance into conformance.

**Why Should I Get an Instrument Calibrated?**

Over time, the active components in the analyzer age and the performance may degrade or drift. To ensure that the analyzer is performing to the published specifications, you must have an instrument [calibration](#) performed periodically.

**How Often Should I Get an Instrument Calibrated?**

The instrument specifications are set to consider the performance drift that may occur over a 12 month period. Therefore, getting the instrument calibrated at 12 month intervals ensures that the analyzer maintains performance within the operating specifications. If you need the analyzer to maintain more consistent operation parameters, you may want to have the instrument calibrated more often than the suggested 12-month interval.

**How Do I Get an Instrument Calibrated?**

To get the instrument calibrated, send it to one of the Agilent Technologies service centers. See [Technical Support](#).

The PNA must be fully functional when it is sent to the service center, or they will charge for their repair services. If the PNA is being used in a secure environment where the hard drive can not be sent with the PNA, a second hard drive must be purchased and configured for use with the PNA in an "unclassified" environment before the PNA is sent to the service center.

To perform the instrument calibration yourself, you must have the following required items:

- Instrument Calibration Test Equipment
- Performance Test Software

**What Are My Choices of Instrument Calibration?**

The following types of instrument calibration are available from Agilent Technologies:

- **Standard**
  Includes a certificate of calibration, stating the instrument has been calibrated and is operating within the published specifications.

- **Option UK6**
  Includes the test data from the calibration and the standard certificate, stating the instrument has been calibrated and is operating within the published specifications.

- **Option M40 (Special)**
  Includes the test data from the calibration and a certificate, stating the instrument has been calibrated using a process in compliance with ANSI Z540 and is operating within the published specifications.
Other Resources

The following network analysis resources are also available.

Document Resources

Application Notes
You can also access application notes at this URL:
http://www.agilent.com/find/PNA

Third-Party Resources
For information about test fixtures and part handlers, contact:
   Inter-Continental Microwave
   www.icmicrowave.com

For information about probing equipment and accessories, contact:
   Cascade Microtech, Inc.
   www.cascademicrotech.com
SCPI Errors

Standard SCPI Errors

-100 to -200 Command Errors

-200 to -299 Execution Errors

-300 to -399 SCPI Specified Device-Specific Errors

-400 to -800 Query and System Errors

PNA specific Errors

Note: See also PNA Errors

-100 to -200 Command Errors

A command error indicates that the test set's GPIB parser has detected an IEEE 488.2 syntax error. When one of these errors is generated, the command error bit in the event status register is set.

-100 std_command Command - This event bit (Bit 5) indicates a syntax error, or a semantic error, or a GET command was entered, see IEEE 488.2, 11.5.1.1.4.

-101 std_invalidChar Invalid character - Indicates a syntactic elements contains a character which is invalid for that type.

-102 std_syntax Syntax - Indicates that an unrecognized command or data type was encountered. For example, a string was received when the device does not accept strings.

-103 std_invalidSeparator Invalid separator - The parser was expecting a separator and encountered an illegal character. For example, the semicolon was omitted after a program message unit.

-104 std_wrongParamType Data type -The parser recognized a data element different than one allowed. For example, numeric or string data was expected but block data was encountered.

-105 std_GETNotAllowed GET not allowed - Indicates a Group Execute Trigger was received within a program message. Correct the program so that the GET does not occur within the program code.
-108 std_tooManyParameters Parameter not allowed - Indicates that more parameters were received than expected for the header. For example, *ESE common command only accepts one parameter, so *ESE 0,1 is not allowed.

-109 std_tooFewParameters Missing parameter - Indicates that less parameters were received than required for the header. For example, *ESE requires one parameter, *ESE is not allowed.

-110 std_cmdHeader Command header - Indicates an error was detected in the header. This error is used when the device cannot detect the more specific errors -111 through -119.

-111 std_headerSeparator Header separator - Indicates that a character that is not a legal header separator was encountered while parsing the header.

-112 std_IDTooLong Program mnemonic too long - Indicates that the header contains more that twelve characters, see IEEE 488.2, 7.6.1.4.1.

-113 std_undefinedHeader Undefined header - Indicates the header is syntactically correct, but it is undefined for this specific device. For example, *XYZ is not defined for any device.

-114 std_suffixOutOfRange Header suffix out of range - Indicates the value of a header suffix attached to a program mnemonic makes the header invalid.

-120 std_numericData Numeric data - This error, as well as errors

-121 std_invalidCharInNumber Invalid character in number - Indicates an invalid character for the data type being parsed was encountered. For example, an alpha in a decimal numeric or a "9" in octal data.

-123 std_exponentTooLarge Exponent too large - Indicates the magnitude of an exponent was greater than 32000, see IEEE 488.2, 7.7.2.4.1.

-124 std_decimalTooLong Too many digits - Indicates the mantissa of a decimal numeric data element contained more than 255 digits excluding leading zeros, see IEEE 488.2, 7.7.2.4.1.

-128 std_numericNotAllowed Numeric data not allowed - Indicates that a legal numeric data element was received, but the device does not accept one in this position for the header.

-130 std_suffix Suffix - This error, as well as errors -131 through -139, are generated when parsing a suffix. This particular error message is used if the device cannot detect a more specific error.

-131 std_badSuffix Invalid suffix - Indicates the suffix does not follow the syntax described in IEEE 488.2, 7.7.3.2, or the suffix is inappropriate for this device.

-134 std_suffixTooLong Suffix too long - Indicates the suffix contain more than 12 characters, see IEEE 488.2, 7.7.3.4.

-138 std_suffixNotAllowed Suffix not allowed - Indicates that a suffix was encountered after a numeric element that does not allow suffixes.

-140 std_charData Character data - This error, as well as errors
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-141 std_invalidCharData</td>
<td>Invalid character data - Indicates that the character data element contains an invalid character or the particular element received is not valid for the header.</td>
</tr>
<tr>
<td>-144 std_charDataTooLong</td>
<td>Character data too long - Indicates the character data element contains more than twelve characters, see IEEE 488.2, 7.7.1.4.</td>
</tr>
<tr>
<td>-148 std_charNotAllowed</td>
<td>Character data not allowed - Indicates a legal character data element was encountered where prohibited by the device.</td>
</tr>
<tr>
<td>-150 std_stringData</td>
<td>String data - This error, as well as errors -151 through -159, are generated when parsing a string data element. This particular error message is used if the device cannot detect a more specific error.</td>
</tr>
<tr>
<td>-151 std_stringInvalid</td>
<td>Invalid string data - Indicates that a string data element was expected, but was invalid, see IEEE 488.2, 7.7.5.2. For example, an END message was received before the terminal quote character.</td>
</tr>
<tr>
<td>-158 std_stringNotAllowed</td>
<td>String data not allowed - Indicates that a string data element was encountered but was not allowed by the device at this point in parsing.</td>
</tr>
<tr>
<td>-160 std_blockData</td>
<td>Block data - This error, as well as errors -161 through -169, are generated when parsing a block data element. This particular error message is used if the device cannot detect a more specific error.</td>
</tr>
<tr>
<td>-161 std_badBlock</td>
<td>Invalid block data - Indicates a block data element was expected, but was invalid, see IEEE 488.2, 7.7.6.2. For example, an END message was received before the end length was satisfied.</td>
</tr>
<tr>
<td>-168 std_blockNotAllowed</td>
<td>Block data not allowed - Indicates a legal block data element was encountered, but not allowed by the device at this point in parsing.</td>
</tr>
<tr>
<td>-170 std_expr</td>
<td>Expression - This error, as well as errors -171 through -179, are generated when parsing an expression data element. This particular error message is used if the device cannot detect a more specific error.</td>
</tr>
<tr>
<td>-171 std_invalidExpression</td>
<td>Invalid expression - Indicates the expression data element was invalid, see IEEE 488.2, 7.7.7.2. For example, unmatched parentheses or an illegal character.</td>
</tr>
<tr>
<td>-178 std_exprNotAllowed</td>
<td>Expression data not allowed - Indicates a legal expression data was encountered, but was not allowed by the device at this point in parsing.</td>
</tr>
<tr>
<td>-180 std_macro</td>
<td>Macro - This error, as well as error -181 through -189, are generated when defining a macro or execution a macro. This particular error message is used if the device cannot detect a more specific error.</td>
</tr>
<tr>
<td>-181 std_validOnlyInsideMacro</td>
<td>Invalid outside macro definition - Indicates that a macro parameter place holder was encountered outside of a macro definition.</td>
</tr>
<tr>
<td>-183 std_invalidWithinMacro</td>
<td>Invalid inside macro definition - Indicates that the program message unit sequence, sent with a *DDT or a *DMC command, is syntactically invalid, see IEEE 488.2, 10.7.6.3.</td>
</tr>
<tr>
<td>-184 std_macroParm</td>
<td>Macro parameter - Indicates that a command inside the macro definition had the wrong number or type of parameters.</td>
</tr>
</tbody>
</table>
**-200 to -299 Execution Errors**

These errors are generated when something occurs that is incorrect in the current state of the instrument. These errors may be generated by a user action from either the remote or the manual user interface.

-200  **stdExecGen**  Execution - This event bit (Bit 4) indicates a PROGRAM DATA element following a header was outside the legal input range or otherwise inconsistent with the device's capabilities, see IEEE 488.2, 11.5.1.1.5.

-201  **stdInvalidWhileInLocal**  Invalid while in local

-202  **stdSettingsLost**  Settings lost due to rtl

-203  **stdCommandProtected**  Command protected - Indicates that a legal password-protected program command or query could not be executed because the command was disabled.

-210  **stdTrigger**  Trigger

-211  **stdTriggerIgnored**  Trigger ignored

-212  **stdArmIgnored**  Arm ignored

-213  **stdInitIgnored**  Init ignored

-214  **stdTriggerDeadlock**  Trigger deadlock

-215  **stdArmDeadlock**  Arm deadlock

-220  **stdParm**  Parameter - Indicates that a program data element related error occurred.

-221  **stdSettingsConflict**  Settings conflict - Indicates that a legal program data element was parsed but could not be executed due to the current device state.

-222  **stdDataOutOfRange**  Data out of range - Indicates that a legal program data element was parsed but could not be executed because the interpreted value was outside the legal range defined by the devices.

-223  **stdTooMuchData**  Too much data - Indicates that a legal program data element of block, expression, or string type was received that contained more data than the device could handle due to memory or related device-specific requirements.

-224  **stdIllegalParmValue**  Illegal parameter value - Indicates that the value selected was not part of the list of values given.

-225  **stdNoMemoryForOp**  Out of memory - The device has insufficient memory to perform the requested operation.

-226  **stdListLength**  Lists not same length - Attempted to use LIST structure having individual LIST's of unequal lengths.
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-230</td>
<td>std_dataCorruptOrStale</td>
</tr>
<tr>
<td></td>
<td>Data corrupt or stale - Indicates invalid data, a new reading started but not completed since the last access.</td>
</tr>
<tr>
<td>-231</td>
<td>std_dataQuestionable</td>
</tr>
<tr>
<td></td>
<td>Data questionable - Indicates that measurement accuracy is suspect.</td>
</tr>
<tr>
<td>-232</td>
<td>std_invalidFormat</td>
</tr>
<tr>
<td></td>
<td>Invalid format</td>
</tr>
<tr>
<td>-233</td>
<td>std_invalidVersion</td>
</tr>
<tr>
<td></td>
<td>Invalid version - Indicates that a legal program data element was parsed but could not be executed because the version of the data is incorrect to the device. For example, a not supported file version, a not supported instrument version.</td>
</tr>
<tr>
<td>-240</td>
<td>std_hardware</td>
</tr>
<tr>
<td></td>
<td>Hardware - Indicates that a legal program command or query could not be executed because of a hardware problem in the device.</td>
</tr>
<tr>
<td>-241</td>
<td>std_hardwareMissing</td>
</tr>
<tr>
<td></td>
<td>Hardware missing - Indicates that a legal program command or query could not be executed because of missing device hardware. For example, an option was not installed.</td>
</tr>
<tr>
<td>-250</td>
<td>std_massStorage</td>
</tr>
<tr>
<td></td>
<td>Mass storage - Indicates that a mass storage error occurred. The device cannot detect the more specific errors described for errors -251 through -259.</td>
</tr>
<tr>
<td>-251</td>
<td>std_missingMassStorage</td>
</tr>
<tr>
<td></td>
<td>Missing mass storage - Indicates that a legal program command or query could not be executed because of missing mass storage.</td>
</tr>
<tr>
<td>-252</td>
<td>std_missingMedia</td>
</tr>
<tr>
<td></td>
<td>Missing media - Indicates that a legal program command or query could not be executed because of missing media. For example, no disk.</td>
</tr>
<tr>
<td>-253</td>
<td>std_corruptMedia</td>
</tr>
<tr>
<td></td>
<td>Corrupt media - Indicates that a legal program command or query could not be executed because of corrupt media. For example, bad disk or wrong format.</td>
</tr>
<tr>
<td>-254</td>
<td>std_mediaFull</td>
</tr>
<tr>
<td></td>
<td>Media full - Indicates that a legal program command or query could not be executed because the media is full. For example, there is no room left on the disk.</td>
</tr>
<tr>
<td>-255</td>
<td>std_directoryFull</td>
</tr>
<tr>
<td></td>
<td>Directory full - Indicates that a legal program command or query could not be executed because the media directory was full.</td>
</tr>
<tr>
<td>-256</td>
<td>std_fileNotFound</td>
</tr>
<tr>
<td></td>
<td>File name not found - Indicates that a legal program command or query could not be executed because the file name was not found on the media.</td>
</tr>
<tr>
<td>-257</td>
<td>std_fileName</td>
</tr>
<tr>
<td></td>
<td>File name - Indicates that a legal program command or query could not be executed because the file name on the device media was in error. For example, an attempt was made to read or copy a nonexistent file.</td>
</tr>
<tr>
<td>-258</td>
<td>std_mediaProtected</td>
</tr>
<tr>
<td></td>
<td>Media protected - Indicates that a legal program command or query could not be executed because the media was protected. For example, the write-protect switch on a memory card was set.</td>
</tr>
<tr>
<td>-260</td>
<td>std_expression</td>
</tr>
<tr>
<td></td>
<td>Expression</td>
</tr>
</tbody>
</table>
-261 std_math Math in expression
-270 std_macroExecution Macro - Indicates that a macro related execution error occurred.
-271 std_macroSyntax Macro syntax - Indicates that a syntactically legal macro program data sequence, according to IEEE 488.2, 10.7.2, could not be executed due to a syntax error within the macro definition.
-272 std_macroExec Macro execution - Indicates that a syntactically legal macro program data sequence could not be executed due to some error in the macro definition, see IEEE 488.2, 10.7.6.3.
-273 std_badMacroName Illegal macro label - Indicates that the macro label was not accepted, it did not agree with the definition in IEEE 488.2, 10.7.3.
-274 std_macroPlaceholder Macro parameter - Indicates that the macro definition improperly used a macro parameter placeholder, see IEEE 4882, 10.7.3.
-275 std_macroTooLong Macro definition too long - Indicates that a syntactically legal macro program data sequence could not be executed because the string of block contents were too long for the device to handle, IEEE 488.2, 10.7.6.1.
-276 std_macroRecursion Macro recursion - Indicates that a syntactically legal macro program data sequence count not be executed because it would be recursive, see IEEE 488.2, 10.7.6.6.
-277 std_cantRedefineMacro Macro redefinition not allowed - Indicates that redefining an existing macro label, see IEEE 488.2, 10.7.6.4.
-278 std_macroNotFound Macro header not found - Indicates that a legal macro label in the *GMS?, see IEEE 488.2, 10.13, could not be executed because the header was not previously defined.
-280 std_program Program
-281 std_cantCreateProgram Cannot create program
-282 std_illegalProgramName Illegal program name
-283 std_illegalVarName Illegal variable name
-284 std_programRunning Program currently running
-285 std_programSyntax Program syntax
-286 std_programRuntime Program runtime
-290 std_memoryUse Memory use
-291 std_execOutOfMemory Out of memory
-292 std_nameNotFound Referenced name does not exist
-293 std_nameAlreadyExists Referenced name already exists
-294 std_incompatibleType Incompatible type

**-300 to -399 SCPI Specified Device-Specific Errors**

A device-specific error indicates that the instrument has detected an error that occurred because some operations did not properly complete, possibly due to an abnormal hardware or firmware condition. For example, an attempt by the user to set an out of range value will generate a device specific error. When one of these errors is generated, the device specific error bit in the event status register is set.

-300 std_deviceSpecific Device specific - This event bit (Bit 3) indicates that a device operation did not properly complete due to some condition, such as overrange see IEEE 488.2, 11.5.1.1.6.
-310 std_system System
-311 std_memory Memory - Indicates some physical fault in the devices memory, such as a parity error.
-312 std_PUDmemoryLost PUD memory lost - Indicates protected user data saved by the *PUD command has been lost, see IEEE 488.2, 10.27.
-313 std_calMemoryLost Calibration memory lost - Indicates that nonvolatile calibration data used by the *CAL? command has been lost, see IEEE 488.2, 10.2.
-314 std_savRclMemoryLost Save/recall memory lost - Indicates that the nonvolatile data saved by the *SAV command has been lost, see IEEE 488.2, 10.33.
-315 std_configMemoryLost Configuration memory lost - Indicates that nonvolatile configuration data saved by the device has been lost.
-320 std_storageFault Storage fault - Indicates that the firmware detected a fault when using data storage. This is not an indication of physical damage or failure of any mass storage element.
-321 std_outOfMemory Out of memory - An internal operation needed more memory than was available
-330 std_selfTestFailed Self-test failed - Indicates a problem with the device that is not covered by a specific error message. The device may require service.
-340 std_calFailed Calibration failed - Indicates a problem during calibration of the device that is not covered by a specific error.
-350 std_queueOverflow Queue overflow - Indicates that there is no room in the queue and an error occurred but was not recorded. This code is entered into the queue in lieu of the code that caused the error.
-360 std_comm  Communication - This is the generic communication error for devices that cannot detect the more specific errors described for error -361 through -363.

-361 std_parity  Parity in program message - Parity bit not correct when data received for example, on a serial port.

-362 std_framing  Framing in program message - A stop bit was not detected when data was received for example, on a serial port (for example, a baud rate mismatch).

-363 std_inputBufferOverrun  Input buffer overrun - Software or hardware input buffer on serial port overflows with data caused by improper or nonexistent pacing.

-400 to -800 Query and System Errors

A Query error is generated either when data in the instrument's GPIB output queue has been lost, or when an attempt is being made to read data from the output queue when no output is present or pending.

-400 std_queryGen  Query - This event bit (Bit 2) indicates that an attempt to read data from the Output Queues when no output is present or pending, to data in the Output Queue has been lost see IEEE488.2, 11.5.1.1.7.

-410 std_interrupted  Query INTERRUPTED - Indicates the test set has been interrupted by a new program message before it finishes sending a RESPONSE MESSAGE see IEEE 488.2, 6.3.2.3.

-420 std_unterminated  Query UNTERMINATED - Indicates an incomplete Query in the program see IEEE 488.2, 6.3.2.2.

-430 std_deadlocked  Query DEADLOCKED - Indicates that the Input Buffer and Output Queue are full see IEEE 488.2, 6.3.1.7.

-440 std_responseNotAllowed  Query UNTERMINATED after indefinite response - Indicates that a query was received in the same program message after a query requesting an indefinite response was executed see IEEE 488.2, 6.5.7.5.

-500 std_powerOn  Power on

-600 std_userRequest  User request

-700 std_requestControl  Request control

-800 std_operationComplete  Operation complete

PNA Specific SCPI Errors

100 dupWindNum  "Duplicate window number"

101 windNumNotFound  "Window number not found"

102 failedWindCreate  "Window creation failed"
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>103</td>
<td>noCalcParamSelection</td>
</tr>
<tr>
<td>104</td>
<td>dupMeasName</td>
</tr>
<tr>
<td>105</td>
<td>dataNotFound</td>
</tr>
<tr>
<td>106</td>
<td>measNotFound</td>
</tr>
<tr>
<td>107</td>
<td>traceNotFound</td>
</tr>
<tr>
<td>108</td>
<td>notImplemented</td>
</tr>
<tr>
<td>109</td>
<td>noDocument</td>
</tr>
<tr>
<td>110</td>
<td>dupTraceNum</td>
</tr>
<tr>
<td>111</td>
<td>titleStrTooLong</td>
</tr>
<tr>
<td>112</td>
<td>memoryNotFound</td>
</tr>
<tr>
<td>113</td>
<td>exceedMaxTraces</td>
</tr>
<tr>
<td>114</td>
<td>SerNumNotFound</td>
</tr>
<tr>
<td>115</td>
<td>LoadFailed</td>
</tr>
<tr>
<td>116</td>
<td>StoreFailed</td>
</tr>
<tr>
<td>117</td>
<td>File</td>
</tr>
<tr>
<td>118</td>
<td>measChanConflict</td>
</tr>
<tr>
<td>119</td>
<td>exceedMaxWindows</td>
</tr>
<tr>
<td>120</td>
<td>markerNotFound</td>
</tr>
<tr>
<td>121</td>
<td>diagnostic</td>
</tr>
<tr>
<td>122</td>
<td>channelNotFound</td>
</tr>
<tr>
<td>123</td>
<td>exceedMaxMeasurements</td>
</tr>
<tr>
<td>124</td>
<td>parameterOutOfRange</td>
</tr>
<tr>
<td>125</td>
<td>userRangeNotValid</td>
</tr>
<tr>
<td>126</td>
<td>referenceMarkerNotFound</td>
</tr>
<tr>
<td>127</td>
<td>sweepSegmentNotFound</td>
</tr>
<tr>
<td>128</td>
<td>markerNotDelta</td>
</tr>
</tbody>
</table>
129 printoutFailed  "Attempt to output to a printer failed."
130 memory_trace_not_compatible  "Memory not compatible. Trace Math not applied."
131 trace_math_reset  "Memory not compatible. Trace Math turned off."
132 hw_read_failed  "Hardware read failed."
133 hw_write_failed  "Hardware write failed."
134 dsp_active  "Failed because DSP was not halted."
135 secure_memory  "Attempt to access secure memory region."
136 snum_protected  "The serial number is protected."
137 snum_format_bad  "The serial number format is bad."
138 snum_already_set  "The serial number is already set."
139 hw_setting_failed  "Hardware setting failed."
140 cal_access_failed  "Calibration data access failed."
141 db_access_failed  "Database access failed."
142 memory_range_exceeded  "Command exceeds usable memory range."
143 lost_phase_lock  "Phase lock has been lost."
144 over_power  "Detected too much power at input."
145 ee_wrt_failed  "EEPROM write failed."
146 yig_cal_failed  "YTO calibration failed."
147 ramp_cal_failed  "Analog ramp calibration failed."
148 dspcom_bad  "DSP communication failed."
149 no_license_found  "Request failed. The required license was not found."
150 argLimited  "The argument was out of range"
151 markerBWNotFound  "The Marker Bandwidth was not found."
153 peakNotFound  "The Peak was not found."
154 targetNotFound  "The Target search value was not found."
155 calNotImpl  "The Calibration feature requested is not implemented."
calClassNotValidForCalType  "SENS:CORR:CCH measurement selection set to none"

calNotValidForConfidenceChe  "Selected measurement does not have a calibration valid for Confidence Check"

invalidPort  "Specified port is out of range"

invalidPortPath  "ROUTE:PATH:DEF:PORT  x, y does not match measurement; setting to defaults"

ioInvalidWrite  "Attempted I/O write while port set to read only."

ioInvalidRead  "Attempted I/O read from write only port."

calsetNotLocated  "Requested Cal Set was not found in Cal Set Storage."

noCalSetSelected  "There is no Cal Set currently selected for the specified channel."

cantDeleteCalSetInUse  "Cannot delete a Cal Set while it is being used."

calsetStimChange  "Channel stimulus settings changed to match selected Cal Set."

exceedMaxCalSets  "Exceeded the maximum number of cal sets."

calCouldNotTurnOn  "A valid calibration is required before correction can be turned on."

standardMeasurementRequired  "The attempted operation can only be performed on a standard measurement type."

noDivisorBuffer  "A valid divisor buffer is required before normalization can be turned on."

InvalidReceiverPowerCalParagraph  "Receiver power cal requires the measurement to be of unratioed power."

calCouldNotConfigure  "Could not configure the Electronic Calibration system. Check to see if the module is plugged into the proper connector."

measHasNoMemoryAlg  "This measurement does not support memory operations"

measHasNoNormalizeAlg  "This measurement does not support normalize operations."

userCharacterizationNotFound  "User characterization was not found in the Electronic Calibration module."

measInvalidBufferSize  "The data provided has an invalid number of points. It could not be stored."
Technical Support

Click on the region of interest.

- For assistance with your test and measurement needs go to [www.agilent.com/find/assist](http://www.agilent.com/find/assist)
- Or contact the test and measurement experts at Agilent Technologies.

**Other Support Topics**

**United States:**
(tel) (+1) 800 452 4844  
(alt) (+1) 303 662 3999  
(fax) (+1) 888 900 8921

**Canada**
(tel) 1 877 894 4414  
(fax) 1 (905) 206 4120

**Europe:**

**Austria**
(tel) 0820 87 44 11*  
(fax) 0820 87 44 22
Belgium
(tel) (+32) (0)2 404 9340
(alt) (+32) (0)2 404 9000
(fax) (+32) (0)2 404 9395

Denmark
(tel) (+45) 7013 1515
(alt) (+45) 7013 7313
(fax) (+45) 7013 1555

Finland
(tel) 08 0052 4000
(alt) (+358) 10 855 2100
(fax) (+358) 92 536 0176

France
(tel) 0825 010 700*
(alt) (+33) (0)1 6453 5623
(fax) 0825 010 701*

Germany
(tel) 01805 24 6333*
(alt) 01805 24 6330*
(fax) 01805 24 6336*

Ireland
(tel) (+353) (0)1 890 924 204
(alt) (+353) (0)1 890 924 206
(fax) (+353) (0)1 890 924 024

Israel
(tel) (+972) 3 9288 500
(fax) (+972) 3 9288 501

Italy
(tel) (+39) (0)2 9260 8484
(fax) (+39) (0)2 9544 1175

Luxemburg
(tel) (+32) (0)2 404 9340
Netherlands
(tel) (+31) (0)20 547 2111
(alt) (+31) (0)20 547 2000
(fax) (+31) (0)20 547 2190

Russia
(tel) (+7) 095 797 3963
(alt) (+7) 095 797 3900
(fax) (+7) 095 797 3901

Spain
(tel) (+34) 91 631 3300
(alt) (+34) 91 631 3000
(fax) (+34) 91 631 3301

Sweden
(tel) 0200 88 22 55*
(alt) (+46) (0)8 5064 8686
(fax) 020 120 2266*

Switzerland (French)
(tel) 0800 80 5353 opt. 2*
(alt) (+33) (0)1 6453 5623
(fax) (+41) (0)22 567 5313

Switzerland (German)
(tel) 0800 80 5353 opt. 1*
(alt) (+49) (0)7031 464 6333
(fax) (+41) (0)1 272 7373

Switzerland (Italian)
(tel) 0800 80 5353 opt. 3*
(alt) (+39) (0)2 9260 8484
(fax) (+41) (0)22 567 5314

United Kingdom
(tel) (+44) (0)7004 666666
Japan:
(tel) 0120 421 345
(alt) (+81) 426 56 7832
(fax) 0120 421 678

Latin America:

Mexico
(tel) (+52) 55 5081 9469
(alt) 01800 5064 800
(fax) (+52) 55 5081 9467

Brazil
(tel) (+55) 11 4197 3600
(fax) (+55) 11 4197 3800

Australia:
((tel) 1800 629 485
(alt) 1800 143 243
(fax) 1800 142 134

New Zealand
(tel) 0 800 738 378
(fax) 64 4 495 8950

Asia Pacific:

China
(tel) 800 810 0189
(alt) (+86) 10800 650 0021
(fax) 800 820 2816

**Hong Kong**
(tel) 800 930 871
(alt) (+852) 3197 7889
(fax) (+852) 2 506 9233

**India**
(tel) 1600 112 929
(fax) 000800 650 1101

**Malaysia**
(tel) 1800 888 848
(alt) 1800 828 848
(fax) 1800 801 664

**Singapore**
(tel) 1800 375 8100
(fax) (+65) 6836 0252

**South Korea**
(tel) 080 769 0800
(alt) (+82) 2 2004 5004
(fax) (+82) 2 2004 5115

**Taiwan**
(tel) 0800 047 866
(alt) 00801 651 317
(fax) 0800 286 331

**Thailand**
(tel) 1800 226 008
(alt) (+66) 2 268 1345
(fax) (+66) 2 661 3714
3.8 GHz Frequency Adjustment

This routing adjusts the internal fixed-frequency YIG Oscillator to 3.8 GHz by changing a DAC value. This DAC value is stored in the analyzer's non-volatile memory. This adjustment is only needed on some PNA models. Typically, the oscillator can be set to within 12 kHz of 3.8GHz; it is not necessary for it to be exactly 3.8GHz.

Spectrum Analyzers Compatibility

This routine is compatible with Agilent 856x and 859x spectrum analyzers, and may also work on some other Agilent spectrum analyzers.

If no compatible analyzer is available, select "NONE" for the spectrum analyzer. You can then adjust the DAC manually by viewing the 3.8 GHz signal on another analyzer.

Procedure (For Compatible Spectrum Analyzers Only)

**Note:** The viewable 3.8 GHz signal level will be low; typically be around -70dBm. Do not use any attenuators in the adjustment, other than the default 10 dB attenuation used in most spectrum analyzers.

1. Connect spectrum analyzer input to the network analyzer's PORT 1 output.
2. Connect GPIB cable from analyzer to spectrum analyzer. Make sure no other controllers are active on the same connection.
3. Set the spectrum analyzer GPIB address to 18.
4. In the analyzer **System** menu, point to **Service, Adjustments**, and click **3.8 GHz Freq. Adjust**.
5. Click **Begin Adj.**, for the program to adjust the internal oscillator for minimal error and store the results. When the status area indicates the adjustment is complete, select **Exit**.

Procedure (For Non-Compatible Spectrum Analyzers Only)

**Note:** The viewable 3.8 GHz signal level will be low; typically be around -70dBm. Do not use any attenuators in the adjustment, other than the default 10 dB attenuation used in most spectrum analyzers.

1. Connect the spectrum analyzer input to the network analyzer's PORT 1 output.
2. Set the spectrum analyzer to the following settings:
   - Center frequency = 3.8 GHz
   - Span = 100 MHz
   - Bandwidth = 10 kHz
   - Scaling where a signal of -70 dBm will be clearly visible

3. In the analyzer **System** menu, point to **Service, Adjustments**, and click **3.8 GHz Freq. Adjust**.

4. Under **Spectrum Analyzer**, select **NONE** option for spectrum analyzer.

5. Click **Begin Adj**.

6. The application presets the DAC to an initial value equal to the current value stored. View the spectrum analyzer to see if the signal is above or below 3.800 GHz.
   - If the signal frequency is above 3.8 GHz, move the slider to adjust the DAC to a lower value (left).
   - If the frequency is below 3.8 GHz, move the slider to adjust the DAC to a higher value (right).

**Note:** The valid DAC values are from 0 to 4095. The oscillator will shift about 23 kHz per DAC value.

7. Set the DAC value to reach a frequency very close to 3.8 GHz. If you made large changes in DAC values, allow several seconds for thermal effects to stabilize.

8. Change the spectrum analyzer settings to better view the frequency signal:
   - Frequency span = 500 kHz
   - Bandwidth = 3 kHz

9. Change the DAC value to keep the signal centered at 3.8 GHz.

10. Once you have determined the correct DAC value, click **SAVE DAC** to permanently store that value into EEPROM. Click **Exit**.

**Note:** If large changes are made to the existing DAC value, then this test should be repeated again after 15-30 minutes. This allows the thermal effects to fully stabilize.
10 MHz Reference Frequency Adjustment

This routine adjusts the analyzer's internal time-base to exactly 10 MHz by changing a DAC value. This DAC value is stored in the analyzer's non-volatile memory. This routine should only be necessary in the following situations:

- The frequency reference assembly is replaced.
- The 10 MHz reference has drifted significantly from the factory adjusted value.

**WARNING:** The range of this adjustment is only about 20 Hz. It is highly recommended that a very accurate frequency standard be used to measure this 10 MHz signal.

Frequency Counter Compatibility

This procedure uses SCPI commands (over GPIB) to communicate with the frequency counter. It should work with the Agilent 5313xA, 5315xA, 53181A series of counters as well as the older 5350 series.

If no compatible counters are available, select the "Manual" mode of operation. If you do choose the manual mode, you must input the measured frequency manually.

Procedure for GPIB Counters Only

1. Connect the analyzer rear panel 10 MHz Reference output to the frequency counter.
2. Connect a GPIB cable from the analyzer to the counter. Make sure no other controllers are active on the same connection.
3. If applicable, connect the house frequency standard to the counter reference input.
4. Set the counter GPIB address to 03. Ensure that the counter is the only device at this address.
5. In the analyzer **System** menu, point to **Service, Adjustments**, and click **10 MHz Freq. Adjust**.
6. Click **Begin Adj**. The application adjusts the internal reference for minimal error and stores the results.
7. Click **Read Freq** to trigger another reading of the 10 MHz signal.
8. Click **Read DAC** to view the current DAC value stored in the analyzer's non-volatile memory (value = 0 - 4095).
9. When the status area indicates the adjustment is complete, click **Exit**.

Procedure for Non-GPIB Counters

1. Connect the counter input to the rear panel 10 MHz Reference Output.
2. Set the counter to at least 1 Hz resolution.
3. If applicable, connect the house-frequency standard to the counter reference input.
4. In the analyzer **System** menu, point to **Service, Adjustments** and click **10 MHz Freq. Adjust**.
5. Under **Frequency Counter**, select **Manual**.

6. Click **Begin Adj**.

7. The application presets the DAC to an initial value. Enter the measured frequency offset from 10 MHz. If the measured frequency is less than 10 MHz, use a minus (-) sign to indicate a negative error. For example:

   - If the counter reads 10000003.5 Hz, enter 3.5 (or +3.5) in the indicated window.
   - If the counter reads 9999997.8 Hz, enter -2.2 in the indicated window.

8. The adjustment loops at least 3 times unless the entered value is exactly zero.

   - Click **Read Freq** to trigger another reading of the 10 MHz signal.
   - Click **Read DAC** to view the current DAC value stored in the analyzer's non-volatile memory (value = 0 - 4095).

9. When the status area indicates the adjustment is complete, click **Exit**.

**Note:** If the counter is misreading the frequency, it may be necessary to attenuate the input, or set the input impedance to 50 ohms, or both.
Display Test

The PNA screen should be bright with all annotations and text readable. The display test allows you to check for non-functioning pixels and other problems.

Note If the display is dim or dark, refer to “Troubleshooting LCD Display Problems” in the PNA Service Guide.

What Is a Damaged Pixel?
A pixel is a picture element that combines to create the image on the display. They are about the size of a small pin point. Damaged pixels can be either “stuck on” or “dark.”

- Stuck on pixel - red, green, or blue; always displayed regardless of the display setting. It will be visible on a dark background.
- Dark pixel - always dark; displayed against a background of its own color.

How to Run the Display Test
On the System menu, point to Service, and then click Display Test.
A multi-color screen is displayed. Be prepared to look for the symptoms described below. Click the Start Test button. To continue to the next test, click the moving Next Test button. The button moves to allow you to see all of the display. After the test is completed, the display defaults to the network analyzer screen.

How to Identify a Faulty Display
One or more of the following indicate a bad display:

- Complete row or column of “stuck on” or “dark” pixels
- More than six “stuck on” pixels (but not more than three green)
- More than twelve “dark” pixels (but not more than seven of the same color)
- Two or more consecutive “stuck on” pixels or three or more consecutive “dark” pixels (but no more than one set of two consecutive dark pixels)
- “Stuck on or “dark” pixels less than 6.5 mm apart (excluding consecutive pixels)

If any of these symptoms occur, your display is considered faulty. See the Service Guide for your PNA model.
**LO Power Adjustment**

This procedure adjusts the receiver's LO input power to a specific level by changing DAC values. These DAC values are then stored in the analyzer's non-volatile memory. The procedure will vary depending upon the model number. This adjustment is only applicable to some PNA models.

**Power Meter Compatibility**

This routine is only compatible with the Agilent EPM series of power meters. Different sensors may be used. For 9 GHz analyzers and below, an 8482 or E4412A sensor can be used. For the higher frequency units (20 GHz or above), a sensor must be able to measure a maximum of 20 GHz. At no time during this test will a frequency higher than 20 GHz be measured, even if the PNA has a maximum frequency of 50 GHz.

If the older HP 84xx series of sensors are used, the correct calibration data should be entered into the appropriate cal table of the EPM series power meter, although for this adjustment, high accuracy is not required. Inaccuracies in the order of several tenths of a dB are acceptable.

**Procedure**

1. Allow the analyzer and power meter to warm up for 30 minutes.
2. Manually zero and calibrate the power sensor. (This allows you to skip this step later)
3. Connect a GPIB cable from the analyzer to the power meter. Make sure no other controllers are active on the same connection.
4. Set the power meter GPIB address to 13. (others can also be used; 13 is the default)
5. Remove the outer cover on the analyzer.
6. In the PNA System menu point to Service, then Adjustments, and click LO Power Adjust.
7. Connect the power sensor to the LO output, using adapters if needed. The LO output location varies with model number. Click on the LO Power Adjust Setup menu selection to see a diagram of the exact location.
8. **For 9 GHz units and below:**
   Click Begin Adj to start the LO power cal routine. The routine adjusts the power level for each band (1 through 3) to fall within certain bounds. If any changes are made, it automatically stores them.

   **For 20GHz units and above:**
   If using an 84xx power sensor, click Configure and select the proper sensor model number. Click Close. Click Calibrate to begin the adjustment. The entire calibration process takes about 5 minutes. Once completed, you can verify the current calibration accuracy by clicking Verify Cal.
   **Note:** Correction constants are defaulted at the beginning of calibration. Once the calibration process has started, it must be completed in order to regenerate proper data.

9. Click Read DAC to view the current DAC values (0-4095) stored in the PNA non-volatile memory for each band (0-7).
10. When the message/status area indicates the adjustment is complete, click Exit.

11. Reconnect the semi-rigid cable and replace the covers.
Offset LO Power Adjustment

**Note:** This adjustment is only performed on PNAs with Frequency Offset Mode (option 080), and only on certain models.

The Offset LO Adjustment sets the LO power for the offset mixer to a consistent value across all bands. It requires access to the internal components of the PNA so that the power sensor can be connected to the LO output. Because the LO frequency does not exceed 3 GHz, almost any power sensor can be used. The adjustment is relatively simple and only takes a couple of minutes.

**When to perform**

This adjustment should be performed when any of the following occur:

- the A13 Frequency Offset Receiver is significantly modified or replaced
- the A9 Synthesizer is replaced (that is where the correction data resides)

**How to perform Offset LO Power adjustment**

1. To start the Offset LO adjustment, click **System**, point to **Service, Adjustments**, then click **Offset LO Adjust**.

2. You will be prompted to zero and cal the power sensor. (You can do this before beginning.)

3. Connect the sensor to J3 of the A13 LO output by removing the existing cable (or simply disconnecting one end) as shown in the Set-up diagram in the program.

4. Connect the power meter to the PNA using a GPIB cable. Make sure the GPIB address shown in the program matches the actual power meter address (default is 13.)

5. Click **Adjust** to begin the test and follow the instructions.

The program automatically adjusts all bands; no user input is needed. The program repeats several times as this is an iterative process. The progress of the adjustment is shown on the screen.

The Configure menu selection is for factory personnel ONLY.

Once completed, to verify the actual results, click Verify.

Upon exiting, the PNA application will restart; this takes several seconds.
Operator's Check

Overview

How to Run the Operator's Check

Operators Check Dialog Box Help

Overview

The Operator's Check should be performed when you first receive your PNA, and any time you wish to have confidence that the PNA is working properly.

Notes

- The Operator's Check does not verify performance to specifications. To verify PNA performance to specifications, run System Verification.

- Allow the PNA to warm up for 90 minutes before considering a failed test to be valid.

- The Operator's Check can NOT be run with a Multiport test set enabled. However, you can run a performance check as described in the Test Set User's Guide. See the N44xx User's Guide.

The Pass/Fail criteria used in the Operator's Check identifies obvious failures in the following portions of the PNA hardware:

- Repeatability of the RF switch in the test set
- Attenuation ranges of the test port attenuators (if installed)
- Calibration of the receivers
- Frequency response of the receivers
- Phase lock and leveling
- Noise floor and trace noise
**How to Run the Operator’s Check**

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
</table>

**For PNA-L and E836x models**

1. **Navigate using** **MENU/DIALOG**
   1. Click **System**
   2. then **Service**
   3. then **Operator’s Check**

**For PNA-X and ’C’ models**

1. Press **SYSTEM**
2. then **[Service]**
3. then **[Operator’s Check]**
4. then **Operator’s Check**

---

1. Connect one or more standards (see [Configure](#)).
2. Click **Begin** and **Continue** (if necessary) until “Operator’s Check is complete!” appears.

![Operator’s Check](image)
This dialog box will look slightly different, depending on PNA model number and installed options. Some of the tests are not run if the appropriate option is not installed.

**Operators Check** dialog box help

**Note:** It is normal for a momentary unleveled condition to appear during portions of the Operators Check.

**Configure**

- **Prompt for attachment of Short / Open**  If you do not have enough shorts or opens for all test ports, you will be prompted to move the standard to the next test port. Connect either a short or open to port 1, then click Begin.

- **Shorts / Opens are attached to all ports**  Connect either a short or open for each test port, then click Begin. All ports are tested without interruption. You can mix shorts and opens on the test ports.

**PNA**  Shows information about the PNA that is being tested.

**Legend**  Shows the status icons used in the Operator's Check and their meaning. **Pending Pass** means that a portion of the testing has been completed successfully.

**Results**  Shows the current status of each test. Click on the test name to learn how that test is performed. This may help in troubleshooting failed tests. If any tests Fail, refer to Chapter 3 of the PNA service guide.

**Begin**  Starts the Operator's Check.

**View Results**  Shows all results in text format. Failed items are preceded by `===>>>`. This text file can be printed or saved with a unique file name to compare results with previous or subsequent testing.

**Exit**  Ends the program and closes the window.

---

**Last Modified:**

5-Feb-2008  Added multiport test set note
**Option H11 Test Verification**

The PNA Option H11 Test / Verification utility verifies the PNA's option H11 functionality. The associated H11 inputs and outputs are tested and the results are compared against expected levels at the factory. These tests are used in conjunction with the Operator's Check and System Verification programs to check PNA functionality.

### How to Run Option H11 Verification

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For PNA-L and E836x models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click System</td>
</tr>
<tr>
<td>2. then Service</td>
<td>2. then Service</td>
</tr>
<tr>
<td>3. then Rear Panel/H11 Test or H11 Verification</td>
<td>3. then Rear Panel/H11 Test or H11 Verification</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Not Available</td>
<td>1. Not Available</td>
</tr>
</tbody>
</table>

**Option H11 Verification** dialog box help

The Option H11 Verification software leads you through each of the tests listed on the Configure Tab.

**Important**: Required Configuration of the PNA being tested:

- All connections to external devices and test sets must be removed from the Option H11 connectors and Test Set I/O connector on the PNA rear panel.

- On the Millimeter Module Configuration dialog box, make the following settings:
- PNA Model E8361A - Select Agilent N5250A, and check **Use Standard PNA operation when N5260A is NOT connected.**
- All other PNA models - Select **Standard PNA.**

**Run Tab**
The list of test equipment is the complete list that is required to run ALL of the tests (recommended).

**Start**  Click to run the tests that are selected on the Configure Tab.

**Exit**  Exits Option H11 Verification.

**Configure Tab**
All of the tests are selected by default. Tests can be run separately for troubleshooting purposes by checking or clearing the boxes.

**Power Tests**
Output power is measured at each of the selected **RF and LO Test Set connectors** on the rear panel. The results are displayed against limit lines that represent the **expected levels at the factory**.

**Equipment Required:**
- **PNA supported power meter and 26.5 GHz power sensor** (Recommended: Agilent E4418/19 Power Meter and E4413A Power Sensor). Set GPIB address = 13.
- GPIB and Sensor cables.
- On E8361A ONLY, a device is required to simulate an N5260A Test Set is connected to the Test Set I/O. All other PNA models do NOT require this device for this test.
  - Agilent Service personnel: Use the E8361-60063 Test Set I/O Tester.
  - All others: On a commercially available DB-25 male connector, connect a jumper between pins 12 and 22. See the **Test Set I/O connector diagram**. Insert the male connector into the PNA rear panel Test set I/O connector.

**External IF Input and Crosstalk Tests**
A 8.33 MHz signal is injected into each of the selected **8.33 MHz IF IN connectors** on the rear panel. The signal is measured at each of the selected receivers. The results are displayed against limit lines that represent the **expected levels at the factory**.

**Equipment Required:**
- 33120A Function Generator. Set GPIB address = 10.
- 10 dB Attenuator (Agilent 8493B or equivalent) and BNC adapters.
- GPIB and BNC cables
**External Pulse Input Test**

A pulse train is injected into each of the selected Pulse IN connectors on the rear panel. The signal is measured at each of the selected receivers. The results are displayed against limit lines that represent the expected levels at the factory.

**Note**: A noisy pulse generator can cause a false failure of the "1% Duty Cycle Pulse Input" portion of the External Pulse Input Test.

Equipment Required:

- 8110A or 81110A Pulse Generator. Set GPIB address = 11.
- GPIB and BNC cables

**Test Set I/O Connector Test**

Tests the ability of the PNA to detect and control an external test set through the PNA rear panel TEST SET I/O connector.

Equipment Required:

- Agilent Service personnel: Use the E8361-60063 Test Set I/O Tester.
- All others: Use a DVM to measure the TTL voltage level on the following pin numbers using pin 1 as ground:
  - Pins: 3, 4, 5, 6, 8, 9, 10, 11, 17, 19, 20, 21, 22, 23, 24
  - See Test Set I/O connector diagram to determine the pin number location.
  - The software toggles all pins HIGH, then LOW.
  - TTL HIGH should read approximately +5.0 volts.
  - TTL LOW should read approximately +0.1 volts.

**Help Tab**

Displays the Option H11 Verification program revision number and information about the PNA being tested.

**Advanced Help** Click to display this help topic.

**View Results** Appears after the test is run. Click to show all results in text format. Failed items are preceded by "=>>> Fail." This text file can be printed or saved with a unique file name to compare results with previous or subsequent testing.

---

Last Modified: 10-Mar-2008  MX added UI
**IF Gain Adjustment**

**Phase-Lock IF Gain Adjustment for E836x models only!**

The E836x A/B PNA models have a variable gain control for the phase-lock loop IF signal. By dynamically changing the gain as a function of frequency and power, the phase-lock signal amplitude can be adjusted to a constant level for the entire operating range of the instrument. This constant level is important for phase-lock acquisition and stability.

**When to perform**

Phase-Lock IF Gain Adjustment should be performed when any of the following occur:

- A source calibration
- An assembly in the reference receiver path (R1,R2) is replaced.
- The Test Set Motherboard is replaced
- The Phase Lock board is replaced
- **Phase Lock Lost** error message appears after replacing a source or receiver assembly
- The external R path has changed. For example, when a multiport test set with R channel path has been added or removed.

**How to perform Phase-Lock IF Gain adjustment**

Ensure the Reference Channel paths are properly configured and the connections are properly torqued.

1. From the **System** menu, click **Service**, then **Adjustments**, then **IF Gain Adjustment**.
2. Under Select Specials, select **None**.
3. No connections to the test ports are required.
4. Click **Begin Adj**. The adjustment takes about a minute to complete.
5. The advanced screen is for factory personnel only.

---

**IF Gain Adjustment for N524x models only!**

The N524x PNA models have a variable gain control for the loop IF signal. By dynamically changing the gain as a function of frequency and power, the phase-lock signal amplitude can be adjusted to a constant level for the entire
operating range of the instrument. This constant level is important for phase-lock acquisition and stability.

**When to perform**

IF Gain Adjustment should be performed when any of the following occur:

- An assembly in the reference receiver path (R1,R2) is replaced.
- The Test Set Motherboard is replaced
- The IF Mux assembly or SPAM assembly is replaced

**How to perform IF Gain adjustment**

Ensure the Reference Channel paths are properly configured and the connections are properly torqued.

1. From the **System** menu, click **Service**, then **Adjustments**, then **IF Gain Adjustment**.
2. An appropriate power meter and sensor will be required. The sensor must cover the upper frequency range of the PNA. A good quality cable will also be needed. Adapters can be used as needed.
3. Connect the power meter to the GPIB port labeled System Controller.
4. Click **Begin Adj.**
5. Set the GPIB address in the program accordingly and select the sensor being used.
6. Follow the instructions displayed in the program.

The adjustment takes about 5 minutes to complete for a 26.5 GHz PNA. Higher frequency units may take longer.

The Default menu selection is for factory personnel only. This will preset all values to default levels for troubleshooting purposes only. If this is selected, a full IF gain adjustment will need to be performed.

Last modified:

3/7/07   Added N5242x information.
System Verification

The performance of the network analyzer is specified in two ways: system specifications, and instrument specifications. It is the end user's responsibility to determine which set of specifications is applicable to their use of the PNA.

A network analyzer measurement “system” includes the analyzer, calibration kit, test cables, and any necessary adapters. The system verification software in the PNA is used to verify the system’s conformance to the “system” specifications. A “pass” result demonstrates that the analyzer, test cables, and adapters, perform correctly as a system. It DOES NOT demonstrate that any one component performs according to its individual specifications. A change to any part of this measurement system requires a re-verification of the system.

Instrument specifications specify the network analyzer’s uncorrected measurement port characteristics and its output and input behavior. The PNA performance tests are used to verify the analyzer’s conformance to “instrument” specifications.

The system verification utility verifies the PNA system specifications by automatically measuring the magnitude and phase for all four S-parameters for each verification device, and comparing the values against the following:

- Factory measured data from files on the verification disk
- Limit lines based on the measurement uncertainty

System Verification requires the use of a calibration kit and verification kit which has been certified within the past 12 months by Agilent. System Verification can NOT be used to perform this kit certification.

Operator's Check should also be performed to verify the basic operation of the PNA.

Equipment Used in the System Verification

Precautions for Handling Airlines

Flow Diagram of Procedure

Procedure for System Verification

If the System Fails the Verification Test

Interpreting the Verification Results

Notes

- Although the performance for all S-parameters is measured, the S-parameter phase uncertainties are less important for verifying system performance. Therefore, the limit lines will not appear on the printouts.

- System Verification can NOT be run with a Multiport test set enabled. However, you can run a performance check as described in the Test Set User's Guide. See the N44xx User's Guide.

Equipment Used in the System Verification

For PNA models:
### E8356A, E8357A, E8358A
N3381A, N3382A, N3383A
E8801A, E8802A, E8803A
(Type-N test ports)

<table>
<thead>
<tr>
<th>Equipment Type</th>
<th>Type-N</th>
<th>3.5 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration kit</td>
<td>85032F</td>
<td>85033E</td>
</tr>
<tr>
<td>or</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECAL Module</td>
<td>85092B</td>
<td>85093B</td>
</tr>
<tr>
<td>Verification kit</td>
<td>85055A</td>
<td>85053B</td>
</tr>
<tr>
<td>RF Cable</td>
<td>N6314A</td>
<td></td>
</tr>
</tbody>
</table>

See Cable substitution

### E8362A/B
N5230A (20 GHz)
(3.5 mm test ports)
N5242A

<table>
<thead>
<tr>
<th>Equipment Type</th>
<th>3.5 mm</th>
<th>Type-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration kit</td>
<td>85052B/C/D</td>
<td>85054B/D</td>
</tr>
<tr>
<td>or</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECAL Module</td>
<td>N4691A</td>
<td>N4690A</td>
</tr>
<tr>
<td>Verification kit</td>
<td>85053B</td>
<td>85055A</td>
</tr>
<tr>
<td>RF Cable(s)</td>
<td>Single: 85131C/E</td>
<td>Single: 85132C/E</td>
</tr>
<tr>
<td></td>
<td>Pair: 85131D/F</td>
<td>Pair: 85132D/F</td>
</tr>
<tr>
<td>Adapters</td>
<td>None</td>
<td>Single: 85130C and one 7mm-to-Type-N from 85054B cal kit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pair: Two 7mm-to-Type-N from 85054B cal kit</td>
</tr>
</tbody>
</table>

### E8363A/B, E8364A/B
N5230A (40 or 50 GHz)
(2.4 mm test ports)
<table>
<thead>
<tr>
<th>Equipment Type</th>
<th>2.4 mm</th>
<th>3.5 mm</th>
<th>Type-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration kit or ECAL Module</td>
<td>85056A/D</td>
<td>85052B/C/D</td>
<td>85054B/D</td>
</tr>
<tr>
<td>Verification kit</td>
<td>N4693A</td>
<td>N4691A</td>
<td>N4690A</td>
</tr>
<tr>
<td>RF Cable(s)</td>
<td>85057B</td>
<td>85053B</td>
<td>85055A</td>
</tr>
<tr>
<td>Pair: 85133D/F</td>
<td>Pair: 85134D/F</td>
<td>Pair: 85135D/F</td>
<td></td>
</tr>
<tr>
<td>Adapters</td>
<td>None</td>
<td>Single: 85130F</td>
<td>Single: 85130E and two 7mm-to-Type-N from 85054B cal kit</td>
</tr>
<tr>
<td>Pair: None</td>
<td>Pair: Two 7mm-to-Type-N from 85054B cal kit</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**E8361A**

(1.85 mm test ports)

<table>
<thead>
<tr>
<th>Equipment Type</th>
<th>1.85 mm</th>
<th>2.4 mm</th>
<th>3.5 mm</th>
<th>Type-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calibration kit or ECAL Module</td>
<td>85058B</td>
<td>See 2.4 mm test port table above</td>
<td>See 2.4 mm test port table above</td>
<td></td>
</tr>
<tr>
<td>Verification kit</td>
<td>N4694A</td>
<td>See 2.4 mm test port table above</td>
<td>See 2.4 mm test port table above</td>
<td></td>
</tr>
<tr>
<td>RF Cable(s)</td>
<td>85058V</td>
<td>See 2.4 mm test port table above</td>
<td>See 2.4 mm test port table above</td>
<td></td>
</tr>
<tr>
<td>Single: N4697E</td>
<td>Single: N4697E</td>
<td>See 2.4 mm test port table above</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pair: N4697F</td>
<td>Pair: N4697F</td>
<td>See 2.4 mm test port table above</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adapters</td>
<td>None</td>
<td>See 2.4 mm test port table above</td>
<td>See 2.4 mm test port table above</td>
<td></td>
</tr>
</tbody>
</table>

**Cable Substitution**

The test port cables specified for the PNA have been characterized for connector repeatability, magnitude and phase stability with flexing, return loss, insertion loss, and aging rate. Since test port cable performance is a significant contributor to the system performance, cables of lower performance will increase the uncertainty of your measurement. It is highly recommended that the test port cables be regularly tested.

If the system verification is performed with a non-Agilent cable, ensure that the cable meets or exceeds the operation of the specified cable. Refer to the cable User's Guide for specifications.

**Cable Flex Factor**

Flex Factor determines how much of the cable phase uncertainty to include in determining the limit lines.

- Set to **0% (zero)** if the cables are held down in a fixture and are not allowed to move during the calibration and verification.
- Set to 100% if the cables are allowed to move a lot.

**Calibration Kit Substitution**

Non-Agilent calibration kits are not recommended nor supported.

**Precautions for Handling Airlines**

When you are using the airlines in the verification kit, observe the following practices to ensure good measurement techniques.

- Be very careful not to drop the airline’s center or outer conductor. Damage will result if these devices are dropped.
- Use proper Electro-Static Discharge (ESD) procedures.
- Clean your hands or wear gloves as skin oils will cause a change in electrical performance.
**Flow Diagram of Procedure**

The operational flow of the software is depicted by the flowchart shown below.
**Procedure for System Verification**

1. If you want printed test outputs, connect a printer to the analyzer. Let the analyzer warm up for at least 30 minutes.

2. Insert the PNA verification kit floppy disk into the analyzer disk drive.
3. On the **System** (or **Utility**) menu, point to **Service**, and click **System Verification**. The System Verification window similar to this will be displayed.

![System Verification Dialog](image)

4. In the **Calibration Kit** box, select the calibration kit or ECAl module that is being used. The corresponding verification kit to use appears in the **Verification Kit** box.

5. Under **Printer Output** click on any of the following options.
   - **Print Tabular Data**: Prints the verification data in tabular form which includes measured data and uncertainty limits. Refer to a tabular data example, later in this topic.
   - **Print Graphs**: Prints the verification data in graphical form. The graphic form includes the measured data trace, factory supplied data trace and uncertainty limits. Refer to a plot data example, later in this topic.
   - **File Tabular Data**: Writes the verification data in tabular form to a text file in the C:\Program Files\Agilent\Network Analyzer\Documents\ directory.
   - **File Graphs**: Saves a screen image in .PNG format in the C:\Program Files\Agilent\Network Analyzer\Documents\ directory.

**Note**: If you want printed output, it is assumed you have already installed the Windows driver for your particular printer, and have tested that you can print to the printer from the network analyzer. This software is designed to print to whichever printer is currently set as the Default printer (see Printers in the Windows Control Panel).

6. To modify the number of ports to be verified, to change the number of devices to measure, or to use a previously stored verification calibration, click on the **Configure** tab and make the desired selections.
   - For the system verification to be truly adequate, the software must measure all devices in the kit with a recent calibration applied. Removing and reattaching any test port cables or adapters invalidates all
previous calibrations.

7. Click Run.

8. Follow the instructions on the analyzer for performing the system verification, inserting the verification devices as prompted.

**Note for 3 Port PNA:**
The System Verification Procedure is repeated three times. The first time, **Ports 1 and 2** are measured as a pair; then **Ports 1 and 3** are measured; and lastly, **Ports 2 and 3** are measured.

**Note for 4 Port PNA:**
The System Verification Procedure is repeated two times. The first time, **Ports 1 and 2** are measured as a pair, then **Ports 3 and 4** are measured.

**Step-by-Step Process Description**

1. Depending upon the selected choice in the Calibration submenu of the Configure menu, the user is either prompted to recall a previous calibrated instrument state, or is guided through a full 2-port calibration using the selected calibration kit. For ECal, the ECal module is connected just once; a standby message is posted while the software is performing the calibration.

2. The user is prompted to connect the first verification device.

3. The software reads the factory measured data for that device and uncertainty values for that data (CITifiles) from the floppy disk supplied with the verification kit.

4. The software sends the factory measured data, calibration kit and instrument state information to the uncertainty calculator DLL, which generates uncertainty values specific to the PNA.

5. The analyzer first sets up for magnitude measurements of all four S-parameters, each parameter in a separate window (lin mag for S11 and S22, log mag for S21 and S12). Each of the factory measured S-parameters are fed to the appropriate window as a memory trace. Limit line offsets are calculated as the sum of the factory measured data uncertainties and PNA uncertainties reported by the DLL. Upper and lower limits are displayed (factory measured data + uncertainty sum, factory measured data - uncertainty sum). The PNA takes a sweep, limit test is turned on and PASS/FAIL status is reported in each of the four windows.

6. The user clicks a button when ready to view phase measurements. The four windows get updated for phase format, phase memory traces, phase limits and PASS/FAIL result.

7. If the limit test of any of the four S-parameters (magnitude or phase) indicates a FAIL status, the software suggests troubleshooting tips and asks if the user would like to repeat measurement of that device or proceed to the next device. If proceeding to the next device, the factory measured data and uncertainties for the next device are read from floppy, the uncertainty DLL gets called with this next set of factory measured data, and the four measurement windows get updated for magnitude measurement of the next device.

8. The software follows this same process until all selected devices have been measured, at which point a summary window is displayed containing the set of PASS/FAIL results for all four parameters of each device.

**If the System Fails the Verification Test**
IMPORTANT: Inspect all connections. Do not remove the cable from the analyzer test port. This will invalidate the calibration that you have done earlier.

1. Repeat this verification test. Make good connections with correct torque specifications for each verification device.

2. Disconnect, clean and reconnect the device that failed the verification test. Then measure the device again.

3. If the analyzer still fails the test, check the measurement calibration by viewing the error terms as described in "Front Panel Access to Error Terms" on page 4-7 of the Service Guide.

4. Refer to the graphic below, for additional troubleshooting steps.

**Verification Fails Flowchart**

```
  VERIFICATION FAILS
    ▲
     △
     △
     △
     △
    △
   △
  VERIFICATION DEVICE REMEASURED
    ▲
     △
     △
     △
     △
    △
   △
  PASS
    ▲
     △
     △
     △
     △
    △
   △
  PRINT OUT RESULTS AND CONTINUE TO MEASURE ALL THE VERIFICATION DEVICES
    ▲
     △
     △
     △
     △
    △
   △
  FAIL
    ▲
     △
     △
     △
     △
    △
   △
  FAILS AT THE SAME FREQUENCY WITH MORE THAN ONE DEVICE
    ▲
     △
     △
     △
     △
    △
   △
   YES
    ▲
     △
     △
     △
     △
    △
   △
  CAL KIT SUBSTITUTED
    ▲
     △
     △
     △
     △
    △
   △
  NO
    ▲
     △
     △
     △
     △
    △
   △
  SUBSTITUTE CAL KIT
    ▲
     △
     △
     △
     △
    △
   △
  PRINT OUT RESULTS AND CONTINUE TO MEASURE ALL THE VERIFICATION DEVICES
    ▲
     △
     △
     △
     △
    △
   △
  FAIL
    ▲
     △
     △
     △
     △
    △
   △
  SUBSTITUTE VERIFICATION KIT
    ▲
     △
     △
     △
     △
    △
   △
  NO
    ▲
     △
     △
     △
     △
    △
   △
  SUBSTITUTE VERIFICATION KIT
    ▲
     △
     △
     △
     △
    △
   △
  YES
    ▲
     △
     △
     △
     △
    △
   △
  GO TO THE TROUBLESHOOTING CHAPTER
    ▲
     △
     △
     △
     △
    △
   △
  SUBSTITUTION KIT
    ▲
     △
     △
     △
     △
    △
   △
  YES
    ▲
     △
     △
     △
     △
    △
   △
  SUBSTITUTE CABLES
    ▲
     △
     △
     △
     △
    △
   △
  NO
    ▲
     △
     △
     △
     △
    △
   △
  SUBSTITUTE CABLES
    ▲
     △
     △
     △
     △
    △
   △
  YES
    ▲
     △
     △
     △
     △
    △
   △
  SUBSTITUTE ADAPTERS
    ▲
     △
     △
     △
     △
    △
   △
  NO
    ▲
     △
     △
     △
     △
    △
   △
  SUBSTITUTE ADAPTERS
    ▲
     △
     △
     △
     △
    △
   △
  YES
    ▲
     △
     △
     △
     △
    △
   △
  GO TO THE TROUBLESHOOTING CHAPTER
```

**Interpreting the Verification Results**

The graphic below shows an example of typical verification results with *Tabular Data* selected in the *Printer Output* area of the *System Verification* window. A graphic later in this topic shows an example of typical verification results with *Measurement Plots* selected in the *Printer Output* area of the *System Verification* windows. These printouts include a comparison of the data from your measurement results with the traceable data and corresponding uncertainty specifications. Use these printouts to determine whether your measured data falls within the total uncertainty limits at all frequencies.

The tabular data consists of:
- Frequency of the data points (in MHz).
- Lower limit line as defined by the total system uncertainty specification.
- Results of the measurement.
- Upper limit line as defined by the total system uncertainty specification.
- Test status (PASS or FAIL) of that measurement point.

### Printout of Tabular Verification Results

The printed graphical results show:

- Upper limit points as defined by the total system uncertainty specifications.
- Lower limit points as defined by the total system uncertainty specifications.
- Data measured at the factory.
- Results of measurements.
- Measurement parameter names and formats (Lin Mag or Log Mag).
- Serial number of device (00810).
- Device being measured (Sys Ver 20 dB attenuator).

### Printout of Graphical Verification Results
Source Calibration

Source calibration adjusts the PNA source power for flatness across its full frequency range. This adjustment is for service only; not for measurement calibration.

Required Equipment

Preferred Power Meter: E4419B
Alternate Power Meters: E4419A or EPM-442A

Note: The power sensor depends on the PNA frequency range. Depending on the PNA model, two power sensors may be required to test the full frequency range. The PNA front panel connector type will determine the cable used and if an adapter is required with the power sensor(s).

<table>
<thead>
<tr>
<th>PNA Model</th>
<th>Power Sensor(s)</th>
<th>Cable</th>
</tr>
</thead>
<tbody>
<tr>
<td>E8356A</td>
<td>8482A</td>
<td>N6314A</td>
</tr>
<tr>
<td>E8357A</td>
<td>8482A</td>
<td>N6314A</td>
</tr>
<tr>
<td>E8358A</td>
<td>8482A and E4412A</td>
<td>N6314A</td>
</tr>
<tr>
<td>E8801A</td>
<td>8482A</td>
<td>N6314A</td>
</tr>
<tr>
<td>E8802A</td>
<td>8482A and E4412A</td>
<td>N6314A</td>
</tr>
<tr>
<td>E8803A</td>
<td>8482A and E4412A</td>
<td>N6314A</td>
</tr>
<tr>
<td>N3381A</td>
<td>8482A</td>
<td>N6314A</td>
</tr>
<tr>
<td>N3382A/N3383A</td>
<td>8482A and E4412A</td>
<td>N6314A</td>
</tr>
<tr>
<td>E8361A</td>
<td>** See E8361A procedure below</td>
<td>N4697-60001</td>
</tr>
<tr>
<td>E8362A / B</td>
<td>E4413A</td>
<td>85131E</td>
</tr>
<tr>
<td>E8363A/B</td>
<td>8487A (use with adapter 11900B) and V8486A (use with adapter V281B)</td>
<td>85133E</td>
</tr>
<tr>
<td>N5242A</td>
<td>E4413A</td>
<td>85131E</td>
</tr>
</tbody>
</table>

See PNA Accessories

Procedure

1. Refer to your power meter documentation to ensure the proper calibration factors for the power sensor have been entered into the table on the power meter.
2. Connect a GPIB cable between the power meter and network analyzer (use the System Controller GPIB port if applicable.)

3. Ensure the power sensor(s) are connected to the power meter.

4. In the analyzer **System** menu, point to **Service, Adjustments**, and click **Source Calibration**.

5. There are 3 different versions of the Source Calibration software; all are slightly different. All have a button that is labeled “Calibrate” or “Adjust”. This is the button that will begin the calibration process. Some versions will also have a button labeled “Verify” that will test the source calibration without making any changes. Other selections are for factory personnel use only.

6. Once begun, you must enter the power meter and sensor information. The software will verify the power meter and sensor. You are then prompted to connect the sensor(s) and cable as needed.

**Connecting sensors to the PNA**

![Diagram of connecting sensors to the PNA]

**Additional Information**

All ports are tested on all PNAs. Source calibration takes approximately 10 to 45 minutes to complete depending on the frequency range and model number of the PNA. The E8361 models may take up to 90 minutes.

**Troubleshooting**

In the event there is a problem with Source Calibration, please refer to the "Troubleshooting" chapter in the PNA Service Guide.
**E8361 Procedure**

Source and Receiver calibration requires the power meter to measure the source power over the full range of each of the PNA internal bands. Because the 8487A can not measure accurately above 50 GHz, it can only be used up to the next highest band switch frequency at 46.2 GHz. The V8486A sensor and V281B adapter are used from 46.2 GHz to 67 GHz.

For highest accuracy, the V8486A and V281B should be sent to Agilent for a custom calibration from 45 GHz to 70 GHz.

For the next highest accuracy level, the following procedure shows how to measure correction factors yourself from 46 to 50 GHz. This procedure assumes you have already loaded correction factors for both sensors into the power meter.

1. On your power meter, add 46 and 48 GHz to the Cal Factor Table.
2. Preset the PNA
3. Tune the PNA to 46 GHz (CW frequency)
4. Using the 8487A, measure power at port 1. Record this value.
5. Tune the PNA to 48 GHz (CW frequency)
6. Using the 8487A, measure power at port 1. Record this value.
7. Connect the V8486A, V281A, and 1.85 f-f adapter to the power meter.
8. Tune the PNA to 46 GHz (CW frequency)
9. Adjust the cal factor table 46 GHz setting until the power meter reading matches the power readings from step 4.
10. Tune the PNA to 48 GHz (CW frequency)
11. Adjust the cal factor table 48 GHz setting until the power meter reading matches the power readings from step 8.
Receiver Calibration

Receiver calibration adjusts the network analyzer receivers for a flat response across its full frequency range. This adjustment is for service only; not for measurement calibration.

Required Equipment

Preferred Power Meter: E4419B
Alternate Power Meters: E4419A or EPM-442A

Note: The power sensor depends on the PNA frequency range. Depending on the PNA model, two power sensors may be required to test the full frequency range. The PNA front panel connector type will determine the cable used and if an adapter is required with the power sensor(s).

See PNA Accessories

Procedure

1. Refer to your power meter documentation to ensure the proper calibration factors for the power sensor have been entered into a table on the power meter.

2. Connect a GPIB cable between the power meter and network analyzer.

3. Ensure the power sensor(s) are connected to the power meter.

4. In the analyzer System menu, point to Service, Adjustments, and click Receiver Calibration.

5. The software presents you with two choices:
   a. Click Inspect Flatness to observe flatness of receiver response versus frequency. Although there is no explicit specification for receiver flatness, Receiver Calibration should improve Transmission and Reflection Tracking error terms which are specified.
   b. Click Calibrate to begin the receiver calibration process. The software prompts you to connect the sensor(s), cable and adapter as needed (see the following graphics).

Connecting sensor(s) to the PNA
Connecting adapter and cable between sensor and PNA

Through connection using the specified cable
Additional Information

Receiver Calibration tests all PNA receivers, taking approximately 15 and 45 minutes. Length is dependent on frequency range and number of ports.

Troubleshooting

In the event there is a problem with Receiver Calibration, please refer to the "Troubleshooting" chapter in your PNA Service Guide.

Last Modified:

12-Jul-2007   Removed outdated table of supported power meters
The Receiver Display as a Troubleshooting Tool

The Receiver Display is a Troubleshooting Tool. It enables the analyzer to isolate faulty functional groups within its own Measurement System. Traces for each Receiver are Displayed in individual windows. Identifying discrepancies of the traces in these windows can help isolate the faulty assembly.

For a thorough description of Receiver Display and the troubleshooting steps see Chapter 3 of the PNA Service Guide. You can download the Service Guide for your PNA model from our website: http://na.tm.agilent.com/pna/

How to Start the Receiver Display

Using front-panel HARDKEY [softkey] buttons

| Using a mouse with PNA Menus |

For PNA-L and E836x models

1. Navigate using MENU/DIALOG

| Programming Commands |

1. Click System
2. then Service
3. then Utilities
4. then Receiver Display

For PNA-X and 'C' models

1. Press SYSTEM
2. then [Service]
3. then [Utilities]
4. then [Receiver Display]
IF Access User Interface Settings

The following IF receiver settings can be made through the UI (User Interface) on the E836x Option H11 and PNA-X Opt 020.

- **IF Gain Configuration**
- **IF Switch Configuration**
- **IF Filter Configuration**

### Other IF Access Topics

**See Also**

E836X Opt H11

- See the [H11 option rear panel connectors](#).
- See if your PNA can be **upgraded to the H11 option**.
- [See the H11 specs](#).
- [Opt H08 Pulsed Application](#).

PNA-X  Opt 020

- See the [PNA-X rear panel connectors](#).
- [See IF Path Config Block Diagram](#).

**IF Gain Configuration**

The IF Gain Configuration settings allow you to manually set the gain of the IF amplifiers.

Available on the [E836x Option H11 ONLY](#).
How to Make IF Access Settings

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Channel</td>
</tr>
<tr>
<td>2. then Advanced</td>
<td>2. then Advanced</td>
</tr>
<tr>
<td>3. then IF Gain Configuration</td>
<td>3. then IF Gain Configuration</td>
</tr>
</tbody>
</table>

IF Gain Configuration dialog box help

**Note:** This dialog is NOT available on the PNA-X models.

**Automatic** - check to allow the PNA to set the IF gain automatically. Clear to enable manual IF gain settings.

**Note:** Clearing the **Automatic** checkbox can result in inaccurate measurements.

**A, R1, R2, B Receivers**

**Auto Gain** - The PNA selects the best gain level to make pulsed measurements.

- **Low** - about 0 dB of gain
- **Med** - about 17 dB of gain
- **Hi** - about 34 dB of gain

**IF Switch Configuration**

The IF Switch setting allows you to select the input path for each IF receiver.

Available on the [E836x with Opt H11](#) and on the [PNA-X with Opt 020](#).
### How to Make IF Switch Settings

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For E836x models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Channel</td>
</tr>
<tr>
<td>2. then Advanced</td>
<td>2. then Advanced</td>
</tr>
<tr>
<td>3. then IF Switch Configuration</td>
<td>3. then IF Switch Configuration</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press TRACE/CHAN</td>
<td>1. Click Trace/Chan</td>
</tr>
<tr>
<td>2. then [Channel]</td>
<td>2. then Channel</td>
</tr>
<tr>
<td>3. then [Hardware Setup]</td>
<td>3. then Hardware Setup</td>
</tr>
<tr>
<td>4. then [IF Switch Configuration]</td>
<td>4. then IF Switch Configuration</td>
</tr>
</tbody>
</table>

**IF Switch Configuration dialog box help**

Select a receiver path to switch to External.

This is **Switch 1** on the [PNA-X IF Path Config Block diagram](#).

- **Unchecked (Normal)** - Allow the PNA to decide the input path to the PNA IF / Receiver path.
- **Checked (External)** - Always use the rear panel input to the selected PNA IF / Receiver path.

**IF Filter Configuration**

The following IF settings allow you to select the filter path for each IF receiver in the **PNA-X ONLY**.
## How to Make IF Access Settings

### Using front-panel HARDKEY [softkey] buttons

For **PNA-X and 'C' models**

1. None

### Using a mouse with PNA Menus

1. Click **Trace/Chan**
2. then **Channel**
3. then **Hardware Setup**
4. then **IF Filter Config** setting

### IF Filter Config dialog box help

Select a PNA-X IF Filter path to switch between Narrowband and Wideband filtering.

This is **Switch 2** on the **IF Path Config Block diagram**.

- **Unchecked** Wideband filtering
- **Checked** - Narrowband filtering

Last modified:

- 26-Jul-2007   Added IF Filter config
- 9/12/06   Added link to programming commands
External Millimeter Module Configuration

You can use external Millimeter Modules to extend the frequency coverage of your PNA. To use this feature your PNA must have the H11 Option. PNA-X models require Opt 020.

- PNA Limitations when using External Millimeter Modules
- How to Configure Millimeter Modules

Other IF Access Topics

PNA Limitations when using External Millimeter Modules

Power Settings  When using external Millimeter Modules, the PNA cannot control the power level into your DUT above 67 GHz. Because of this limitation, PNA power settings will not function correctly. Some of these settings are: Power level in standard or segment sweep, source and receiver power calibrations, and calibration interpolation. Your modules may have a manual power control.

Frequency Offset and FCA Measurements  Because of the various switch settings and configurations, Frequency Offset and FCA measurements are NOT supported when using external Millimeter Modules.

PNA-X models

- **CAUTION:** Connect a 10 dB attenuator to the N5260A LO input from the PNA-X LO Output. Otherwise, damage will occur to the N5260A test set.

- The PNA-X rear panel IF Inputs use 4 SMA connectors. Previous PNA models use BNC. Adapters may be required.

How to Configure Millimeter Modules

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>For <strong>N5230A</strong> and <strong>E836xA/B</strong> models</td>
<td>No programming comands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click <strong>System</strong></td>
</tr>
<tr>
<td></td>
<td>2. then <strong>Configure</strong></td>
</tr>
<tr>
<td></td>
<td>3. then <strong>Millimeter Module Config</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>For <strong>PNA-X</strong> and 'C' models</th>
<th>No programming comands</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Press <strong>SYSTEM</strong></td>
<td>1. Click <strong>Utility</strong></td>
</tr>
<tr>
<td>2. then <strong>[Configure]</strong></td>
<td>2. then <strong>System</strong></td>
</tr>
</tbody>
</table>
3. then **[Millimeter Module]**

4. then **Configure**

### Millimeter Head Setup dialog box help

#### Important Notes

- The External Sources will become UNSYNCHRONIZED after recalling an instrument state. When this occurs, launch this dialog box again and click **OK**. We strongly recommend using only one channel to make measurements using external sources.

- For measurements up to 110 GHz, the amount of phase noise is comparable when using either the internal PNA sources or external sources. For the best measurement accuracy at frequencies above 110 GHz, it is strongly recommended that external sources be used.

- Only Agilent PSG sources are supported.

- When **Use External Sources** is checked, the PNA trigger settings are automatically configured and must not be changed.

**Selected Module**  Displays the currently selected module. To select and configure a different module, click **Module Config**.

**Use External Sources**  Check to use external sources to provide an LO for the Millimeter Modules.

**Source Config**  Click to invoke the **External Source Configuration** dialog box. This dialog box is also used to configure external sources for the **FCA application**. Ignore references to FCA in the help topic.
Millimeter Module Configuration dialog box help

**Millimeter Module Configuration List**  Lists the standard PNA configuration and others that you have created.

**New**  Click to create a new Millimeter Module Configuration. Invokes the Name dialog box. Type a unique name for the new configuration using only alphanumeric characters and underscore, then click OK.

**Delete**  Remove a Millimeter Module Configuration.

**Restore Defaults**  Prompts, then REMOVES all configurations except for the standard PNA (and N5250A if installed).

**Save**  Saves the current settings to the PNA hard drive.

**Testport Frequency Range**  Set the Start and Stop frequencies of the selected configuration at the test ports. When Activated (click Activate Selected Config), this becomes the displayed Start and Stop frequency of the PNA.

**RF Freq Range Multiplier**  RF Frequency Range (displayed in grey fields) multiplied by this value = Testport Frequency Range.

Use the Multiplier values that are specified in your test head documentation.

**LO Freq Range Multiplier**  LO Frequency Range (displayed in grey field) multiplied by this value + 8.33 MHz = Testport Frequency Range.

**Note:** If the LO and RF frequency ranges are not within the operating range of the PNA, a warning message appears along with a red box around the invalid field. Click the appropriate Multiplier value up or down to correct the problem.

**Use Standard PNA operation when N5260A is NOT connected**  When Activate Selected Config is clicked, the PNA detects if a N5260A is connected. If one is NOT connected and:
- This box is checked, then the selected configuration is NOT activated, but uses the Standard PNA configuration.

- This box is cleared, then the selected configuration IS activated anyhow.

**Cancel**  Closes dialog box without saving changes.

**Close**  Prompts to save changes, then closes the dialog box.

**Activate Selected Config**  Saves the configuration, then closes and restarts the PNA application with the new configuration. To change to another configuration, including the standard PNA configuration, you must make this selection again.

---

**Last modified:**

- **April 9, 2007**  Updated for PNA-X
- **9/12/06**  Note - NOT compatible with Freq Offset
IF Path Configuration Settings

For Advanced Users: the following block diagram shows the IF Configuration capability of the PNA-X.

**Note:** The PNA Pulsed Application (Opt H08) provides an intuitive user interface to easily make average pulse, point-in-pulse, and pulse profiling measurements.

![Diagram of IF Path Configuration](image)

**Legend:**

Blue boxes with Ref #s are configurable elements. Click a blue box, or scroll down, to see how to make settings using SCPI and COM commands.

**Description:**

**Receiver / IF Path (top block)**

This section is duplicated for each of the receivers (A, B, C/R1, D/R2, R). See how to make settings remotely below.

- **Switch 1** Internal input is a test port or reference receiver input. External Input is through the rear-panel connectors (A, B, C/R1, D/R2, R)

- **Switch 2** Select Wideband IF or Narrowband IF, which includes the ability to pulse gates.

- **Programmable attenuator #3** Specify IF attenuation for the selected receiver.
Switch 4  IF Gates - 1 of 7 lines from pulse generator section to each of the receiver gates.

Switch 5  Select Auto, 9 MHz, or 16 MHz filters for all receivers.

ADC  Analog to digital converter for the selected receiver, with the ability to be pulsed by the P0 generator. ADC output goes to Digital Signal Processor (DSP).

**Pulse Generators (middle block)**

*See how to make settings remotely below.*

Switch 6  Internal or External inputs provide the REPETITION RATE (clock) for the pulse generator. From that rate, the generator creates five pulse outputs with unique delays and pulse widths. External input is on the Pulse I/O connector pin 7 (Synch IN).

P0  Hardwired to pulse the ADC for each receiver.

P1 thru P4  These four pulse generator outputs are hardwired to rear panel outputs on the Pulse I/O connector (pins 10 - 13). They are also routed to two switches (#4 and #7 on the above diagram) along with the following three lines:

- **Rear Panel**  External pulse generator input from Pulse I/O connector pin 8 (RFPulseModIn).
- **OFF**  Pulse is constantly in LOW state causing gate and source to be OFF.
- **ON**  Pulse is constantly in HIGH state causing gate and source to be ON.

Switch 7  Pulse Modulation - 1 of 7 lines to each of the sources. **Important:** When internally modulating the sources, source leveling must be set to Open-loop.

Rear-panel Pulse I/O connector (pins 10 - 13) hardwired.

Source1 and Source2 pulse modulators (#8 and #9 on the above diagram)

**DSP (bottom block)**

- Filters the ADC (digital) output from top block and outputs data to the PNA display.

- See SCPI and COM commands to control DSP settings.

See Also
Remote RF Path Configuration

Rear Panel Pulse I/O connector

Most of the following elements highlighted in BLUE in the above diagram, have settings that are made using SCPI or COM commands ONLY. These are the same commands that are used to make settings in the RF Path Configurator. In general the command specifies an element name and a setting.

See SCPI command
See COM object and example.

<table>
<thead>
<tr>
<th>Ref#</th>
<th>Element Name</th>
<th>Description</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;IFSWn&quot;</td>
<td>For 2-port PNA-X, n = A, B, R1, R2</td>
<td>• &quot;Internal&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For 4-port PNA-X, n = A, B, C, D, R (for R1 to R4)</td>
<td>• &quot;External&quot; [Rear Panel IF connectors], 4-port PNA-X use R for Ref 1 to 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example: &quot;IFSWB&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Requires Opt 020 external IF inputs on the rear panel</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>This switch CAN also be set from the PNA-X UI.</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>&quot;IFSigPathn&quot;</td>
<td>For 2-port PNA-X, n = A, B, R1, R2</td>
<td>• &quot;WBF&quot; Wide Band Filter Path (default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For 4-port PNA-X, n = A, B, C, D, R1, R2, R3, R4</td>
<td>• &quot;NBF&quot; Narrow Band Filter Path</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;IFSigPathAll&quot; makes setting for ALL receivers.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>This switch CAN also be set from the PNA-X UI.</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>&quot;NBFATNn&quot;</td>
<td>For 2-port PNA-X, n = A, B, R1, R2</td>
<td>0 to 31 in 1 dB steps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For 4-port PNA-X, n = A, B, C, D, R1, R2, R3, R4</td>
<td>For example: &quot;28&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example: &quot;NBFATNB&quot;</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>&quot;IFGaten&quot;</td>
<td>For 2-port PNA-X, n = A, B, R1, R2</td>
<td>• &quot;On&quot; Gate is always ON</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For 4-port PNA-X, n = A, B, C, D, R1, R2, R3, R4</td>
<td>• &quot;Off&quot; Gate is always OFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example: &quot;IFGateB&quot;</td>
<td>• &quot;RearPanel&quot; (use Pulse IO pins 1 to 5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• &quot;Pulse1&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• &quot;Pulse2&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• &quot;Pulse3&quot;</td>
</tr>
<tr>
<td></td>
<td>Setting</td>
<td>Options</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>IF Anti-Alias Filter</td>
<td>&quot;Auto&quot; PNA selects which filter to use based on other IF settings.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;9MHz&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;16MHz&quot;</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Pulse Trigger Input</td>
<td>&quot;Internal&quot; Internal Pulse In - pulse generators are triggered each period.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;External&quot; External Pulse Synch In - Pulse I/O pin 7 - An external trigger signal is required to trigger the pulse generators for each pulse.</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Pulse Mod Drive</td>
<td>&quot;On&quot; Pulse Mod drive is always ON, leaving &quot;SRC1</td>
<td>2 Out 1&quot; ON and not modulated. Default setting.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Off&quot; Pulse Mod drive is always OFF, leaving &quot;SRC1</td>
<td>2 Out 1&quot; OFF.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;RearPanel&quot; (use Pulse I/O pin 8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Pulse1&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Pulse2&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Pulse3&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Pulse4&quot;</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Source 1 Out 1 Pulse Mod Enable</td>
<td>&quot;Enable&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Disable&quot;</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Source 2 Out 1 Pulse Mod Enable</td>
<td>&quot;Enable&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Disable&quot;</td>
<td></td>
</tr>
</tbody>
</table>

**Important:** When Pulse 1-4 is selected to modulate the sources, source leveling must be set to Open-loop.

**Requires Opt 025 - Four Internal Pulse Generators**

**Requires Opt 021 - Source 1 Pulse Modulator**

**Requires Opt 022 - Source 2 Pulse Modulator**

Last Modified:

5-Sep-2007   New Image and minor edits

February 5, 2007   MX New topic
E5091 Testset Control

The E5091A is a popular Agilent Technologies 7-port / 9-port testset. Although the testset was originally designed to work with the ENA Network Analyzer, it also works well with the PNA. This topic describes how to control the testset from the PNA. For more information about the testset, refer to your E5091A documentation.

Overview

Connecting the E5091A

How to make E5091A Testset Control Settings

Calibrating with the E5091A

Other External Device Control Topics

Overview

When connected to the PNA, the E5091A Testset provides full 7-port or 9-port test capability. The E5091A can be configured to switch a different testset path for each PNA channel. When all channels have been configured, the entire measurement setup and calibration can be saved to a .cst or .csa file to be recalled later. In addition, the Channel Settings Table that is appended to a printed hardcopy of a measurement includes the E5901A Port Control settings.

Notes:

- The E5091A Testset has a maximum useful frequency of 11 GHz.
- The E5091A Testset Control can be automated using SCPI and COM commands.
- When enabled, a second status bar row appears which indicates the testset that is being controlled and the current switch state.
- Testset path switching occurs just before a channel is triggered. If a channel trigger state is Hold, switching for that channel does not occur.
- PNA sweep speed will be slightly slower when using the E5091A to switch measurement paths.

Connect and Configure the E5091A

The E5091A can be connected to any one of the PNA USB ports. When first installed, Windows will automatically launch the "Add New Hardware" wizard. Click Next to install the E5091A Testset.

Note: See the power handling limitations of the PNA USB ports.

Connect the PNA test ports to the E5091A test ports. Match PNA test port 1 to E5091A test port 1, and so forth.

Selecting ID for E5091A

The PNA can control up to two E5091A testsets. Set the Instrument ID bit switch to 1 or 2. The testsets will then be
identified automatically and referred to by the DIP switch setting on the E5091A rear-panel. Change the ID bit switch setting before connecting to the PNA USB.

Power ON
Immediately after power-on, all of the port connection indicator LEDs of the E5091A go ON. Then, after the PNA detects the E5091A, the four LEDs that indicate the connected test ports remain ON. If the PNA is not powered on or if the E5091A is not connected using a USB cable, all of the LEDs stay ON.

How to make E5091A Testset Control Settings

Using front-panel HARDKEY [softkey] buttons

<table>
<thead>
<tr>
<th>For N5230A and E836xA/B models</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Channel</td>
</tr>
<tr>
<td>2. then External Testset</td>
<td>2. then External Testset</td>
</tr>
<tr>
<td>3. then E5091A</td>
<td>3. then E5091A</td>
</tr>
</tbody>
</table>

For PNA-X and 'C' models

| 1. Press TRACE/CHAN | 1. Click Trace/Chan |
| 2. then [Channel] | 2. then Channel |
| 3. then [More] | 3. then More |
| 4. then [External Testset] | 4. then External Testset |
| 5. then [E5091A] | 5. then E5091A |
The title of the dialog shows the testset model and ID number of the active testset.

Select ID  ID of the testset to be configured. Up to two E5091A testsets can be controlled. Click to change testset ID. Learn how to set the testset ID.

Enable Test Set Control  When cleared, port switching and control line settings are disabled. This selection affects all channels using the selected testset.

Show Test Set Property  When checked, a second row on the status bar appears which indicates the testset that is being controlled and the current port control selection.  For example, the following image shows the status bar when controlling an E5091A testset and a Z5623A K64 testset:

A. Configured channel
B. Port Control settings for E5091A
C. Port Control settings for Z5623A K64
D. Testset Label. E5091A control does not use this field. It is shared between Interface Control and External Testset Control. The two labels are separated by /.

Control of the second status bar is completely separate from the first status bar, which is controlled from the View, Status Bar menu.

Port Control  Controls mapping of Physical ports to Logical ports.

- Physical ports are the port numbers that are labeled on the test set front panel. (see N44xx test sets)
- Logical ports are the port numbers that are referred to by most of the PNA application prompts and dialog boxes.
Port Mapping Notes

- Port Control and Control Line settings effect the channel of the active (selected) measurement. These settings will occur as the channel is being measured.
- Correction is NOT turned OFF when port mappings are changed. However, the calibration is NO LONGER VALID!

Control Lines  Specifies the values of individual control lines. These general purpose control lines on the test set front-panel can be used in your test setup. Each button toggles the control line HIGH and LOW. When first opened, the selections reflect the current control lines. See your test set documentation for more information about the control lines.

OK  When clicked, the changes to the dialog box are implemented and the port selections and control values are immediately sent to the specified test sets. The Port Control and Control line settings are stored with other channel data and used when those channels are swept.

Cancel  (or Escape) Changes to the dialog are not implemented and revert to the settings before the dialog box was opened.

Calibrating with the E5091A

The following are a few changes in the way you calibrate the PNA with the E5091A connected:

1. Create the measurements for the channel and configure the Port Control (switching) on the E5091A Testset Control dialog box. Enable Show Testset Property.
2. To calibrate, start the Calibration wizard and select a Calibration method (ECAL, SmartCal, Unguided).
3. Select the DUT connectors that are used at the E5091A measurement reference plane.
4. When prompted to connect a standard to a PNA port, instead connect the standard to the E5091A port as indicated on the Testset status bar. For example, if the status bar indicates 1 A, instead of connecting a standard to port 1, connect it to port A reference plane of the E5091A.

Last Modified:

18-Jun-2007    MX added UI
External Multiport Test Set Control

- **Supported Test Sets**
  - Option 551
  - Option 550
  - E5091A (separate topic)

- **Procedure**
  1. Connect Test Set
  2. Restart as Multiport
  3. Optional External Test Set Control Settings

- **External Test Set Control and other PNA Functions**

### Other External Device Control Topics

#### Supported Test Sets
The list of test sets that provide integrated solutions with the PNA is constantly growing. For a current list of supported multiport test sets, see [www.agilent.com/find/multiport](http://www.agilent.com/find/multiport)

**Option 551**

- **With** Option 551 enabled on your PNA, any supported multiport test set can be controlled directly from the PNA to make fully integrated measurements at ALL of the available test ports. To understand what test ports are available to source and receive, see the test set documentation.

- **Without** Option 551, basic operation depends on the number of PNA test ports.
  
  - For a 2-port PNA, configure two available test ports.
  
  - For a 4-port PNA, configure four available test ports.

**Option 550**

- **With** Option 550 enabled on your PNA, the N44xx test sets, and other supported test sets, can be controlled directly from the PNA to make fully integrated 4-port balanced measurements.

- **Without** Option 550, basic operation allows you to configure any two of the test ports to make standard S-parameter or receiver measurements. A different pair of ports can be configured for each channel.
### N44xx Test Set Model

<table>
<thead>
<tr>
<th>N4419B</th>
<th>E8362B and N5230A Opt 225</th>
<th>10 MHz to 20 GHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>N4420B</td>
<td>E8363B and N5230A Opt 425</td>
<td>10 MHz to 40 GHz</td>
</tr>
<tr>
<td>N4421B</td>
<td>E8364B and N5230A Opt 525</td>
<td>10 MHz to 50 GHz</td>
</tr>
<tr>
<td>N4421B Opt H67</td>
<td>E8361A</td>
<td>10 MHz to 67 GHz</td>
</tr>
</tbody>
</table>

**Note:** By default, the system logical test ports are mapped as follows:

- **Port 1** - PNA port 1
- **Port 2** - Test Set port 2
- **Port 3** - PNA port 2
- **Port 4** - Test Set port 4
The ports can be **remapped** using the Port Control Setting.

**Block Diagram of a 2-port PNA and N44xx Test Set**

Click one of the S-parameters to see switch and path changes. Because the test set does NOT contain receivers, **measurement speed** and **calibration** can be affected.

---

**Procedure - How to enable full Multiport Capability**

1. **Enable Option 550 or 551.**
2. Connect the test set to the PNA using the documentation that was shipped with the test set.
3. **Restart as Multiport PNA**
4. Make optional External Test Set Control Settings

Connect and Configure the Test Set

Connect the test set to the PNA using the test set documentation. Most test set documentation can be found at www.Agilent.com

Test Set I/O-controlled test sets

Test sets that are controlled using the Test Set I/O connector, have NO return communication capability. The PNA sends commands out the rear panel connector. It is assumed that the test set is responding appropriately.

GPIB-controlled test sets

Connect the test set to the GPIB using one of the following methods:

- If the PNA will NOT be controlled by a remote computer using GPIB, then the test set can be connected directly to the PNA GPIB port. The PNA is automatically switched to System Controller mode.
- If the PNA WILL be controlled by a remote computer using GPIB, then learn how to connect the test set.

Restart as Multiport PNA

How to Enable Multiport capability

Note: If Option 550 or Option 551 has not been enabled, the following Multiport Capability menu selection will NOT be available.

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For N5230A and E836xA/B models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click System</td>
</tr>
<tr>
<td></td>
<td>2. then Configure</td>
</tr>
<tr>
<td></td>
<td>3. then Multiport Capability</td>
</tr>
<tr>
<td><strong>For PNA-X and 'C' models</strong></td>
<td></td>
</tr>
<tr>
<td>1. Press SYSTEM</td>
<td>1. Click Utility</td>
</tr>
<tr>
<td>2. then [Configure]</td>
<td>2. then System</td>
</tr>
<tr>
<td>3. then [Multiport Capability]</td>
<td>3. then Configure</td>
</tr>
<tr>
<td></td>
<td>4. then Multiport Capability</td>
</tr>
</tbody>
</table>
After the test set is connected and PNA Option 550 or 551 is enabled, the following settings are used to enable Multiport operation.

**Test Set** Select the test set file to load. Only the files that are appropriate for use on that PNA model are displayed.

**To Add a new Test Set file:**

1. On the PNA, click **File**, then **Exit** to quit the PNA application.
3. Save it to the PNA C:\program files\agilent\network analyzer\testsets
4. Start the PNA application.
5. Click **System, Configure, Multiport, Testset**. The new test set should now be visible.

**Address** Enter the test set address.

- **Restart as a standalone PNA.** The PNA shuts down and restarts as a standard PNA. If the test set is left connected to the PNA, switch the test set OFF, then back ON to ensure that the test set routes signals to ports 1 and 2 of the PNA. In this condition, there is more loss in the test paths than without a test set connected. If the power switch is OFF, there is SIGNIFICANTLY more loss in the test paths.

- **Restart as a multiport PNA with this testset.** The PNA shuts down and restarts as a multiport PNA with the selected test set.

Click **OK**. The PNA shuts down and restarts in the selected configuration.

To learn how to change port mapping, see **Port Control**.

**Problems**

If the PNA cannot find the test set, the following error is displayed on the PNA:
GPIB ERROR Address xx cannot open VISA session.

To correct the problem, verify the following:

- The test set is connected to the PNA using one of the methods described above.
- The correct test set address is set.
- The test set is turned ON.

Important: After the problem has been fixed:

1. On the External Test Set Control dialog, click Enable Test Set Control.
2. Restart Triggering - click Sweep, Trigger, Continuous.
3. The PNA again tries to find the test set.

External Test Set Control Settings

The following External Test Set Control Settings are used to configure Multiport test sets. For the N44xx test sets, the only setting that is necessary is port control.

How to access the External Test Set Control Settings

<table>
<thead>
<tr>
<th>Using front-panel HARDKEY [softkey] buttons</th>
<th>Using a mouse with PNA Menus</th>
</tr>
</thead>
<tbody>
<tr>
<td>For N5230A and E836xA/B models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Navigate using MENU/DIALOG</td>
<td>1. Click Channel</td>
</tr>
<tr>
<td></td>
<td>2. then External Testset</td>
</tr>
<tr>
<td></td>
<td>3. then Other</td>
</tr>
<tr>
<td>For PNA-X and 'C' models</td>
<td>Programming Commands</td>
</tr>
<tr>
<td>1. Press TRACE/CHAN</td>
<td>1. Click Trace/Chan</td>
</tr>
<tr>
<td>2. then [Channel]</td>
<td>2. then Channel</td>
</tr>
<tr>
<td>3. then [More]</td>
<td>3. then More</td>
</tr>
<tr>
<td>4. then [External Testset]</td>
<td>4. then External Testset</td>
</tr>
<tr>
<td>5. then [Other]</td>
<td>5. then Other</td>
</tr>
</tbody>
</table>
External Test Set Control dialog box help

**Important Notes:**

- With Options 500 and 551, **first** use the *Multiport Restart* dialog to **Restart as Multiport PNA with this test set**. The test set file is loaded and the test set is enabled automatically.

- When using GPIB to control an external test set, the PNA is automatically put in *System Controller mode*. It can NOT also be in talker-listener mode. To have the PNA control a GPIB test set AND be controlled by a remote PC, use a *USB to GPIB adapter* to control the external test set. This does NOT apply for PNA models with a **1.1 GHz CPU board**, which has both a GPIB controller port and a talker/listener port.

- See also *External Test Set Control and other PNA Functions*

**Select ID**

- For N44xx test sets: the GPIB address

- Multiport test sets: either GPIB address or 0 for Test Set I/O controlled test sets.

**Enable Test Set Control**  When cleared, port switching and control line settings are disabled. This selection affects all channels using the selected test set. When checked, the 'Show Test Set Properties' checkbox is also checked automatically.

**Load Test Set File**  For operating **without** Options 500 or 551.

If your Test Set is not visible, see *Add a new Test Set*.

The selected test set file is loaded.

1. Navigate to the folder: C:/Program Files/Agilent/Network Analyzer/testsets/

2. Select a test set control file.
The title of the dialog shows the model of the test set file that is currently loaded.

See a list of supported test sets.

**Show Test Set Properties**  This box becomes checked by default when the Enable Test Set Control is checked. When checked, a second row on the status bar appears which indicates the test set that is being controlled and the current port control selection. For example, the following image shows the status bar when controlling an **E5091A** test set and a **Z5623A K64** test set:

![Status Bar Example](image)

A. Configured channel

B. Port Control settings for E5091A

C. Port Control settings for Z5623A K64

D. Test Set Label. This field is shared between Interface Control and External Test Set Control. The two labels are separated by `/`.

Control of the second status bar is completely separate from the first status bar, which is controlled from the View, Status Bar menu.

**Test Set Label**  NOT available with options 550 and 551. Add text to appear on the second status bar when Show Test Set Properties is checked. See image above.

**Port Control**  Controls mapping of Physical ports to Logical ports. *(Refer to image of dialog box above.)*

- Physical ports are the port numbers that are labeled on the test set front panel.
- Logical ports are the port numbers that are referred to by most of the PNA application prompts and dialog boxes.

### Port Mapping Notes

- Port Control and Control Line settings effect the channel of the active (selected) measurement. These settings occur as the channel is being measured.
- Correction is turned OFF when port mappings are changed.
- After the physical ports are mapped to logical ports, all PNA references to port numbers refer to LOGICAL port numbers. The only exception to this is during calibrations.

**Control Lines**  Specifies the values of individual control lines. These general purpose control lines on the test set front-panel can be used in your test setup. Each button toggles the control line HIGH and LOW. When first opened, the selections reflect the current control lines. See your test set documentation for more information about the control lines.

**OK**  When clicked, the changes to the dialog box are implemented and the port selections and control values are immediately sent to the specified test set. The Port Control and Control line settings are stored with other channel data and used when those channels are swept.

**Cancel**  (or Escape) Changes to the dialog are not implemented and revert to the settings before the dialog box was opened.
External Test Sets and other PNA Functions

The following features may work differently with a test set connected to the PNA.

Remote Commands

See [SCPI](https://www.ni.com) and [COM](https://www.ni.com) commands for controlling an External Test Set.

Use *OPT? (SCPI) or NumberofPorts (COM) to query the number of ports for a PNA/External Test set.

Interface Control

When both Interface Control and External Test Set Control are configured, the commands on the Interface Control Before Sweep Start tab are sent out before any External Test Set Control commands are executed on that channel. Similarly, commands on the After Sweep End tab are sent after Test Set Control commands.

Calibration

With an External Test Set connected, calibration is performed exactly like a PNA with the following exceptions:

- Correction is turned OFF when port mappings are modified. This also applies to Source Power Cal.

- A TRL Cal, QSOLT, or Unknown Thru can be performed on these systems ONLY when other portions of the calibration fully characterize all ports using SOLT with Defined Thru or Flush Thru. For example, when calibrating all four ports of a PNA with N44xx test set, perform a SOLT between ports 1 and 2, and also between ports 3 and 4 using a Defined Thru or Flush Thru. Then Unknown Thru could be used between any combination of the remaining ports.

A Delta Match Calibration is NOT POSSIBLE on External Test Set systems, although revision A.06.04 will allow it.

- With an External Test Set connected, you may be required to perform more than 3 THRU connections.

As with ALL PNA calibrations, when error correction is ON, both forward and reverse sweeps are required for EACH port pair that is corrected, even if only one reflection measurement is displayed. For example, any displayed measurement with full 4-port calibration ON will require 12 measurement sweeps. Learn more.

Source Power Cal

Source power calibration involves adjusting the source so that the power at an output port is flat across a frequency range. Because of additional loss through some of the test set paths, it may NOT be possible to obtain corrected output power because of limitations on the source signal.

During a Source Power Cal, you are prompted when and where to connect the power sensor. When one of the supported test sets are connected, the prompt refers to the PHYSICAL port number, NOT the LOGICAL port number. To help with translating physical to logical port mappings, enable Show Test Set Properties.

Measurements with Shared Receivers

External test sets do not contain receivers. The PNA receivers are always used to measure signals at the external test set ports. Therefore, when a channel contains two measurements that share a PNA test port receiver, additional sweeps are necessary.
For example, to make S34 and S44 measurements in the same channel with correction OFF:

- On a 4-port PNA, only ONE sweep is required using the C (port 3), D (port 4), and R (reference for All receivers).
- On a N44xx system, TWO sweeps are required since both measurements use the B and R2 receivers. See interactive block diagram above.

Create Ratioed and Unratioed Measurements

When using an external test set, it IS possible to create a Ratioed measurement using two logical receivers that share the same physical PNA receiver. However, this measurement data is NOT valid. Invalid measurement traces show all data at -200dB. Learn about Logical Receiver Notation

Logical Port References

When an external test set is enabled, all references to PNA port numbers and test set port numbers (except during calibrations) refer to LOGICAL port numbers. Logical ports can be remapped using the Port Control settings. During a calibration, you are prompted to connect standards to physical port numbers.

Balanced Port Mapping

"Logical Ports" is a term that is used with both External Test Sets and balanced measurements. While the concept is the same, they refer to different scenarios. The two can be easily confused when making Balanced measurements with an External Test Set connected. The important principle to remember is the order in which the logical ports for each are mapped:

1. In the External Test Set - Port Control settings dialog, the physical PNA ports and test set ports are mapped to logical ports as noted above.
2. In the Balanced Topology Dialog, the new (step 1) logical ports are mapped again to become Balanced logical ports.

Preset

Instrument Preset will reset Port Control settings to defaults and remove the test set label. All other settings remain. To maintain port control settings and the test set label, create a User Preset.
**Instrument State Save and Recall**

*Instrument State files* include Test Set model, Enable and Status bar settings, and Port mappings and DUT control values for each channel.

If an Instrument State recall requires that a test set configuration file be loaded, recall time may be significant. For example, this would occur if a 2-port PNA with attached test set is configured as a 2-port PNA and then recalls a state file which requires 4-port operation.

**Recall Cal Sets**

If a Cal Set is saved while an external test set is enabled, when the Cal Set is recalled, then the external test set must be enabled or an error message is displayed.

**Copy Channel**

*Copy Channel* copies all relevant test set data from the source channel to the target channel.

**Applications**

No PNA applications are supported with External Test Set Control. These include FCA (opt 083), SMC (opt 082), GCA (opt 086), NFA (opt 029), Pulsed (opt H08).

**Print**

Port mapping information appears on the *Channel Settings Table* when printing.

**Save sNp Files**

To save sNp data with an external test set enabled, click File, *Save As*, then select *Snp File(*.s*p)*, then complete the *Choose ports dialog*.

---

Last modified:

- 15-Jan-2008  No App support
- 19-Nov-2007  Note to Add TS files
- 9-Nov-2007  Fixed S13 image and added delta match notes
- 17-Sep-2007  Added note for PNA-X test set files
- 15-Jan-2007  MX Added UI
- 10/16/06  Added clarification to opts.
- 9/18/6  MQ Many edits for Opt 551
- 9/12/06  Added link to programming commands
Because the E8356A, E8357A, E8358A network analyzers are no longer produced, the technical specifications are stored only on the Internet. To view or print the .pdf version of the specifications, visit our web site at http://www.agilent.com/find/pna, and search for "E8356A Specifications"

The uncertainty curves contained in this document apply only to the setup conditions listed. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the curves for your PNA setup. View the equations used to generate the uncertainty curves.
Because the E8801A, E8802, E8803A network analyzers are no longer produced, the technical specifications are stored only on the Internet. To view or print the .pdf version of the specifications, visit our web site at http://www.agilent.com/find/pna, and search for "E8801A Specifications"

The uncertainty curves contained in this document apply only to the setup conditions listed. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the curves for your PNA setup. View the equations used to generate the uncertainty curves.
Technical Specifications for the N3381A, N3382A, N3383A
(Rev. 2005-09-26)

Because the N3381A, N3382A, N3383A network analyzers are no longer produced, the technical specifications are stored only on the Internet. To view or print the .pdf version of the specifications, visit our web site at http://www.agilent.com/find/pna, and search for "N3381A Specifications"

The uncertainty curves contained in this document apply only to the setup conditions listed. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the curves for your PNA setup. View the equations used to generate the uncertainty curves.
This is a complete list of the E8361A network analyzer technical specifications.

- To optimize viewing of uncertainty curves, click the Maximize button.

- To view or print the .pdf version of the specifications, visit our web site at http://www.agilent.com/find/pna, and search for “E8361A Specifications”

- The uncertainty curves contained in this document apply only to the setup conditions listed. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the curves for your PNA setup. View the equations used to generate the uncertainty curves.

### Definitions

**Corrected System Performance**

- **System Dynamic Range**
- **Receiver Dynamic Range**
- **1.85mm Connectors**
- **2.4mm Connectors**

**Uncorrected System Performance**

**Test Port Output**

**Test Port Input**

**Dynamic Accuracy**

**Group Delay**

**General Information**

**Measurement Throughput Summary**

**Front-panel Jumper Specs** (Option 014 only)

**Test Set Block Diagrams**

**Test Set with Option 014 Block Diagrams**

See [Specs for other PNA models](#)
Definitions

All specifications and characteristics apply over a 25 °C ±5 °C range (unless otherwise stated) and 90 minutes after the instrument has been turned on.

**Specification (spec.):** Warranted performance. Specifications include guardbands to account for the expected statistical performance distribution, measurement uncertainties, and changes in performance due to environmental conditions.

**Characteristic (char.):** A performance parameter that the product is expected to meet before it leaves the factory, but that is not verified in the field and is not covered by the product warranty. A characteristic includes the same guardbands as a specification.

**Typical (typ.):** Expected performance of an average unit which does not include guardbands. It is not covered by the product warranty.

**Nominal (nom.):** A general, descriptive term that does not imply a level of performance. It is not covered by the product warranty.

**Calibration:** The process of measuring known standards to characterize a network analyzer's systematic (repeatable) errors.

**Corrected (residual):** Indicates performance after error correction (calibration). It is determined by the quality of calibration standards and how well "known" they are, plus system repeatability, stability, and noise.

**Uncorrected (raw):** Indicates instrument performance without error correction. The uncorrected performance affects the stability of a calibration.

**Standard:** When referring to the analyzer, this includes no options unless noted otherwise.

Corrected System Performance

The specifications in this section apply for measurements made with the E8361A analyzer with the following conditions:

- 10 Hz IF bandwidth
- No averaging applied to data

- **System Dynamic Range**
- **Receiver Dynamic Range**
- **1.85mm Connectors**
- **2.4mm Connectors**

| Table 1. System Dynamic Range^A  |
|------------------|----------------|------------------|-----------------|
| Description      | Specification (dB) at Test Port^B | Typical (dB) at Direct Receiver Access Input^C | Supplemental Information |
| Dynamic Range (in a 10 Hz BW) | 2770 | 2770 | |

2770
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Standard Configuration (E8361A - Standard)</th>
<th>Configurable Test Set (E8361A - Option 014 or Option 014 and 080)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz(^d)</td>
<td>61</td>
<td>61</td>
</tr>
<tr>
<td>45 MHz to 500 MHz(^e)</td>
<td>87</td>
<td>99</td>
</tr>
<tr>
<td>500 MHz to 750 MHz</td>
<td>112</td>
<td>102</td>
</tr>
<tr>
<td>750 MHz to 2 GHz</td>
<td>111</td>
<td>125.5</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>111</td>
<td>125</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>114</td>
<td>125.5</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>103</td>
<td>101</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>104</td>
<td>128</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>96</td>
<td>117.5</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>100</td>
<td>117.5</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>97</td>
<td>117.5</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>94</td>
<td>117.5</td>
</tr>
<tr>
<td>67 GHz to 70 GHz(^d)</td>
<td>94</td>
<td>117.5</td>
</tr>
<tr>
<td>70 GHz to 80 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80 GHz to 90 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>90 GHz to 100 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100 GHz to 110 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>110 GHz to 120 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>120 GHz to 130 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>130 GHz to 140 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>140 GHz to 150 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>150 GHz to 160 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>160 GHz to 170 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>170 GHz to 180 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>180 GHz to 190 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>190 GHz to 200 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200 GHz to 220 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>220 GHz to 240 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>240 GHz to 260 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>260 GHz to 280 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>280 GHz to 300 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 GHz to 320 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>320 GHz to 340 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>340 GHz to 360 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>360 GHz to 380 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>380 GHz to 400 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>400 GHz to 420 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>420 GHz to 440 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>440 GHz to 460 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>460 GHz to 480 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>480 GHz to 500 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>500 GHz to 520 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>520 GHz to 540 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>540 GHz to 560 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>560 GHz to 580 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>580 GHz to 600 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>600 GHz to 620 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>620 GHz to 640 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>640 GHz to 660 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>660 GHz to 680 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>680 GHz to 700 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>700 GHz to 720 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>720 GHz to 740 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>740 GHz to 760 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>760 GHz to 780 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>780 GHz to 800 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>800 GHz to 820 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>820 GHz to 840 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>840 GHz to 860 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>860 GHz to 880 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>880 GHz to 900 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>900 GHz to 920 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>920 GHz to 940 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>940 GHz to 960 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>960 GHz to 980 GHz(^d)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>980 GHz to 1000 GHz(^d)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(^d\) Standard Configuration

\(^e\) Configurable Test Set

Option 016 degrades performance by 2 dB.
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Lower Limit</th>
<th>Upper Limit</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>60 GHz to 67 GHz</td>
<td>90</td>
<td>101</td>
<td>Option 016 degrades performance by 3 dB.</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>90</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

**Configurable Test Set & Extended Power Range**
*(E8361A - Option 014 & UNL or Option 014 & UNL & 080)*

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Lower Limit</th>
<th>Upper Limit</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>61</td>
<td>99</td>
<td></td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>87</td>
<td>102</td>
<td></td>
</tr>
<tr>
<td>500 MHz to 750 MHz</td>
<td>112</td>
<td>125.5</td>
<td></td>
</tr>
<tr>
<td>750 MHz to 2 GHz</td>
<td>111</td>
<td>124</td>
<td></td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>111</td>
<td>124</td>
<td></td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>112</td>
<td>125</td>
<td></td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>101</td>
<td>114.5</td>
<td>Option 016 degrades performance by 2 dB.</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>99</td>
<td>112</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>92</td>
<td>105</td>
<td></td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>94</td>
<td>106.5</td>
<td></td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>91</td>
<td>103</td>
<td></td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>84</td>
<td>95</td>
<td>Option 016 degrades performance by 3 dB.</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>84</td>
<td>94</td>
<td></td>
</tr>
</tbody>
</table>

* Typical performance.

---

The system dynamic range is calculated as the difference between the noise floor and the source maximum output power. System Dynamic Range is a specification when the source is set to Port 1, and a characteristic when the source is set to Port 2. The effective dynamic range must take measurement uncertainties and interfering signals into account as well as the insertion loss resulting from a thru cable connected between Port 1 and Port 2.

b The test port system dynamic range is calculated as the difference between the test port noise floor and the source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account as well as the insertion loss resulting from a thru cable connected between Port 1 and Port 2.

c The direct receiver access input system dynamic range is calculated as the difference between the receiver access input noise floor and the source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account. This set-up should only be used when the receiver input will never exceed its damage level. When the analyzer is in segment sweep mode, the analyzer can have predefined frequency segments which will output a higher power level when the extended dynamic range is required (i.e. devices with high insertion loss), and reduced power when receiver damage may occur (i.e. devices with low insertion loss). The extended range is only available in one-path transmission measurements.

d May be limited to 100 dB at particular frequencies below 500 MHz due to spurious receiver residuals. Methods are available to regain the full dynamic range.

**Note:** This E8361A document does NOT provide technical specifications for Receiver Dynamic Range.

**Note:** This E8361A document provides technical specifications for the following calibration kits and Ecal modules only: 85056A, 85058B, N4693A, N4694A. Please download our free Uncertainty Calculator from [http://www.agilent.com/find/na_calculator](http://www.agilent.com/find/na_calculator) to generate the curves for your PNA setup.
Table 10. Uncorrected System Performance\textsuperscript{a}
Specifications apply over environmental temperature of 23° ±3 °C, with < 1 °C deviation from the calibration temperature

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directivity</strong></td>
<td></td>
<td><strong>typical:</strong></td>
</tr>
<tr>
<td>10 MHz to 45 MHz\textsuperscript{b}</td>
<td>--</td>
<td>22 dB</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>24 dB</td>
<td>27 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>20 dB</td>
<td>24 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>16 dB</td>
<td>20 dB</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>14 dB</td>
<td>17 dB</td>
</tr>
<tr>
<td>30 GHz to 50 GHz</td>
<td>13 dB</td>
<td>17 dB</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>13 dB</td>
<td>17 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>10 dB</td>
<td>18 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz\textsuperscript{b}</td>
<td>--</td>
<td>14 dB</td>
</tr>
<tr>
<td><strong>Source Match - Standard</strong></td>
<td></td>
<td><strong>typical:</strong></td>
</tr>
<tr>
<td>10 MHz to 45 MHz\textsuperscript{b}</td>
<td>--</td>
<td>7 dB</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>18 dB</td>
<td>23 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>14 dB</td>
<td>18 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>12 dB</td>
<td>15 dB</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>8 dB</td>
<td>11.5 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>7.5 dB</td>
<td>10 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>8 dB</td>
<td>11 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>7 dB</td>
<td>10 dB</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>6 DB</td>
<td>8.5 dB</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Source Match - Option 014</td>
<td>Source Match - UNL &amp; 014</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>5.5 dB</td>
<td>6 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>7.5 dB</td>
</tr>
<tr>
<td><strong>Source Match - Option 014</strong></td>
<td><strong>typical:</strong></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>7 dB</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>17 dB</td>
<td>21 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>12 dB</td>
<td>17 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>11 dB</td>
<td>14 dB</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>10 dB</td>
<td>13 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>8.5 dB</td>
<td>11 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>8.5 dB</td>
<td>11 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>8.5 dB</td>
<td>11.5 dB</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>6.5 dB</td>
<td>9 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>6 dB</td>
<td>8.5 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>8.5 dB</td>
</tr>
<tr>
<td><strong>Source Match - UNL &amp; 014</strong></td>
<td><strong>typical:</strong></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>5 dB</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>15 dB</td>
<td>20 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>9 dB</td>
<td>13 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>7.5 dB</td>
<td>10.5 dB</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>8.5 dB</td>
<td>11 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>8 dB</td>
<td>11 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>8.5 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>8 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Load Match - Standard</td>
<td>Load Match - Option 014</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>7 dB</td>
<td>5.5 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>6 dB</td>
<td>8.5 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>6.5 dB</td>
</tr>
<tr>
<td>70 GHz to 72 GHz</td>
<td>--</td>
<td>5 dB</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>5.5 dB</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>9 dB</td>
<td>8.5 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>9 dB</td>
<td>6 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>8.5 dB</td>
<td>6 db</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>7 dB</td>
<td>6.5 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>6 dB</td>
<td>5.5 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>6.5 dB</td>
<td>5.5 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>6.5 dB</td>
<td>7.5 dB</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>5.5 dB</td>
<td>8 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>5.5 dB</td>
<td>6 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>5 dB</td>
</tr>
</tbody>
</table>

**Load Match - Standard**

**typical:**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Load Match - Standard</th>
<th>Load Match - Option 014</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>9 dB</td>
<td>8.5 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>8 dB</td>
<td>6 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>8 dB</td>
<td>6.5 dB</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>7.5 dB</td>
<td>5.5 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>7 dB</td>
<td>4.5 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>7.5 dB</td>
<td>9.5 dB</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Reflection Tracking</td>
<td>Load Match - Option UNL &amp; 014</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>7.5 dB</td>
<td>typical</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>6 dB</td>
<td>10 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>6 dB</td>
<td>8.5 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>5 dB</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>6 dB</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>8.5 dB</td>
<td>10 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>7 dB</td>
<td>9 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>6 dB</td>
<td>9 dB</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>7.5 dB</td>
<td>11 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>8 dB</td>
<td>11.5 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>8 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>8 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>7.5 dB</td>
<td>11.5 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>6 dB</td>
<td>10 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>13 dB</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>±1.5 dB</td>
</tr>
<tr>
<td>45 MHz to 20 GHz</td>
<td>--</td>
<td>±1.5 dB</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>±2.0 dB</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>±2.0 dB</td>
</tr>
<tr>
<td>50 GHz to 67 GHz</td>
<td>--</td>
<td>±3.0 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>±4.5 dB</td>
</tr>
</tbody>
</table>
## Transmission Tracking

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>--</th>
<th>Typical:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>±1.5 dB</td>
</tr>
<tr>
<td>45 MHz to 20 GHz</td>
<td>--</td>
<td>±1.5 dB</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>±2.0 dB</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>±2.0 dB</td>
</tr>
<tr>
<td>50 GHz to 67 GHz</td>
<td>--</td>
<td>±3.0 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>±4.5 dB</td>
</tr>
</tbody>
</table>

## Crosstalk - Standard

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>--</th>
<th>Typical:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz b</td>
<td>--</td>
<td>-63 dB</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>-87 dB</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>-110 dB</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>-105 dB</td>
<td>--</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>-111 dB</td>
<td>--</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>-106 dB</td>
<td>--</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>-104 dB</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>-98 dB</td>
<td>--</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>-100 dB</td>
<td>--</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>-97 dB</td>
<td>--</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>-94 dB</td>
<td>--</td>
</tr>
<tr>
<td>67 GHz to 70 GHz b</td>
<td>-94 dB</td>
<td>--</td>
</tr>
</tbody>
</table>

## Crosstalk - Option 014

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>--</th>
<th>Typical (for Option 080 enabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz b</td>
<td>--</td>
<td>-63 dB</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>-87 dB</td>
<td>-87 dB</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Crosstalk $^{d}$</td>
<td>Typical $^{d}$</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>-110 dB</td>
<td>-110 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>-105 dB</td>
<td>-105 dB</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>-111 dB</td>
<td>-111 dB</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>-104 dB</td>
<td>-104 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>-102 dB</td>
<td>-102 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>-96 dB</td>
<td>-96 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>-98 dB</td>
<td>-98 dB</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>-95 dB</td>
<td>-95 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>-90 dB</td>
<td>-90 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz $^{b}$</td>
<td>--</td>
<td>-90 dB</td>
</tr>
</tbody>
</table>

**Crosstalk $^{d}$ - Option 014 & UNL**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Crosstalk $^{d}$</th>
<th>Typical (for Option 080 enabled $^{e}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz $^{b}$</td>
<td>--</td>
<td>-63 dB</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>-87 dB</td>
<td>-87 dB</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>-110 dB</td>
<td>-110 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>-104 dB</td>
<td>-104 dB</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>-108 dB</td>
<td>-108 dB</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>-101 dB</td>
<td>-101 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>-99 dB</td>
<td>-99 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>-92 dB</td>
<td>-92 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>-94 dB</td>
<td>-94 dB</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>-91 dB</td>
<td>-91 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>-84 dB</td>
<td>-84 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz $^{b}$</td>
<td>--</td>
<td>-84 dB</td>
</tr>
</tbody>
</table>

$^{a}$ Specifications apply over environment temperature of 23°C +/- 3°C, with less than 1°C deviation from the calibration temperature.

$^{b}$ Dependent on Option 080.
b Typical performance.

c Transmission tracking performance noted here is normalized to the insertion loss characteristics of the cable used so that the indicated performance is independent of the cable used.

d Measurement conditions: normalized to a thru, measured with two shorts, 10 Hz IF bandwidth, averaging factor of 16, alternate mode, source power set to the lesser of the maximum power out or the maximum receiver power.

e 0 Hz offset.

**Table 11. Test Port Output**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Frequency Range</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standard</td>
<td>Options</td>
</tr>
<tr>
<td>E8361A</td>
<td>45 MHz to 67 GHz</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>(Operates 10 MHz to 70 GHz)</td>
<td></td>
</tr>
<tr>
<td><strong>Nominal Power</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Std and Options without UNL</td>
<td>Options with UNL</td>
</tr>
<tr>
<td>E8361A</td>
<td>-15 dBm</td>
<td>-17 dBm</td>
</tr>
<tr>
<td><strong>Frequency Resolution</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 Hz</td>
<td>--</td>
</tr>
<tr>
<td><strong>CW Accuracy</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>+/-1 ppm</td>
<td>--</td>
</tr>
<tr>
<td><strong>Frequency Stability</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>+/-0.05 ppm. -10° to 70° C, typical; +/-0.1 ppm/yr maximum, typical</td>
</tr>
<tr>
<td><strong>Power Level Accuracy</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standard</td>
<td>Opt 014</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>+/-1.5 dB (typical)</td>
<td>+/-1.5 dB (typical)</td>
</tr>
<tr>
<td>45 MHz to</td>
<td>+/-1.5 dB</td>
<td>+/-1.5 dB</td>
</tr>
<tr>
<td>Variation from nominal power</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Linearity 10 MHz to 45 MHz</td>
<td>Linearity 45 MHz to 67 GHz</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>+/-1.5 dB</td>
<td>+/-1.5 dB</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>+/-2.0 dB</td>
<td>+/-2.0 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>+/-3.0 dB</td>
<td>+/-3.0 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>+/-3.0 dB</td>
<td>+/-3.0 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>+/-3.5 dB</td>
<td>+/-3.5 dB</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>+/-4.0 dB</td>
<td>+/-4.0 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>+/-4.0 dB</td>
<td>+/-4.0 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>+/-4.0 dB (typical)</td>
<td>+/-4.0 dB (typical)</td>
</tr>
</tbody>
</table>

**Power Level Linearity**

Test reference is at the nominal power level (step attenuator at 0 dB)

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Linearity 10 MHz to 45 MHz</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>+/-1.0 dB for power&lt;=-5 dBm (\text{typical})</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 67 GHz</td>
<td>+/-1.0 dB for power&lt;=-5 dBm (\text{typical})</td>
<td>--</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>+/-1.0 dB for power&lt;=-5 dBm (\text{typical})</td>
<td>--</td>
</tr>
</tbody>
</table>

**Power Range**

Note: If power is set above the maximum specified leveled power, the test port output signal may show non-linear effects that are dependent on the DUT.

<table>
<thead>
<tr>
<th>Power Range</th>
<th>Standard</th>
<th>Opt 014</th>
<th>Opt 014 &amp; UNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Power Level 1</td>
<td>Power Level 2</td>
<td>Power Level 3</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>-25 to -9 dBm</td>
<td>-25 to -9 dBm</td>
<td>-75 to -9 dBm</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>-25 to -3 dBm</td>
<td>-25 to -3 dBm</td>
<td>-75 to 0 dBm</td>
</tr>
<tr>
<td>500 MHz to 750 MHz</td>
<td>-25 to 0 dBm</td>
<td>-25 to 0 dBm</td>
<td>-75 to 0 dBm</td>
</tr>
<tr>
<td>750 MHz to 10 GHz</td>
<td>-27 to -1 dBm</td>
<td>-27 to -1 dBm</td>
<td>-77 to -1 dBm</td>
</tr>
<tr>
<td>10 GHz to 30 GHz</td>
<td>-27 to -2 dBm</td>
<td>-27 to -3 dBm</td>
<td>-77 to 0 dBm</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>-27 to -1 dBm</td>
<td>-27 to -2 dBm</td>
<td>-77 to 0 dBm</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>-27 to -7 dBm</td>
<td>-27 to -8 dBm</td>
<td>-77 to -10 dBm</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>-27 to -1 dBm</td>
<td>-27 to -2 dBm</td>
<td>-77 to -6 dBm</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>-27 to -3 dBm</td>
<td>-27 to -4 dBm</td>
<td>-77 to -8 dBm</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>-27 to -5 dBm</td>
<td>-27 to -7 dBm</td>
<td>-77 to -13 dBm</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>-27 to -5 dBm</td>
<td>-27 to -7 dBm</td>
<td>-77 to -13 dBm</td>
</tr>
</tbody>
</table>

### Power Sweep Range (ALC)

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Standard</th>
<th>Opt 014</th>
<th>Opt 014 &amp; UNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>16 dB (typical)</td>
<td>16 dB (typical)</td>
<td>16 dB (typical)</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>22 dB</td>
<td>22 dB</td>
<td>22 dB</td>
</tr>
</tbody>
</table>

ALC range starts at maximum leveled output power and decreases by power level indicated in the table.
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Power Resolution 1</th>
<th>Power Resolution 2</th>
<th>Power Resolution 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>500 MHz to 750 MHz</td>
<td>25 dB</td>
<td>25 dB</td>
<td>25 dB</td>
</tr>
<tr>
<td>750 GHz to 10 GHz</td>
<td>26 dB</td>
<td>26 dB</td>
<td>26 dB</td>
</tr>
<tr>
<td>10 GHz to 30 GHz</td>
<td>25 dB</td>
<td>24 dB</td>
<td>24 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>26 dB</td>
<td>25 dB</td>
<td>22 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>20 dB</td>
<td>19 dB</td>
<td>17 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>26 dB</td>
<td>25 dB</td>
<td>21 dB</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>24 dB</td>
<td>23 dB</td>
<td>19 dB</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>22 dB</td>
<td>20 dB</td>
<td>14 dB</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>22 dB (typical)</td>
<td>20 dB (typical)</td>
<td>14 dB (typical)</td>
</tr>
</tbody>
</table>

**Power Resolution**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.01 dB</td>
<td>--</td>
</tr>
</tbody>
</table>

**Phase Noise**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Any Option</td>
</tr>
</tbody>
</table>

**10 kHz offset from center frequency, nominal power at test port**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Power Resolution 1</th>
<th>Power Resolution 2</th>
<th>Power Resolution 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 10 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>24 GHz to 70 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**typical:**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Power Resolution 1</th>
<th>Power Resolution 2</th>
<th>Power Resolution 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
<td>-80 dBC</td>
</tr>
<tr>
<td>45 MHz to 10 GHz</td>
<td>--</td>
<td>--</td>
<td>-70 dBC</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>--</td>
<td>--</td>
<td>-60 dBC</td>
</tr>
<tr>
<td>24 GHz to 70 GHz</td>
<td>--</td>
<td>--</td>
<td>-55 dBC</td>
</tr>
<tr>
<td>10 kHz offset from center frequency, nominal power at test port - Option 080 enabled</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td></td>
<td>typical:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
<td>-80 dBC</td>
</tr>
<tr>
<td>45 MHz to 10 GHz</td>
<td>--</td>
<td>--</td>
<td>-70 dBC</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>--</td>
<td>--</td>
<td>-60 dBC</td>
</tr>
<tr>
<td>24 GHz to 70 GHz</td>
<td>--</td>
<td>--</td>
<td>-55 dBC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>100 kHz offset from center frequency, nominal power at test port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
</tr>
<tr>
<td>45 MHz to 10 GHz</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
</tr>
<tr>
<td>24 GHz to 70 GHz</td>
</tr>
</tbody>
</table>

| 100 kHz offset from center frequency, nominal power at test port - Option 080 enabled |
|----------------------------------|----------------|----------------|----------------|
|                                    | typical:       |                |                |
| 10 MHz to 45 MHz                  | --             | --             | -85 dBC        |
| 45 MHz to 10 GHz                  | --             | --             | -80 dBC        |
| 10 GHz to 24 GHz                  | --             | --             | -70 dBC        |
| 24 GHz to 70 GHz                  | --             | --             | -60 dBC        |

<table>
<thead>
<tr>
<th>1 MHz offset from center frequency, nominal power at test port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

2783
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>10 MHz to 45 MHz</th>
<th>45 MHz to 10 GHz</th>
<th>10 GHz to 24 GHz</th>
<th>24 GHz to 70 GHz</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-115 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-110 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-105 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-95 dBc</td>
</tr>
</tbody>
</table>

**1 MHz offset from center frequency, nominal power at test port - Option 080 enabled**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>10 MHz to 45 MHz</th>
<th>45 MHz to 10 GHz</th>
<th>10 GHz to 24 GHz</th>
<th>24 GHz to 70 GHz</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-110 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-105 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-95 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-85 dBc</td>
</tr>
</tbody>
</table>

**Harmonics (2nd or 3rd)**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>10 MHz to 500 MHz</th>
<th>500 MHz to 10 GHz</th>
<th>10 GHz to 24 GHz</th>
<th>24 GHz to 50 GHz</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-10 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-15 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-23 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-16 dBc</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-13 dBc</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Standard or UNL</td>
<td>Opt 014 or 014 &amp; UNL</td>
<td>Supplemental</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------</td>
<td>-----------------------</td>
<td>--------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60 GHz to 70 GHz</td>
<td>--</td>
<td>--</td>
<td>-19 dBC</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Non-Harmonic Spurious (at Nominal Output Power)**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Standard or UNL</th>
<th>Opt 014 or 014 &amp; UNL</th>
<th>Supplemental</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 20 GHz</td>
<td>--</td>
<td>--</td>
<td>-50 dBC typical, for offset frequency &gt; 1 kHz</td>
</tr>
<tr>
<td>20 GHz to 70 GHz</td>
<td>--</td>
<td>--</td>
<td>-30 dBC typical, for offset frequency &gt; 1 kHz</td>
</tr>
</tbody>
</table>

a. Test port output is a specification when the source is set to Port 1, and a characteristic when the source is set to Port 2.
b. Typical performance.
c. Preset power.
d. Power Level Linearity is a specification when the source is set to Port 1, and a typical when the source is set to Port 2.
e. Test port power is specified into nominal 50 ohms.
f. Power to which the source can be set and phase lock is assured.
g. +/-1.6 dB for power >-5 dBm.

**Table 12: Test Port Input**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Test Port Noise Floor</strong>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Hz IF Bandwidth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>&lt;-70 dBm (typical)</td>
<td>&lt;-70 dBm (typical)</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;-90 dBm</td>
<td>&lt;-90 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>&lt;-112 dBm</td>
<td>&lt;-112 dBm</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>&lt;-112 dBm</td>
<td>&lt;-112 dBm</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>&lt;-116 dBm</td>
<td>&lt;-115 dBm</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>&lt;-105 dBm</td>
<td>&lt;-104 dBm</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>&lt;-105 dBm</td>
<td>&lt;-104 dBm</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>&lt;-103 dBm</td>
<td>&lt;-102 dBm</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Voltage 1</td>
<td>Voltage 2</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>&lt;-101 dBm</td>
<td>&lt;-100 dBm</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>&lt;-100 dBm</td>
<td>&lt;-99 dBm</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>&lt;-99 dBm</td>
<td>&lt;-97 dBm</td>
</tr>
<tr>
<td>67 GHz to 70 GHz&lt;sup&gt;2&lt;/sup&gt;</td>
<td>&lt;-99 dBm (typical)</td>
<td>&lt;-97 dBm (typical)</td>
</tr>
<tr>
<td>1 KHz IF Bandwidth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz&lt;sup&gt;2&lt;/sup&gt;</td>
<td>&lt;-50 dBm (typical)</td>
<td>&lt;-50 dBm (typical)</td>
</tr>
<tr>
<td>45 MHz to 500 MHz&lt;sup&gt;3, 4&lt;/sup&gt;</td>
<td>&lt;-70 dBm</td>
<td>&lt;-70 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>&lt;-92 dBm</td>
<td>&lt;-92 dBm</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>&lt;-92 dBm</td>
<td>&lt;-92 dBm</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>&lt;-96 dBm</td>
<td>&lt;-95 dBm</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>&lt;-85 dBm</td>
<td>&lt;-84 dBm</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>&lt;-85 dBm</td>
<td>&lt;-84 dBm</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>&lt;-83 dBm</td>
<td>&lt;-82 dBm</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>&lt;-81 dBm</td>
<td>&lt;-80 dBm</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>&lt;-80 dBm</td>
<td>&lt;-79 dBm</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>&lt;-79 dBm</td>
<td>&lt;-77 dBm</td>
</tr>
<tr>
<td>67 GHz to 70 GHz&lt;sup&gt;2&lt;/sup&gt;</td>
<td>&lt;-79 dBm (typical)</td>
<td>&lt;-77 dBm (typical)</td>
</tr>
<tr>
<td>Test Port Noise Floor&lt;sup&gt;1&lt;/sup&gt; Option 080 enabled&lt;sup&gt;5&lt;/sup&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Hz IF Bandwidth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz&lt;sup&gt;3, 4&lt;/sup&gt;</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>IF Bandwidth</td>
<td>Noise Floor</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>--</td>
<td>&lt;=-112 dBm</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>--</td>
<td>&lt;=-115 dBm</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>--</td>
<td>&lt;=-104 dBm</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>--</td>
<td>&lt;=-104 dBm</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>--</td>
<td>&lt;=-102 dBm</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>--</td>
<td>&lt;=-100 dBm</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>--</td>
<td>&lt;=-99 dBm</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>--</td>
<td>&lt;=-97 dBm</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>&lt;=-97 dBm</td>
</tr>
<tr>
<td><strong>1 KHz IF Bandwidth</strong></td>
<td></td>
<td><strong>typical:</strong></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>&lt;=-50 dBm</td>
</tr>
<tr>
<td>45 MHz to 500 MHz 3, 4</td>
<td>--</td>
<td>&lt;=-70 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>--</td>
<td>&lt;=-92 dBm</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>--</td>
<td>&lt;=-92 dBm</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>--</td>
<td>&lt;=-95 dBm</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>--</td>
<td>&lt;=-84 dBm</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>--</td>
<td>&lt;=-84 dBm</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>--</td>
<td>&lt;=-82 dBm</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>--</td>
<td>&lt;=-80 dBm</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>--</td>
<td>&lt;=-79 dBm</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>--</td>
<td>&lt;=-77 dBm</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>&lt;=-77 dBm</td>
</tr>
</tbody>
</table>

Direct Receiver Access Input Noise Floor

---

1 KHz IF Bandwidth

- **typical:**
  - 10 MHz to 45 MHz: <=-50 dBm
  - 45 MHz to 500 MHz 3, 4: <=-70 dBm
  - 500 MHz to 2 GHz: <=-92 dBm
  - 2 GHz to 10 GHz: <=-92 dBm
  - 10 GHz to 24 GHz: <=-95 dBm
  - 24 GHz to 30 GHz: <=-84 dBm
  - 30 GHz to 40 GHz: <=-84 dBm
  - 40 GHz to 45 GHz: <=-82 dBm
  - 45 GHz to 50 GHz: <=-80 dBm
  - 50 GHz to 60 GHz: <=-79 dBm
  - 60 GHz to 67 GHz: <=-77 dBm
  - 67 GHz to 70 GHz: <=-77 dBm

**Option 016** degrades performance by 2 dB.

**Option 016** degrades performance by 3 dB.
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Standard</th>
<th>Option 014 or 014 &amp; UNL (typical)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>10 Hz IF Bandwidth</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>&lt;-106 dBm</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>--</td>
<td>&lt;-105 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>--</td>
<td>&lt;-125.5 dBm</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>--</td>
<td>&lt;-125 dBm</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>--</td>
<td>&lt;-128 dBm</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>--</td>
<td><strong>Option 016 degrades performance by 2 dB.</strong> &lt;-117.5 dBm</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>--</td>
<td>&lt;-117 dBm</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>--</td>
<td>&lt;-115 dBm</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>--</td>
<td>&lt;-112.5 dBm</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>--</td>
<td>&lt;-111 dBm</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>--</td>
<td><strong>Option 016 degrades performance by 3 dB</strong> &lt;-108 dBm</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
<td>&lt;-107 dBm</td>
</tr>
<tr>
<td><strong>1 KHz IF Bandwidth</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>&lt;-86 dBm</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>--</td>
<td>&lt;-85 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>--</td>
<td>&lt;-105.5 dBm</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>--</td>
<td>&lt;-105 dBm</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>--</td>
<td>&lt;-108 dBm</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>--</td>
<td><strong>Option 016 degrades performance by 2 dB.</strong> &lt;-97.5 dBm</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>--</td>
<td>&lt;-97 dBm</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>--</td>
<td>&lt;-95 dBm</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Default Noise Floor</td>
<td>Option 016 Degradation</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>&lt;=-92.5 dBm</td>
<td></td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>&lt;=-91 dBm</td>
<td></td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>&lt;=-88 dBm</td>
<td>Option 016 degrades performance by 3 dB.</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>&lt;=-87 dBm</td>
<td></td>
</tr>
</tbody>
</table>

**Direct Receiver Access Input Noise Floor - Option 080 enabled**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Standard Noise Floor</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 Hz IF Bandwidth</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>&lt;=-106 dBm</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;=-105 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>&lt;=-125.5 dBm</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>&lt;=-125 dBm</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>&lt;=-128 dBm</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>&lt;=-117.5 dBm</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>&lt;=-117 dBm</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>&lt;=-115 dBm</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>&lt;=-112.5 dBm</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>&lt;=-111 dBm</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>&lt;=-108 dBm</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>&lt;=-107 dBm</td>
</tr>
</tbody>
</table>

**1 KHz IF Bandwidth**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Typical Noise Floor</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>&lt;=-86 dBm</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;=-85 dBm</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Receiver Compression Level (measured at test ports)</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>--</td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>--</td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>--</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>--</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>--</td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>--</td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>--</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
</tr>
</tbody>
</table>

### Receiver Compression Level (measured at test ports)

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Specifications</th>
<th>014 &amp; UNL</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>negligible</td>
<td></td>
<td></td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;0.1 dB at -9.5 dBm and &lt;0.25 dB at -3 dBm</td>
<td>&lt;0.1 dB at -9.5 dBm and &lt;0.25 dB at +3 dBm</td>
<td>&lt;0.1 dB at +0.5 dBm and &lt;0.25 dB at +8 dBm</td>
</tr>
<tr>
<td>500 MHz to 5 GHz</td>
<td>&lt;0.1 dB at -8 dBm and &lt;0.25 dB at -1 dBm</td>
<td>&lt;0.1 dB at -8 dBm and &lt;0.25 dB at 0 dBm</td>
<td>&lt;0.1 dB at -4.0 dBm and &lt;0.25 dB at +3 dBm</td>
</tr>
<tr>
<td>5 GHz to 30 GHz</td>
<td>&lt;0.1 dB at -8.5 dBm and &lt;0.25 dB at -2 dBm</td>
<td>&lt;0.1 dB at -8.5 dBm and &lt;0.25 dB at +1 dBm</td>
<td>&lt;0.1 dB at -1.0 dBm and &lt;0.25 dB at +6 dBm</td>
</tr>
<tr>
<td>30 GHz to 67 GHz</td>
<td>&lt;0.1 dB at -10.5 dBm and &lt;0.15 dB at -7 dBm</td>
<td>&lt;0.1 dB at -8.0 dBm and &lt;0.15 dB at -3 dBm</td>
<td>&lt;0.1 dB at -9.5 dBm and &lt;0.15 dB</td>
</tr>
</tbody>
</table>

2790
### 67 GHz to 70 GHz

<table>
<thead>
<tr>
<th>System Compression Level</th>
<th>Standard</th>
<th>Option 014 or 014 &amp; UNL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NA</td>
<td></td>
</tr>
</tbody>
</table>

### Third Order Intercept\(^{10}\) - Tone spacing from 100 kHz - 5 MHz

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Any Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 500 MHz</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 24 GHz</td>
<td>--</td>
</tr>
<tr>
<td>24 GHz to 40 GHz</td>
<td>--</td>
</tr>
<tr>
<td>40 to 50 GHz</td>
<td>--</td>
</tr>
<tr>
<td>50 to 67 GHz</td>
<td>--</td>
</tr>
</tbody>
</table>

### Third Order Intercept\(^{10}\) - Tone spacing from 5 MHz - 20 MHz

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Any Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 500 MHz</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 24 GHz</td>
<td>--</td>
</tr>
<tr>
<td>24 GHz to 40 GHz</td>
<td>--</td>
</tr>
<tr>
<td>40 to 50 GHz</td>
<td>--</td>
</tr>
<tr>
<td>50 to 67 GHz</td>
<td>--</td>
</tr>
</tbody>
</table>

### Third Order Intercept\(^{10}\) - Tone spacing from 20 MHz - 50 MHz

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Any Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>24 GHz to 40 GHz</td>
<td>--</td>
</tr>
<tr>
<td>40 to 50 GHz</td>
<td>--</td>
</tr>
<tr>
<td>50 to 67 GHz</td>
<td>--</td>
</tr>
</tbody>
</table>

---

2791
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Standard or 014</th>
<th>014 &amp; UNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 500 MHz</td>
<td>NA</td>
<td>+26 dBm</td>
</tr>
<tr>
<td>500 MHz to 24 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>24 to 40 GHz</td>
<td>--</td>
<td>+24 dBm</td>
</tr>
<tr>
<td>40 to 50 GHz</td>
<td>--</td>
<td>+25 dBm</td>
</tr>
<tr>
<td>50 to 67 GHz</td>
<td>--</td>
<td>+27 dBm</td>
</tr>
</tbody>
</table>

**Trace Noise Magnitude**

1 kHz IF bandwidth. Ratio measurement, nominal power at test port.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Standard or 014</th>
<th>014 &amp; UNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>&lt;0.150 dB rms (typical)</td>
<td>&lt;0.150 dB rms (typical)</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;0.010 dB rms</td>
<td>&lt;0.010 dB rms</td>
</tr>
<tr>
<td>500 MHz to 24 GHz</td>
<td>&lt;0.006 dB rms</td>
<td>&lt;0.006 dB rms</td>
</tr>
<tr>
<td>24 GHz to 67 GHz</td>
<td>&lt;0.006 dB rms</td>
<td>&lt;0.009 dB rms</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>&lt;0.006 dB rms (typical)</td>
<td>&lt;0.009 dB rms (typical)</td>
</tr>
</tbody>
</table>

**Trace Noise Magnitude - Option 080 enabled\(^2,5\)**

1 kHz IF bandwidth. Ratio measurement, nominal power at test port.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Standard or 014</th>
<th>014 &amp; UNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>&lt;0.150 dB rms (typical)</td>
<td>&lt;0.150 dB rms (typical)</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;0.010 dB rms (typical)</td>
<td>&lt;0.010 dB rms (typical)</td>
</tr>
<tr>
<td>500 MHz to 24 GHz</td>
<td>&lt;0.006 dB rms (typical)</td>
<td>&lt;0.006 dB rms (typical)</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Trace Noise Phase (dB rms)</td>
<td>Phase (° rms)</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>24 GHz to 67 GHz</td>
<td>&lt;0.009 dB rms (typical)</td>
<td>&lt;0.012 dB rms (typical)</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>&lt;0.009 dB rms (typical)</td>
<td>&lt;0.012 dB rms (typical)</td>
</tr>
</tbody>
</table>

**Trace Noise Phase**

1 kHz IF bandwidth. Ratio measurement, nominal power at test port.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Trace Noise Phase (° rms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any Option</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz²</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz⁴</td>
<td>&lt;0.100° rms</td>
</tr>
<tr>
<td>500 MHz to 24 GHz</td>
<td>&lt;0.060° rms</td>
</tr>
<tr>
<td>24 GHz to 67 GHz</td>
<td>&lt;0.100° rms</td>
</tr>
<tr>
<td>67 GHz to 70 GHz²</td>
<td>--</td>
</tr>
</tbody>
</table>

**Trace Noise Phase - Option 080 enabled², ⁵**

1 kHz IF bandwidth. Ratio measurement, nominal power at test port.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Phase (° rms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any Option</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz⁴</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 24 GHz</td>
<td>--</td>
</tr>
<tr>
<td>24 GHz to 67 GHz</td>
<td>--</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Minimum Response</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>--</td>
</tr>
</tbody>
</table>

**Reference Level Magnitude**

<table>
<thead>
<tr>
<th></th>
<th>Range</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+/-500 dB</td>
<td>0.001 dB</td>
</tr>
</tbody>
</table>

**Reference Level Phase**

<table>
<thead>
<tr>
<th></th>
<th>Range</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+/-500°</td>
<td>0.01°</td>
</tr>
</tbody>
</table>

**Stability Magnitude**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>+/-0.05 dB/°C</td>
</tr>
<tr>
<td>45 MHz to 50 GHz</td>
<td>+/-0.02 dB/°C</td>
</tr>
<tr>
<td>50 GHz to 70 GHz</td>
<td>+/-0.04 dB/°C</td>
</tr>
</tbody>
</table>

**Stability Phase**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>+/-0.5°/°C</td>
</tr>
<tr>
<td>45 MHz to 20 GHz</td>
<td>+/-0.2°/°C</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>+/-0.5°/°C</td>
</tr>
<tr>
<td>40 GHz to 70 GHz</td>
<td>+/-0.8°/°C</td>
</tr>
</tbody>
</table>

*Typical ratio measurement, made at the test port.*

*Typical ratio measurement, measured at the test port.*
<table>
<thead>
<tr>
<th>Damage Input Level</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Port 1 and 2</td>
<td></td>
</tr>
<tr>
<td>R1, R2 in</td>
<td></td>
</tr>
<tr>
<td>A, B in</td>
<td></td>
</tr>
<tr>
<td>Coupler Thru (Option 014 or 014 &amp; UNL)</td>
<td></td>
</tr>
<tr>
<td>Coupler Arm (Option 014 or 014 &amp; UNL)</td>
<td></td>
</tr>
<tr>
<td>Typical:</td>
<td></td>
</tr>
<tr>
<td>+27 dBm or +/-40 VDC</td>
<td></td>
</tr>
<tr>
<td>+15 dBm or +/-15 VDC</td>
<td></td>
</tr>
<tr>
<td>+15 dBm or +/-7 VDC</td>
<td></td>
</tr>
<tr>
<td>+27 dBm or +/-40 VDC</td>
<td></td>
</tr>
<tr>
<td>+30 dBm or +/-7 VDC</td>
<td></td>
</tr>
</tbody>
</table>

1. Total average (rms) noise power calculated as the mean value of a linear magnitude trace expressed in dBm.
2. Typical performance.
3. Noise floor may be degraded by 10 dB at particular frequencies (multiples of 5 MHz) due to spurious receiver residuals.
4. Specified value is for worst-case noise floor at 45 MHz
5. 50 Hz offset
6. Coupler roll-off will reduce compression below 500 MHz. Ultimately, at 45 MHz, compression is negligible.
7. Specified value is for worst-case compression at 500 MHz.
8. This compression level comes from the dynamic accuracy curve with -30 dBm reference test port power.
9. Option 016 degrades performance by 3 dB.
10. TOI is a typical specification that applies while the network analyzer receiver is in its linear range.
11. Trace noise magnitude may be degraded to 20 mdB rms at harmonic frequencies of the first IF (8.33 MHz) below 80 MHz.
12. Stability is defined as a ratio measurement made at the test port.

**Table 13. Dynamic Accuracy (Specification)***

Accuracy of the test port input power reading relative to the reference input power level.

**Note:** If power is set above the maximum specified leveled power, the test port output signal may show non-linear effects that are dependent on the DUT

**Dynamic Accuracy, 0.045 GHz**
Dynamic Accuracy, 0.500 GHz
Dynamic Accuracy, 0.500 - 5 GHz

**Magnitude**

![Magnitude Graph]

**Phase**

![Phase Graph]
Dynamic Accuracy, 5 - 30 GHz
Dynamic Accuracy, 30 - 67 GHz
Dynamic Accuracy, 67 - 70 GHz

Phase

Magnitude

Testport Power (dBm)

Accuracy (degrees)

Accuracy (dB)

Testport Power (dBm)
a Dynamic accuracy is verified with the following measurements:

- compression over frequency
- IF linearity at a single frequency of 1.195 GHz and a reference level of -20 dBm for an input power range of 0 to -120 dBm.

Table 14. Test Port Input (Group Delay)\textsuperscript{a}

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental Information (typ.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aperture (selectable)</td>
<td>--</td>
<td>(frequency span)/(number of points -1)</td>
</tr>
<tr>
<td>Maximum Aperture</td>
<td>--</td>
<td>20% of frequency span</td>
</tr>
<tr>
<td>Range</td>
<td>--</td>
<td>0.5 x (1/minimum aperture)</td>
</tr>
<tr>
<td>Maximum Delay</td>
<td>--</td>
<td>Limited to measuring no more than 180° of phase change within the minimum aperture.</td>
</tr>
<tr>
<td>Accuracy</td>
<td>--</td>
<td>See graph below. Char.</td>
</tr>
</tbody>
</table>

The following graph shows characteristic group delay accuracy with full 2-port calibration and a 10 Hz IF bandwidth. Insertion loss is assumed to be < 2 dB and electrical length to be ten meters.
In general, the following formula can be used to determine the accuracy, in seconds, of specific group delay measurement:

\[ \pm \text{Phase Accuracy (deg)}/(360 \times \text{Aperture (Hz)}) \]

Depending on the aperture and device length, the phase accuracy used is either incremental phase accuracy or worst case phase accuracy.

Group delay is computed by measuring the phase change within a specified frequency step (determined by the frequency span and the number of points per sweep).

**General Information**

- **Miscellaneous Information**
- **Front Panel**
- **Rear Panel**
- **Environment and Dimensions**

**Table 15. Miscellaneous Information**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>System IF Bandwidth Range</td>
<td>--</td>
<td>1 Hz to 40 kHz, nominal</td>
</tr>
<tr>
<td>CPU</td>
<td>--</td>
<td>Intel® 1.1 GHz Pentium® M with 1 GByte RAM.</td>
</tr>
</tbody>
</table>

**Table 16. Front Panel Information**
<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF Connectors</td>
<td></td>
</tr>
<tr>
<td>E8361A</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>1.85 mm (male), 50 ohm, (nominal)</td>
</tr>
<tr>
<td>Center Pin Recession</td>
<td>0.002 in. (characteristic)</td>
</tr>
<tr>
<td>Display</td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>21.3 cm (8.4 in) diagonal color active matrix LCD; 640 (horizontal) X 480 (vertical) resolution</td>
</tr>
<tr>
<td>Refresh Rate</td>
<td>Vertical 59.83 Hz; Horizontal 31.41 kHz</td>
</tr>
<tr>
<td>Pixels</td>
<td>When running the analyzer's built-in Display Test, one or more of the following symptoms indicate a faulty display assembly:</td>
</tr>
<tr>
<td></td>
<td>• A complete row or column of &quot;stuck on&quot; or &quot;dark&quot; pixels.</td>
</tr>
<tr>
<td></td>
<td>• More than six &quot;stuck on&quot; pixels (but not more than three green)</td>
</tr>
<tr>
<td></td>
<td>• More than twelve &quot;dark&quot; pixels (but not more than seven of the same color)</td>
</tr>
<tr>
<td></td>
<td>• Two or more consecutive &quot;stuck on&quot; pixels or three or more consecutive &quot;dark&quot; pixels (but no more than one set of two consecutive &quot;dark&quot; pixels)</td>
</tr>
<tr>
<td></td>
<td>• &quot;Stuck on&quot; or &quot;dark&quot; pixels less than 6.5 mm apart (excluding consecutive pixels)</td>
</tr>
<tr>
<td>Display Range</td>
<td></td>
</tr>
<tr>
<td>Magnitude</td>
<td>±200 dB (at 20 dB/div), max</td>
</tr>
<tr>
<td>Phase</td>
<td>±500°, max</td>
</tr>
<tr>
<td>Polar</td>
<td>10 pUnits, min</td>
</tr>
<tr>
<td></td>
<td>1000 Units, max</td>
</tr>
<tr>
<td>Display Resolution</td>
<td></td>
</tr>
<tr>
<td>Magnitude</td>
<td>0.001 dB/div, min</td>
</tr>
<tr>
<td>Phase</td>
<td>0.01°/div, min</td>
</tr>
<tr>
<td>Marker Resolution</td>
<td></td>
</tr>
<tr>
<td>Magnitude</td>
<td>0.001 dB, min</td>
</tr>
<tr>
<td>Phase</td>
<td>0.01°, min</td>
</tr>
<tr>
<td>Polar</td>
<td>0.01 mUnit; 0.01°, min</td>
</tr>
</tbody>
</table>
### Table 17. Rear Panel Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>10 MHz Reference In</strong></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Input Frequency</td>
<td>10 MHz ± 10 ppm, typical</td>
</tr>
<tr>
<td>Input Level</td>
<td>-15 dBm to +20 dBm, typical</td>
</tr>
<tr>
<td>Input Impedance</td>
<td>200 Ω, nom.</td>
</tr>
<tr>
<td><strong>10 MHz Reference Out</strong></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Output Frequency</td>
<td>10 MHz ± 1 ppm, typical</td>
</tr>
<tr>
<td>Signal Type</td>
<td>Sine Wave, typical</td>
</tr>
<tr>
<td>Output Level</td>
<td>+10 dBm ± 4 dB into 50 Ω, typical</td>
</tr>
<tr>
<td>Output Impedance</td>
<td>50 Ω, nominal</td>
</tr>
<tr>
<td>Harmonics</td>
<td>&lt; -40 dBc, typical</td>
</tr>
<tr>
<td><strong>Option H08 &amp; H11 Rear Panel Connectors (Typical)</strong></td>
<td></td>
</tr>
<tr>
<td>IF Connectors</td>
<td>A, R1, R2, B (BNC Connectors)</td>
</tr>
<tr>
<td>IF Connector Input Frequency</td>
<td>8 1/3 MHz</td>
</tr>
<tr>
<td>Nominal Input Impedance at IF Inputs</td>
<td>50</td>
</tr>
<tr>
<td>RF Damage Level to IF Connector Inputs</td>
<td>-20.0 dBm</td>
</tr>
<tr>
<td>DC Damage Level to IF Connector Inputs</td>
<td>25 volts</td>
</tr>
<tr>
<td>0.1 dB Compression Point at IF Inputs</td>
<td>-27.0 dBm</td>
</tr>
<tr>
<td>Pulse Input Connectors†</td>
<td>A, R1, R2, B (BNC Connectors)</td>
</tr>
<tr>
<td>Nominal Input Impedance at Pulse Inputs</td>
<td>1 Kohm</td>
</tr>
<tr>
<td><strong>Minimum IF Gate Width</strong></td>
<td>20 ns for less than 1 dB deviation from theoretical performance².</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>DC Damage Level to Pulse Connector Inputs</strong></td>
<td>5.5 volts</td>
</tr>
<tr>
<td><strong>Drive Voltage</strong></td>
<td>TTL (0, +5.0) Volts</td>
</tr>
</tbody>
</table>

### Rear Panel LO Power - Test Port Frequency

- **1.7 GHz to 20 GHz**
  - -7 to -16 dBm

### Rear Panel RF Power - Test Port Frequencies

- **1.7 GHz to 20 GHz**
  - -2 to -12 dBm (at -5 dBm test port power³)
- **10 GHz to 16 GHz**
  - 0 to -8 dBm (at -5 dBm test port power³)
- **16 GHz to 20 GHz**
  - +5 to -1 dBm (at -5 dBm test port power³)

### VGA Video Output

- **Connector** 15-pin mini D-Sub; Drives VGA compatible monitors
- **Devices Supported:**
  - **Resolutions:**
    - Flat Panel (TFT) 1024 X 768, 800 X 600, 640 X 480
    - Flat Panel (DSTN) 800 X 600, 640 X 480
    - CRT Monitor 1280 X 1024, 1024 X 768, 800 X 600, 640 X 480

Simultaneous operation of the internal and external displays is allowed, but with 640 X 480 resolution only. If you change resolution, you can only view the external display (internal display will "white out").

### Bias Input Connectors (Option UNL)

- **Bias current** 500 mA, maximum
- **Bias voltage** 40 Volts, maximum

### Test Set IO

- 25-pin D-Sub connector, available for external test set control

### Aux IO

<p>| 2816 |</p>
<table>
<thead>
<tr>
<th><strong>Handler IO</strong></th>
<th>25-pin D-Sub connector, male, analog and digital IO</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GPIB</strong></td>
<td>36-pin parallel I/O port; all input/output signals are default set to negative logic; can be reset to positive logic via GPIB command</td>
</tr>
<tr>
<td><strong>Parallel Port (LPT1)</strong></td>
<td>24-pin D-sub (Type D-24), female; compatible with IEEE-488.</td>
</tr>
<tr>
<td><strong>Serial Port (COM 1)</strong></td>
<td>25-pin D-Sub miniature connector, female; provides connection to printers or any other parallel port peripherals</td>
</tr>
<tr>
<td><strong>USB Port</strong></td>
<td>9-pin D-Sub, male; compatible with RS-232</td>
</tr>
</tbody>
</table>

**USB Port**

| Contact 1 | Vcc: 4.75 to 5.25 VDC, 500 mA, maximum |
| Contact 2 | -Data |
| Contact 3 | +Data |
| Contact 4 | Ground |

**LAN**

| **LAN** | 10/100BaseT Ethernet, 8-pin configuration; auto selects between the two data rates |

**Line Power**

<table>
<thead>
<tr>
<th>Frequency, Voltage</th>
<th>50/60/400 Hz for 100 - 120 V</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50/60 Hz for 220 - 240 V</td>
</tr>
<tr>
<td></td>
<td>Power supply is auto switching</td>
</tr>
<tr>
<td>Max</td>
<td>350 Watts</td>
</tr>
</tbody>
</table>

1 Pulse input connectors are operational only with Option H08 (Pulse Measurement Capability) enabled.
2 Based on deviation from signal reduction equation: Signal Reduction (dB) = $20 \log_{10}(\text{Duty cycle}) = 20 \log_{10}(\text{pulse width/period})$.
   Measured at Pulse Repetition Frequency (PFR) of 1 MHz.
3 Test port power has to be at a high enough level such that the Drop Cal does not occur. If Drop Cal occurs then the power out of the rear panel RF connector will drop by about 15 dB.
### Table 18. Analyzer Dimensions and Weight

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cabinet Dimensions</strong></td>
<td></td>
</tr>
<tr>
<td>Excluding front and rear panel hardware and feet</td>
<td>Height: 267 mm (10.5 in) Width: 426 mm (16.75 in) Depth: 427 mm (16.8 in)</td>
</tr>
<tr>
<td>As shipped - includes front panel connectors, rear panel bumpers, and feet.</td>
<td>Height: 280 mm (11.0 in) Width: 435 mm (17.1 in) Depth: 470 mm (18.5 in)</td>
</tr>
<tr>
<td>As shipped plus handles</td>
<td>Height: 280 mm (11.0 in) Width: 458 mm (18 in) Depth: 501 mm (19.7 in)</td>
</tr>
<tr>
<td>As shipped plus rack-mount flanges</td>
<td>Height: 280 mm (11.0 in) Width: 483 mm (19 in) Depth: 470 mm (18.5 in)</td>
</tr>
<tr>
<td>As shipped plus handles and rack-mount flanges</td>
<td>Height: 280 mm (11.0 in) Width: 483 mm (19 in) Depth: 501 mm (19.70 in)</td>
</tr>
<tr>
<td><strong>Weight</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Net</strong></td>
<td>E8361A 29 kg (64 lb), nominal</td>
</tr>
<tr>
<td><strong>Shipping</strong></td>
<td>E8361A 36.3 kg (80 lb), nominal</td>
</tr>
</tbody>
</table>

**Note:** For Regulatory and Environmental information, refer to the PNA Series Installation and Quick Start Guide, located at http://cp.literature.agilent.com/litweb/pdf/E8356-90001.pdf.

### Measurement Throughput Summary

- **Typical Cycle Time for Measurement Completion**
- **Cycle Time vs IF Bandwidth**
- **Cycle Time vs Number of Points**
- **Data Transfer Time**

**Table 19** Typical Cycle Time\(^a,\(^b\) (ms) for Measurement Completion
<table>
<thead>
<tr>
<th>Number of Points</th>
<th>201</th>
<th>401</th>
<th>1601</th>
<th>16,001</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start 28 GHz, Stop 30 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncorrected, 1-port cal</td>
<td>12</td>
<td>19</td>
<td>55</td>
<td>503</td>
</tr>
<tr>
<td>2-Port cal</td>
<td>29</td>
<td>44</td>
<td>124</td>
<td>1112</td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 10 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncorrected, 1-port cal</td>
<td>86</td>
<td>93</td>
<td>121</td>
<td>583</td>
</tr>
<tr>
<td>2-Port cal</td>
<td>179</td>
<td>199</td>
<td>267</td>
<td>1301</td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 20 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncorrected, 1-port cal</td>
<td>126</td>
<td>130</td>
<td>153</td>
<td>597</td>
</tr>
<tr>
<td>2-Port cal</td>
<td>264</td>
<td>275</td>
<td>335</td>
<td>1321</td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 40 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncorrected, 1-port cal</td>
<td>185</td>
<td>190</td>
<td>213</td>
<td>621</td>
</tr>
<tr>
<td>2-Port cal</td>
<td>382</td>
<td>401</td>
<td>459</td>
<td>1374</td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 50 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncorrected, 1-port cal</td>
<td>210</td>
<td>216</td>
<td>243</td>
<td>643</td>
</tr>
<tr>
<td>2-Port cal</td>
<td>436</td>
<td>450</td>
<td>522</td>
<td>1405</td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 67 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncorrected 1-port cal</td>
<td>244</td>
<td>254</td>
<td>300</td>
<td>645</td>
</tr>
<tr>
<td>2-Port cal</td>
<td>502</td>
<td>524</td>
<td>591</td>
<td>1423</td>
</tr>
</tbody>
</table>

a Typical performance.

b Includes sweep time, retrace time and band-crossing time. Analyzer display turned off with DISPLAY:ENABLE OFF. Add 21 ms for display on. Data for one trace (S11) measurement.
Table 20. Cycle Time vs IF Bandwidth\textsuperscript{a}

Applies to the Preset condition (201 points, correction off) except for the following changes:

- CF = 28 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)

<table>
<thead>
<tr>
<th>IF Bandwidth (Hz)</th>
<th>Cycle Time (ms)\textsuperscript{b}</th>
<th>Cycle Time (ms) Option 080 enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>40,000</td>
<td>11</td>
<td>100</td>
</tr>
<tr>
<td>35,000</td>
<td>12</td>
<td>101</td>
</tr>
<tr>
<td>30,000</td>
<td>13</td>
<td>102</td>
</tr>
<tr>
<td>20,000</td>
<td>16</td>
<td>106</td>
</tr>
<tr>
<td>10,000</td>
<td>30</td>
<td>127</td>
</tr>
<tr>
<td>7000</td>
<td>38</td>
<td>138</td>
</tr>
<tr>
<td>5000</td>
<td>50</td>
<td>152</td>
</tr>
<tr>
<td>3000</td>
<td>74</td>
<td>182</td>
</tr>
<tr>
<td>1000</td>
<td>274</td>
<td>326</td>
</tr>
<tr>
<td>300</td>
<td>694</td>
<td>782</td>
</tr>
<tr>
<td>100</td>
<td>1905</td>
<td>2054</td>
</tr>
<tr>
<td>30</td>
<td>6091</td>
<td>6355</td>
</tr>
<tr>
<td>10</td>
<td>17916</td>
<td>18372</td>
</tr>
</tbody>
</table>

\textsuperscript{a} Typical performance.
\textsuperscript{b} Cycle time includes sweep and retrace time.

Table 21. Cycle Time vs Number of Points\textsuperscript{a}

Applies to the Preset condition (35 kHz IF bandwidth, correction off) except for the following changes:

- CF = 28 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)
<table>
<thead>
<tr>
<th>Number of Points</th>
<th>Cycle Time (ms)(^b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>51</td>
<td>7</td>
</tr>
<tr>
<td>101</td>
<td>9</td>
</tr>
<tr>
<td>201</td>
<td>12</td>
</tr>
<tr>
<td>401</td>
<td>18</td>
</tr>
<tr>
<td>801</td>
<td>30</td>
</tr>
<tr>
<td>1601</td>
<td>55</td>
</tr>
<tr>
<td>16,001</td>
<td>497</td>
</tr>
</tbody>
</table>

a Typical performance.

b Cycle time includes sweep and retrace time.

**Table 22. Data Transfer Time (ms)**

<table>
<thead>
<tr>
<th>Number of Points</th>
<th>201</th>
<th>401</th>
<th>1601</th>
<th>16,001</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCPI over GPIB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(program executed on external PC)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32-bit floating point</td>
<td>7</td>
<td>12</td>
<td>43</td>
<td>435</td>
</tr>
<tr>
<td>64-bit floating point</td>
<td>12</td>
<td>22</td>
<td>84</td>
<td>856</td>
</tr>
<tr>
<td>ASCII</td>
<td>64</td>
<td>124</td>
<td>489</td>
<td>5054</td>
</tr>
<tr>
<td>SCPI (program executed in the analyzer)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32-bit floating point</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>64-bit floating point</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>ASCII</td>
<td>29</td>
<td>56</td>
<td>222</td>
<td>2220</td>
</tr>
<tr>
<td>COM (program executed in the analyzer)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32-bit floating point</td>
<td>&lt;0.4</td>
<td>0.4</td>
<td>0.5</td>
<td>1.9</td>
</tr>
<tr>
<td>Variant type</td>
<td>0.7</td>
<td>1</td>
<td>3</td>
<td>32</td>
</tr>
<tr>
<td>DCOM over LAN (program executed on external PC)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32-bit floating point</td>
<td>&lt;0.8</td>
<td>1</td>
<td>1.5</td>
<td>7.1</td>
</tr>
<tr>
<td>Variant type</td>
<td>1.8</td>
<td>2.7</td>
<td>8.5</td>
<td>80</td>
</tr>
</tbody>
</table>

a Typical performance

**Test Set Block Diagrams**

**E8361A - Standard Configuration and Standard Power Range**

**E8361A - Option UNL Standard Configuration with Extended Power Range and Bias - Tees**
Test Set with Option 014 Block Diagrams

E8361A - Option 014 – Configurable Test Set and Standard Power Range
### Item Table

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

**E8361A - Option 014 – Configurable Test Set and Standard Power Range, and Option 081 Reference Channel Transfer Switch**
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

E8361A - Option 014 Configurable Test Set, and Option UNL Extended Power Range and Bias - Tees
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

E8361A - Option 014 Configurable Test Set, and Option UNL Extended Power Range and Bias - Tees, and Option 081 Reference Channel Transfer Switch
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

E8361A - Option 014 Configurable Test Set and Option UNL, Extended Power Range and Bias - Tees and Option 016 Receiver Attenuators
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

E8361A - Option 014 Configurable Test Set, and Option UNL Extended Power Range and Bias - Tees, and Option 016 Receiver Attenuators, and Option 081 Reference Channel Transfer Switch
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

Last modified:

- **Jan. 10, 2007**  Revised line power, and CPU values; moved regulatory, operating, and non-operating info to the I&QS Guide.
- **Oct. 05, 2006**  Added 350W typical to line power
- **July 10, 2006**  Previous revision
Technical Specifications for the E8362A, E8363A, E8364A

Because the E8362A, E8363A, and E8364A network analyzer is no longer produced, the technical specifications are stored only on the Internet. To view or print the .pdf version of the specifications, visit our web site at http://www.agilent.com/find/pna, and search for "E836xA Specifications".

The uncertainty curves contained in this document apply only to the setup conditions listed. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the curves for your PNA setup. View the equations used to generate the uncertainty curves.
Technical Specifications for the E8362B, E8363B, E8364B
(Rev. 2007-10-11)

This is a complete list of the E8362B, E8363B, and E8364B network analyzer technical specifications.

- To optimize viewing of uncertainty curves, click the Maximize button.

- To view or print the .pdf version of the specifications, visit our web site at http://www.agilent.com/find/pna, and search for "E836xB Specifications"

- The uncertainty curves contained in this document apply only to the setup conditions listed. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the curves for your PNA setup. View the equations used to generate the uncertainty curves.

- Definitions

- Corrected System Performance
  - System Dynamic Range
  - Receiver Dynamic Range
  - 2.4mm Connectors
  - 2.92mm Connectors
  - 3.5mm Connectors
  - 7mm Connectors
  - Type-N Connectors
  - WR-28 Connectors
  - WR-42 Connectors
  - WR-62 Connectors
  - WR-90 Connectors

- Uncorrected System Performance
  - Test Port Output
  - Test Port Input
  - Dynamic Accuracy
  - Group Delay
Definitions

All specifications and characteristics apply over a 25 °C ±5 °C range (unless otherwise stated) and 90 minutes after the instrument has been turned on.

**Specification (spec.):** Warranted performance. Specifications include guardbands to account for the expected statistical performance distribution, measurement uncertainties, and changes in performance due to environmental conditions.

**Characteristic (char.):** A performance parameter that the product is expected to meet before it leaves the factory, but that is not verified in the field and is not covered by the product warranty. A characteristic includes the same guardbands as a specification.

**Typical (typ.):** Expected performance of an average unit which does not include guardbands. It is not covered by the product warranty.

**Nominal (nom.):** A general, descriptive term that does not imply a level of performance. It is not covered by the product warranty.

**Calibration:** The process of measuring known standards to characterize a network analyzer's systematic (repeatable) errors.

**Corrected (residual):** Indicates performance after error correction (calibration). It is determined by the quality of calibration standards and how well "known" they are, plus system repeatability, stability, and noise.

**Uncorrected (raw):** Indicates instrument performance without error correction. The uncorrected performance affects the stability of a calibration.

**Standard:** When referring to the analyzer, this includes no options unless noted otherwise.

Corrected System Performance

The specifications in this section apply for measurements made with the E836xB analyzer with the following conditions:

- 10 Hz IF bandwidth
- No averaging applied to data
- Isolation calibration with an averaging factor of 8
- System Dynamic Range
- Receiver Dynamic Range
- 2.4mm Connectors
- 2.92mm Connectors
- 3.5mm Connectors
- 7mm-Connectors
- Type-N Connectors
- WR-28 Connectors
- WR-42-Connectors
- WR-62 Connectors
- WR-90 Connectors

### Table 1. System Dynamic Range\(^a\)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB) at Test Port(^d)</th>
<th>Typical (dB) at Direct Receiver Access Input(^c)</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic Range (in a 10 Hz BW)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Standard Configuration and Standard Power Range</strong> (E836xB - Standard)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz(^d)</td>
<td>79 (typical)</td>
<td>NA</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz(^e)</td>
<td>94</td>
<td>NA</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>119</td>
<td>NA</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>122</td>
<td>NA</td>
<td>--</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>123</td>
<td>NA</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>114</td>
<td>NA</td>
<td>--</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>110</td>
<td>NA</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>109</td>
<td>NA</td>
<td>--</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>104</td>
<td>NA</td>
<td>--</td>
</tr>
<tr>
<td><strong>Configurable Test Set and Standard Power Range</strong> (E836xB - Option 014)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(^{a}\) See manufacturer's specification for details.\n\(^{b}\) Test Port refers to the location of the test signal.\n\(^{c}\) Direct Receiver Access Input refers to the location of the receiver input.\n\(^{d}\) Frequency band starts from 10 MHz to 45 MHz.\n\(^{e}\) Frequency band starts from 45 MHz to 500 MHz.
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Standard Configuration and Extended Power Range &amp; Bias-Tees (E836xB - Option UNL)</th>
<th>Configurable Test Set and Extended Power Range &amp; Bias-Tees (E836xB - Option 014/UNL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz(^d)</td>
<td>79 (typical)</td>
<td>79 (typical)</td>
</tr>
<tr>
<td>45 MHz to 500 MHz(^e)</td>
<td>94</td>
<td>92</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>119</td>
<td>117</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>122</td>
<td>120</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>121</td>
<td>121</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>111</td>
<td>112</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>107</td>
<td>108</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>105</td>
<td>105</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>100</td>
<td>99</td>
</tr>
</tbody>
</table>

Option 016 degrades performance by 2 dB.

---

\(^d\) The system dynamic range is calculated as the difference between the noise floor and the source maximum output power. System dynamic range is a specification when the source is set to Port 1, and a characteristic when the source is set to Port 2. The effective dynamic range must take measurement uncertainties and interfering signals into account as well as the insertion loss resulting from a thru cable connected between Port 1 and Port 2.
The system dynamic range is calculated as the difference between the noise floor and the source maximum output power. System dynamic range is a specification when the source is set to Port 1, and a characteristic when the source is set to Port 2. The effective dynamic range must take measurement uncertainties and interfering signals into account as well as the insertion loss resulting from a thru cable connected between Port 1 and Port 2.

The test port system dynamic range is calculated as the difference between the test port noise floor and the source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account as well as the insertion loss resulting from a thru cable connected between Port 1 and Port 2.

The direct receiver access input system dynamic range is calculated as the difference between the receiver access input noise floor and the source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account. This set-up should only be used when the receiver input will never exceed its damage level. When the analyzer is in segment sweep mode, the analyzer can have predefined frequency segments which will output a higher power level when the extended dynamic range is required (i.e. devices with high insertion loss), and reduced power when receiver damage may occur (i.e. devices with low insertion loss). The extended range is only available in one-path transmission measurements.

d Typical performance.

e May be limited to 100 dB at particular frequencies below 500 MHz due to spurious receiver residuals. Methods are available to regain the full dynamic range.

f E8362B only: Option H11 decreases value by 1 dB.

g E8362B only: Option H11 decreases value by 2 dB.

Table 2. Receiver Dynamic Range

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB) at Test Port</th>
<th>Typical (dB) at Direct Receiver Access Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic Range (in a 10 Hz BW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard Configuration and Standard Power Range (E836xB - Standard) OR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard Configuration and Extended Power Range &amp; Bias Tees (E836xB - Option UNL)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>82 (typical)</td>
<td>NA</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>94</td>
<td>NA</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>119</td>
<td>NA</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>122</td>
<td>NA</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>125</td>
<td>NA</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>114</td>
<td>NA</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>111</td>
<td>NA</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>111</td>
<td>NA</td>
</tr>
<tr>
<td>Configurable Test Set and Standard Power Range (E836xB - Option 014) OR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configurable Test Set and Extended Power Range &amp; Bias Tees (E836xB - Option 014/UNL)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>82 (typical)</td>
<td>132 (typical)</td>
</tr>
</tbody>
</table>

Note: Option 016 degrades performance by 2 dB.
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>94</th>
<th>132</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>45 MHz to 500 MHz</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>119</td>
<td>138</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>122</td>
<td>137</td>
<td>--</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>124</td>
<td>139</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>113</td>
<td>125</td>
<td>Option 016 degrades performance by 2 dB.</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>110</td>
<td>122</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>109</td>
<td>120</td>
<td></td>
</tr>
</tbody>
</table>

- The receiver dynamic range is calculated as the difference between the noise floor and the receiver maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account.
- The test port receiver dynamic range is calculated as the difference between the test port noise floor and the receiver maximum input level. The effective dynamic range must take measurement uncertainties and interfering signals into account.
- The direct receiver access input receiver dynamic range is calculated as the difference between the direct receiver access input noise floor and the receiver maximum input level. The effective dynamic range must take measurement uncertainties and interfering signals into account. This set-up should only be used when the receiver input will never exceed its compression or damage level. When the analyzer is in segment sweep mode, the analyzer can have predefined frequency segments which will output a higher power level when the extended dynamic range is required (i.e. devices with high insertion loss), and reduced power when compression or receiver damage may occur (i.e. devices with low insertion loss). The extended range is only available in one-path transmission measurements.
- Typical performance.
- May be degraded by 10 dB at particular frequencies (multiples of 5 MHz) below 500 MHz due to spurious receiver residuals. Methods are available to regain the full dynamic range.

**Note:** This E836xB document provides technical specifications for the following calibration kits only: 85056A, 85056D, 85056K, 85052B, 85052C, 85052D, 85050B, 85050C, 85050D, 85054B, 85054D, K11644A, P11644A, R11644A, and the X11644A.

### Table 33. Uncorrected System Performance

Specifications apply over environmental temperature of 23° ±3 °C, with < 1 °C deviation from the calibration temperature.

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directivity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>11 db (typical)</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>24 dB</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>22 dB</td>
<td>--</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>16 dB</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>16 dB</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>15 dB</td>
<td>--</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>13 dB</td>
<td>--</td>
</tr>
</tbody>
</table>

**Source Match - Standard**
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Source Match - Opt UNL, 014 or 014/UNL</th>
<th>Load Match - Standard</th>
<th>Load Match - Opt UNL, 014 or 014/UNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz&lt;sup&gt;b&lt;/sup&gt;</td>
<td>11 dB (typical)</td>
<td>11 dB (typical)</td>
<td>11 dB (typical)</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>23 dB</td>
<td>23 dB</td>
<td>23 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>16 dB</td>
<td>14 dB</td>
<td>14 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>14 dB</td>
<td>12 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>10 dB</td>
<td>9 dB</td>
<td>9 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>9 dB</td>
<td>8 dB</td>
<td>8 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>7.5 dB</td>
<td>6 dB</td>
<td>6 dB</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Reflection Tracking</td>
<td>Typical:</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>±1.5 dB</td>
<td></td>
</tr>
<tr>
<td>45 MHz to 20 GHz</td>
<td>--</td>
<td>±1.5 dB</td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>±1.5 dB</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>±2.0 dB</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Transmission Tracking</th>
<th>Typical:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>±3.0 dB</td>
</tr>
<tr>
<td>45 MHz to 2 GHz</td>
<td>--</td>
<td>±1.5 dB</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>--</td>
<td>±2.0 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>--</td>
<td>±2.5 dB</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>±3.5 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>--</td>
<td>±4.0 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>--</td>
<td>±4.5 dB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Crosstalk</th>
<th>Typical:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>-65 dB (typical)</td>
</tr>
<tr>
<td>45 MHz to 1 GHz</td>
<td>-85 dB</td>
<td>--</td>
</tr>
<tr>
<td>1 GHz to 2 GHz</td>
<td>-100 dB</td>
<td>--</td>
</tr>
<tr>
<td>Frequency Band</td>
<td>Crosstalk (dB)</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>2 GHz to 20 GHz</td>
<td>-110 dB</td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>-108 dB</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>-105 dB</td>
<td></td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>-100 dB</td>
<td></td>
</tr>
</tbody>
</table>

**Crosstalkd - Option UNL or 014**

<table>
<thead>
<tr>
<th>Frequency Band</th>
<th>Crosstalk (dB)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>-65 dB (typical)</td>
</tr>
<tr>
<td>45 MHz to 1 GHz</td>
<td>-85 dB</td>
<td></td>
</tr>
<tr>
<td>1 GHz to 2 GHz</td>
<td>-100 dB</td>
<td></td>
</tr>
<tr>
<td>2 GHz to 20 GHz</td>
<td>-109 dB</td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>-106 dB</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>-103 dB</td>
<td></td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>-98 dB</td>
<td></td>
</tr>
</tbody>
</table>

**Crosstalkd - Option 014/UNL**

<table>
<thead>
<tr>
<th>Frequency Band</th>
<th>Crosstalk (dB)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>-65 dB (typical)</td>
</tr>
<tr>
<td>45 MHz to 1 GHz</td>
<td>-85 dB</td>
<td></td>
</tr>
<tr>
<td>1 GHz to 2 GHz</td>
<td>-98 dB</td>
<td></td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>-108 dB</td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>-107 dB</td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>-104 dB</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>-100 dB</td>
<td></td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>-95 dB</td>
<td></td>
</tr>
</tbody>
</table>

**Crosstalk - Option 080 enabled**

<table>
<thead>
<tr>
<th>Frequency Band</th>
<th>Crosstalk (dB)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>-65</td>
</tr>
<tr>
<td>45 MHz to 1 GHz</td>
<td>--</td>
<td>-85</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Standard</td>
<td>Opt 014</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------</td>
<td>---------</td>
</tr>
<tr>
<td>1 GHz to 2 GHz</td>
<td>--</td>
<td>-100</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>--</td>
<td>-109</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>--</td>
<td>-110</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>-106</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>--</td>
<td>-103</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>--</td>
<td>-98</td>
</tr>
</tbody>
</table>

a Specifications apply over environment temperature of 23°C +/- 3°C, with less than 1°C deviation from the calibration temperature.
b Typical performance.
c Transmission tracking performance is strongly dependent on cable used. These typical specifications are based on the use of the Agilent thru cable (part number 85133-60016).
d Measurement conditions: normalized to a thru, measured with two shorts, 10 Hz IF bandwidth, averaging factor of 16, alternate mode, source power set to the lesser of the maximum power out or the maximum receiver power.
e 0 Hz offset.

Table 34. Test Port Output

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Frequency Range</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E8362B</td>
<td>10 MHz to 20 GHz</td>
<td>--</td>
</tr>
<tr>
<td>E8363B</td>
<td>10 MHz to 40 GHz</td>
<td>--</td>
</tr>
<tr>
<td>E8364B</td>
<td>10 MHz to 50 GHz</td>
<td>--</td>
</tr>
<tr>
<td><strong>Nominal Power(^{C})</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E8362B</td>
<td>0 dBm</td>
<td>-5 dBm</td>
</tr>
<tr>
<td>E8363/4B</td>
<td>-12 dBm</td>
<td>-17 dBm</td>
</tr>
<tr>
<td><strong>Frequency Resolution</strong></td>
<td>1 Hz</td>
<td>--</td>
</tr>
<tr>
<td><strong>CW Accuracy</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequency Stability</td>
<td>+/−1 ppm</td>
<td>--</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------</td>
<td>----</td>
</tr>
<tr>
<td><strong>Power Level Accuracy</strong>&lt;sup&gt;a&lt;/sup&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz&lt;sup&gt;b&lt;/sup&gt;</td>
<td>+/−2.0 dB (typical)</td>
<td>+/−2.0 dB (typical)</td>
</tr>
<tr>
<td>45 MHz to 10 GHz</td>
<td>+/−1.5 dB</td>
<td>+/−1.5 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>+/−2.0 dB</td>
<td>+/−2.0 dB</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>+/−3.0 dB</td>
<td>+/−3.0 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>+/−3.0 dB</td>
<td>+/−3.5 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>+/−3.0 dB</td>
<td>+/−4.0 dB</td>
</tr>
<tr>
<td><strong>Power Level Linearity</strong>&lt;sup&gt;d&lt;/sup&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz&lt;sup&gt;b&lt;/sup&gt;</td>
<td>+/−1.0 dB&lt;sup&gt;g&lt;/sup&gt; (typical)</td>
<td>Test reference is at the nominal power level (step attenuator at 0 dB)</td>
</tr>
<tr>
<td>45 MHz to 20 GHz</td>
<td>+/−1.0 dB&lt;sup&gt;g&lt;/sup&gt;</td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>+/−1.0 dB&lt;sup&gt;g&lt;/sup&gt;</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>+/−1.0 dB&lt;sup&gt;g&lt;/sup&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>Power Range</strong>&lt;sup&gt;a, e, f&lt;/sup&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz&lt;sup&gt;b&lt;/sup&gt;</td>
<td>-25 to +2 dBm (typical)</td>
<td>-25 to +2 dBm (typical)</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>-25 to +5 dBm</td>
<td>-25 to +5 dBm</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td>45 MHz to 10 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>-24 to +3 dBm</td>
<td>-25 to +2 dBm</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>-23 to 0 dBm</td>
<td>-25 to -2 dBm</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>-23 to -4 dBm</td>
<td>-25 to -6 dBm</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>-25 to -5 dBm</td>
<td>-27 to -7 dBm</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>-25 to -10 dBm</td>
<td>-27 to -12 dBm</td>
</tr>
</tbody>
</table>

**Power Sweep Range (ALC)**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>27 dB (typical)</th>
<th>27 dB (typical)</th>
<th>29 dB (typical)</th>
<th>29 dB (typical)</th>
<th>--</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 10 GHz</td>
<td>30 dB</td>
<td>30 dB</td>
<td>30 dB</td>
<td>30 dB</td>
<td>30 dB</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>27 dB</td>
<td>27 dB</td>
<td>27 dB</td>
<td>27 dB</td>
<td>27 dB</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>23 dB</td>
<td>23 dB</td>
<td>23 dB</td>
<td>23 dB</td>
<td>23 dB</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>19 dB</td>
<td>19 dB</td>
<td>19 dB</td>
<td>19 dB</td>
<td>19 dB</td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>20 dB</td>
<td>20 dB</td>
<td>18 dB</td>
<td>16 dB</td>
<td>16 dB</td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>15 dB</td>
<td>15 dB</td>
<td>12 dB</td>
<td>10 dB</td>
<td>10 dB</td>
</tr>
</tbody>
</table>

**Power Resolution**

|                     | 0.01 dB | -- |

**Phase Noise**

<table>
<thead>
<tr>
<th>1 kHz offset from center frequency, nominal power at test port</th>
<th>--</th>
<th>--</th>
<th>--</th>
<th>--</th>
<th>Typical:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 10 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-60 dBc</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>-55 dBc</td>
<td>-50 dBc</td>
<td>-50 dBc</td>
<td>-55 dBc</td>
<td>-60 dBc</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 GHz to 50 GHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 kHz offset from center frequency, nominal power at test port - Option 080 enabled

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Typical</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 10 GHz</td>
<td>-60 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>-60 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 GHz to 50 GHz</td>
<td>-50 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10 kHz offset from center frequency, nominal power at test port

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Typical</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>-70 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45 MHz to 10 GHz</td>
<td>-70 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>-65 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>-55 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>-55 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10 kHz offset from center frequency, nominal power at test port - Option 080 enabled

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Typical</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>-70 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>45 MHz to 10 GHz</td>
<td>-70 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>-65 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>-55 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>-55 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

100 kHz offset from center frequency, nominal power at test port

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Typical</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 10 GHz</td>
<td>-60 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>-55 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequency Range</td>
<td>1 MHz Offset</td>
<td>100 kHz Offset</td>
<td>Typical:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------</td>
<td>----------------</td>
<td>----------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>-50 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 10 GHz</td>
<td>--</td>
<td>--</td>
<td>-75 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
<td>-70 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>-65 dBc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**1 MHz offset from center frequency, nominal power at test port**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>1 MHz Offset</th>
<th>100 kHz Offset</th>
<th>Typical:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 10 GHz</td>
<td>--</td>
<td>--</td>
<td>-106 dBc</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
<td>-103 dBc</td>
</tr>
<tr>
<td>20 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>-90 dBc</td>
</tr>
</tbody>
</table>

**1 MHz offset from center frequency, nominal power at test port - Option 080 enabled**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>1 MHz Offset</th>
<th>100 kHz Offset</th>
<th>Typical:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 10 GHz</td>
<td>--</td>
<td>--</td>
<td>-103 dBc</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
<td>-97 dBc</td>
</tr>
<tr>
<td>20 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>-85 dBc</td>
</tr>
</tbody>
</table>

**Harmonics (2nd or 3rd)**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>1 MHz Offset</th>
<th>100 kHz Offset</th>
<th>Typical:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>-23 dBc typical, in power range 0</td>
</tr>
</tbody>
</table>

**Non-Harmonic Spurious (at Nominal Output Power)**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>1 MHz Offset</th>
<th>100 kHz Offset</th>
<th>Typical:</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
<td>-50 dBc typical, for offset frequency &gt; 1 kHz</td>
</tr>
<tr>
<td>Description</td>
<td>Specification</td>
<td>Supplemental</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>---------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td><strong>Test Port Noise Floor</strong>&lt;sup&gt;a&lt;/sup&gt;</td>
<td>Standard</td>
<td>Opt 014</td>
<td>Opt UNL</td>
</tr>
<tr>
<td><strong>10 Hz IF Bandwidth</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz&lt;sup&gt;b&lt;/sup&gt;</td>
<td>&lt;-77 dBm (typical)</td>
<td>&lt;-77 dBm (typical)</td>
<td>&lt;-77 dBm (typical)</td>
</tr>
<tr>
<td>45 MHz to 500 MHz&lt;sup&gt;c&lt;/sup&gt;</td>
<td>&lt;-89 dBm</td>
<td>&lt;-89 dBm</td>
<td>&lt;-89 dBm</td>
</tr>
</tbody>
</table>

---

a Test port output is a specification when the source is set to Port 1, and a characteristic when the source is set to Port 2.
b Typical performance.
c Preset power.
d Power Level Linearity is a specification when the source is set to Port 1, and a typical when the source is set to Port 2.
e Test port power is specified into nominal 50 ohms.
f Power to which the source can be set and phase lock is assured.
g +/-1.5 dB for power <= -23 dBm.
h E8362B only: Option H11 decreases maximum power level by 1 dB.
i E8362B only: Option H11 decreases maximum power level by 2 dB.
j E8362B only: Option H11 decreases power level by 1 dB.
k E8362B only: Option H11 decreases power level by 2 dB.

---

<table>
<thead>
<tr>
<th></th>
<th>45 MHz to 20 GHz</th>
<th>20 GHz to 40 GHz</th>
<th>40 GHz to 50 GHz</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>-50 dBc typical, for offset frequency &gt; 1 kHz</td>
<td>-30 dBc typical, for offset frequency &gt; 1 kHz</td>
<td>-30 dBc typical, for offset frequency &gt; 1 kHz</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>1 kHz IF Bandwidth</td>
<td>10 kHz IF Bandwidth</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>-14 dBm</td>
<td>-14 dBm</td>
<td></td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>-17 dBm</td>
<td>-17 dBm</td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>-20 dBm</td>
<td>-20 dBm</td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>-14 dBm</td>
<td>-13 dBm</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>-14 dBm</td>
<td>-12 dBm</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>-57 dBm (typical)</td>
<td>-57 dBm (typical)</td>
<td></td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>-69 dBm</td>
<td>-69 dBm</td>
<td></td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>-94 dBm</td>
<td>-94 dBm</td>
<td></td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>-97 dBm</td>
<td>-97 dBm</td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>-100 dBm</td>
<td>-100 dBm</td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>-94 dBm</td>
<td>-93 dBm</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>-94 dBm</td>
<td>-92 dBm</td>
<td></td>
</tr>
</tbody>
</table>

**Test Port Noise Floor**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>10 Hz IF Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>-77 dBm (typical)</td>
</tr>
</tbody>
</table>

---

1. **1 KHz IF Bandwidth**
2. **10 kHz IF Bandwidth**
3. **Test Port Noise Floor**
4. **Option 080 enabled**

Option 016 degrades performance by 2 dB.
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Noise Floor (-dBm)</th>
<th>Frequency Range</th>
<th>Noise Floor (-dBm)</th>
<th>Frequency Range</th>
<th>Noise Floor (-dBm)</th>
<th>Frequency Range</th>
<th>Noise Floor (-dBm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;-88 dBm</td>
<td>500 MHz to 2 GHz</td>
<td>&lt;-113 dBm</td>
<td>2 GHz to 10 GHz</td>
<td>&lt;-116 dBm</td>
<td>10 GHz to 20 GHz</td>
<td>&lt;-118 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>&lt;-116 dBm</td>
<td>10 GHz to 20 GHz</td>
<td>&lt;-118 dBm</td>
<td>20 GHz to 40 GHz</td>
<td>&lt;-112 dBm</td>
<td>40 GHz to 50 GHz</td>
<td>&lt;-111 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>&lt;-113 dBm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>&lt;-116 dBm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>&lt;-118 dBm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>&lt;-112 dBm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Option 016 degrades performance by 2 dB.</td>
</tr>
<tr>
<td>Direct Receiver Access Input Noise Floor <strong>a</strong>, <strong>b</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Hz IF Bandwidth</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Option 016 degrades performance by 2 dB.**
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>IF Bandwidth</th>
<th>Sensitivity</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>1 KHz IF Bandwidth</td>
<td>--</td>
<td>&lt;-127 dBm</td>
<td>--</td>
<td>&lt;-127 dBm</td>
</tr>
<tr>
<td>45 MHz to 500 MHz&lt;sup&gt;C&lt;/sup&gt;</td>
<td>1 KHz IF Bandwidth</td>
<td>--</td>
<td>&lt;-127 dBm</td>
<td>--</td>
<td>&lt;-127 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>1 KHz IF Bandwidth</td>
<td>--</td>
<td>&lt;-133 dBm</td>
<td>--</td>
<td>&lt;-133 dBm</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>1 KHz IF Bandwidth</td>
<td>--</td>
<td>&lt;-132 dBm</td>
<td>--</td>
<td>&lt;-132 dBm</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>1 KHz IF Bandwidth</td>
<td>--</td>
<td>&lt;-134 dBm</td>
<td>--</td>
<td>&lt;-134 dBm</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>1 KHz IF Bandwidth</td>
<td>--</td>
<td>&lt;-125 dBm</td>
<td>--</td>
<td>&lt;-125 dBm</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>1 KHz IF Bandwidth</td>
<td>--</td>
<td>&lt;-123 dBm</td>
<td>--</td>
<td>&lt;-123 dBm</td>
</tr>
</tbody>
</table>

Option 016 degrades performance by 2 dB.
### Direct Receiver Access Input Noise Floor\textsuperscript{a,b} - Option 080 enabled\textsuperscript{d}

#### 10 Hz IF Bandwidth

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>10 MHz to 45 MHz</th>
<th>45 MHz to 500 MHz\textsuperscript{c}</th>
<th>500 MHz to 2 GHz</th>
<th>2 GHz to 10 GHz</th>
<th>10 GHz to 20 GHz</th>
<th>20 GHz to 40 GHz</th>
<th>40 GHz to 50 GHz</th>
<th>40 GHz to 50 GHz</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>--</td>
<td>&lt;-127 dBm</td>
<td>--</td>
<td>&lt;-127 dBm</td>
<td>--</td>
<td>&lt;-127 dBm</td>
<td>--</td>
<td>&lt;-127 dBm</td>
</tr>
<tr>
<td>45 MHz to 500 MHz\textsuperscript{c}</td>
<td>--</td>
<td>&lt;-126 dBm</td>
<td>--</td>
<td>&lt;-126 dBm</td>
<td>--</td>
<td>&lt;-126 dBm</td>
<td>--</td>
<td>&lt;-126 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>--</td>
<td>&lt;-132 dBm</td>
<td>--</td>
<td>&lt;-132 dBm</td>
<td>--</td>
<td>&lt;-132 dBm</td>
<td>--</td>
<td>&lt;-132 dBm</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>--</td>
<td>&lt;-131 dBm</td>
<td>--</td>
<td>&lt;-131 dBm</td>
<td>--</td>
<td>&lt;-131 dBm</td>
<td>--</td>
<td>&lt;-131 dBm</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>--</td>
<td>&lt;-133 dBm</td>
<td>--</td>
<td>&lt;-133 dBm</td>
<td>--</td>
<td>&lt;-133 dBm</td>
<td>--</td>
<td>&lt;-133 dBm</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>&lt;-124 dBm</td>
<td>--</td>
<td>&lt;-124 dBm</td>
<td>--</td>
<td>&lt;-124 dBm</td>
<td>--</td>
<td>&lt;-124 dBm</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>&lt;-122 dBm</td>
<td>--</td>
<td>&lt;-122 dBm</td>
<td>--</td>
<td>&lt;-122 dBm</td>
<td>--</td>
<td>&lt;-122 dBm</td>
</tr>
</tbody>
</table>

Option 016 degrades performance by 2 dB.

#### 1 KHz IF Bandwidth

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>10 MHz to 45 MHz</th>
<th>45 MHz to 500 MHz\textsuperscript{c}</th>
<th>500 MHz to 2 GHz</th>
<th>2 GHz to 10 GHz</th>
<th>10 GHz to 20 GHz</th>
<th>20 GHz to 40 GHz</th>
<th>40 GHz to 50 GHz</th>
<th>40 GHz to 50 GHz</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>--</td>
<td>&lt;-107 dBm</td>
<td>--</td>
<td>&lt;-107 dBm</td>
<td>--</td>
<td>&lt;-107 dBm</td>
<td>--</td>
<td>&lt;-107 dBm</td>
</tr>
<tr>
<td>45 MHz to 500 MHz\textsuperscript{c}</td>
<td>--</td>
<td>&lt;-106 dBm</td>
<td>--</td>
<td>&lt;-106 dBm</td>
<td>--</td>
<td>&lt;-106 dBm</td>
<td>--</td>
<td>&lt;-106 dBm</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>--</td>
<td>&lt;-112 dBm</td>
<td>--</td>
<td>&lt;-112 dBm</td>
<td>--</td>
<td>&lt;-112 dBm</td>
<td>--</td>
<td>&lt;-112 dBm</td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>--</td>
<td>&lt;-111 dBm</td>
<td>--</td>
<td>&lt;-111 dBm</td>
<td>--</td>
<td>&lt;-111 dBm</td>
<td>--</td>
<td>&lt;-111 dBm</td>
</tr>
<tr>
<td>10 GHz to 20 GHz</td>
<td>--</td>
<td>&lt;-113 dBm</td>
<td>--</td>
<td>&lt;-113 dBm</td>
<td>--</td>
<td>&lt;-113 dBm</td>
<td>--</td>
<td>&lt;-113 dBm</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>&lt;-104 dBm</td>
<td>--</td>
<td>&lt;-104 dBm</td>
<td>--</td>
<td>&lt;-104 dBm</td>
<td>--</td>
<td>&lt;-104 dBm</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>&lt;-102 dBm</td>
<td>--</td>
<td>&lt;-102 dBm</td>
<td>--</td>
<td>&lt;-102 dBm</td>
<td>--</td>
<td>&lt;-102 dBm</td>
</tr>
</tbody>
</table>

Option 016 degrades performance by 2 dB.
### Receiver Compression Level

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Compression Level</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 20 GHz</td>
<td>&lt;0.1 dB at -5 dBm and &lt;0.45 dB at +5 dBm</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 30 GHz</td>
<td>&lt;0.1 dB at -9.5 dBm and &lt;0.45 dB at 0 dBm</td>
<td>--</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>&lt;0.1 dB at -12.5 dBm and &lt;0.45 dB at -3 dBm</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>&lt;0.1 dB at -12.5 dBm and &lt;0.45 dB at -3 dBm</td>
<td>--</td>
</tr>
</tbody>
</table>

### System Compression Level

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Compression Level</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>maximum output power</td>
<td>See dynamic accuracy table</td>
</tr>
</tbody>
</table>

### Third Order Intercept - Tone spacing from 100 kHz - 5 MHz

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Compression Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 150 MHz</td>
<td>--</td>
</tr>
<tr>
<td>150 MHz to 300 MHz</td>
<td>--</td>
</tr>
<tr>
<td>300 MHz to 500 MHz</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 20 GHz</td>
<td>--</td>
</tr>
<tr>
<td>20 to 40 GHz</td>
<td>--</td>
</tr>
<tr>
<td>40 to 50 GHz</td>
<td>--</td>
</tr>
</tbody>
</table>

#### Typical

- +33 dBm
- +34 dBm
- +30 dBm
- +24 dBm
- +18 dBm
- +15 dBm

### Third Order Intercept - Tone spacing from 5 MHz - 20 MHz

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Compression Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 500 MHz</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 20 GHz</td>
<td>--</td>
</tr>
</tbody>
</table>

#### Typical

- +20 dBm
- +20 dBm
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Third Order Intercept</th>
<th>Trace Noise Magnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 to 40 GHz</td>
<td>--</td>
<td>+16 dBm</td>
</tr>
<tr>
<td>40 to 50 GHz</td>
<td>--</td>
<td>+15 dBm</td>
</tr>
<tr>
<td>20 MHz to 500 MHz</td>
<td>--</td>
<td>+26 dBm</td>
</tr>
<tr>
<td>500 MHz to 20 GHz</td>
<td>--</td>
<td>+26 dBm</td>
</tr>
<tr>
<td>20 MHz to 40 GHz</td>
<td>--</td>
<td>+20 dBm</td>
</tr>
<tr>
<td>40 MHz to 50 GHz</td>
<td>--</td>
<td>+19 dBm</td>
</tr>
</tbody>
</table>

**Third Order Intercept - Tone spacing from 20 MHz - 50 MHz**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 MHz to 45 MHz</td>
<td>&lt;0.050 dB rms (typical)</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;0.010 dB rms</td>
</tr>
<tr>
<td>500 MHz to 20 GHz</td>
<td>&lt;0.006 dB rms</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>&lt;0.006 dB rms</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>&lt;0.006 dB rms</td>
</tr>
</tbody>
</table>

**Trace Noise Magnitude**

1 kHz IF bandwidth. Ratio measurement, nominal power at test port.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Trace Noise Magnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;0.010 dB rms</td>
</tr>
<tr>
<td>500 MHz to 20 GHz</td>
<td>&lt;0.006 dB rms</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>&lt;0.006 dB rms</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>&lt;0.006 dB rms</td>
</tr>
</tbody>
</table>

**Trace Noise Magnitude - Option 080 enabled**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Trace Noise Magnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;0.010 dB rms</td>
</tr>
</tbody>
</table>

2851
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>RMS Phase Error</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>500 MHz to 20 GHz</td>
<td>&lt;0.006 dB rms</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>&lt;0.007 dB rms</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>&lt;0.008 dB rms</td>
<td>--</td>
</tr>
<tr>
<td><strong>Trace Noise Phase</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 kHz IF bandwidth. Ratio measurement, nominal power at test port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>&lt;0.350° rms (typical)</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;0.100° rms</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 20 GHz</td>
<td>&lt;0.060° rms</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>&lt;0.100° rms</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>&lt;0.100° rms</td>
<td>--</td>
</tr>
<tr>
<td><strong>Trace Noise Phase - Option 080 enabled</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 kHz IF bandwidth. Ratio measurement, nominal power at test port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>&lt;0.350° rms</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>&lt;0.100° rms</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 20 GHz</td>
<td>&lt;0.060° rms</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>&lt;0.100° rms</td>
<td>--</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Reference Level Magnitude</td>
<td>Stability Magnitude&lt;sup&gt;d&lt;/sup&gt;</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>&lt;0.100° rms</td>
<td>--</td>
</tr>
</tbody>
</table>

**Reference Level Magnitude**

<table>
<thead>
<tr>
<th></th>
<th>Range</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+/-200 dB</td>
<td>0.001 dB</td>
</tr>
</tbody>
</table>

**Reference Level Phase**

<table>
<thead>
<tr>
<th></th>
<th>Range</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+/-500°</td>
<td>0.01°</td>
</tr>
</tbody>
</table>

**Stability Magnitude<sup>d</sup>**

Typical ratio measurement, made at the test port.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Stability Magnitude&lt;sup&gt;d&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>+/-0.05 dB/°C</td>
</tr>
<tr>
<td>45 MHz to 20 GHz</td>
<td>+/-0.02 dB/°C</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>+/-0.03 dB/°C</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>+/-0.04 dB/°C</td>
</tr>
</tbody>
</table>

**Stability Phase<sup>d</sup>**

Typical ratio measurement, measured at the test port.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Stability Phase&lt;sup&gt;d&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>+/-0.5°/°C</td>
</tr>
<tr>
<td>45 MHz to 20 GHz</td>
<td>+/-0.2°/°C</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>+/-0.5°/°C</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>+/-0.8°/°C</td>
</tr>
</tbody>
</table>
### Damage Input Level

<table>
<thead>
<tr>
<th></th>
<th>Test Port 1 and 2</th>
<th>R1, R2 in</th>
<th>A, B in</th>
<th>Coupler Thru (Option 014 or UNL/014)</th>
<th>Coupler Arm (Option 014 or UNL/014)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>+30 dBm or +/-40 VDC, typical</td>
<td>+30 dBm or +/-40 VDC, typical</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>+15 dBm or +/-15 VDC, typical</td>
<td>+15 dBm or +/-15 VDC, typical</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>+30 dBm or +/-40 VDC, typical</td>
<td>+30 dBm or +/-7 VDC, typical</td>
</tr>
</tbody>
</table>

a Total average (rms) noise power calculated as the mean value of a linear magnitude trace expressed in dBm.

b Typical performance.

c Noise floor may be degraded by 10 dB at particular frequencies (multiples of 5 MHz) due to spurious receiver residuals.

d 0 Hz offset

e Trace noise magnitude may be degraded to 20 mDB rms at harmonic frequencies of the first IF (8.33 MHz) below 80 MHz.

f Stability is defined as a ratio measurement made at the test port.

g This compression level comes from the dynamic accuracy curve with -30 dBm reference test port power.

**Table 36. Dynamic Accuracy (Specification)**

Accuracy of the test port input power reading relative to the reference input power level.
Below 800 MHz the coupling factor rolls off 20 dB per decade causing a shift in the dynamic accuracy curves. Please see the Uncertainty Calculator (http://www.agilent.com/find/na_calculator) for detailed compression values.
Dynamic accuracy is verified with the following measurements:

- Compression over frequency.
- IF linearity at a single frequency of 1.195 GHz and a reference level of -20 dBm for an input power range of 0 to -120 dBm.

### Table 37. Test Port Input (Group Delay)\(^a\)
<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental Information (typ.)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Aperture (selectable)</strong></td>
<td>--</td>
<td>(frequency span)/(number of points -1)</td>
</tr>
<tr>
<td><strong>Maximum Aperture</strong></td>
<td>--</td>
<td>20% of frequency span</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>--</td>
<td>0.5 x (1/minimum aperture)</td>
</tr>
<tr>
<td><strong>Maximum Delay</strong></td>
<td>--</td>
<td>Limited to measuring no more than 180° of phase change within the minimum aperture.)</td>
</tr>
<tr>
<td><strong>Accuracy</strong></td>
<td>--</td>
<td>See graph below. Char.</td>
</tr>
</tbody>
</table>

The following graph shows characteristic group delay accuracy with full 2-port calibration and a 10 Hz IF bandwidth. Insertion loss is assumed to be < 2 dB and electrical length to be ten meters.

![Group Delay (Typical)](image)

In general, the following formula can be used to determine the accuracy, in seconds, of specific group delay measurement:

\[
\pm \text{Phase Accuracy (deg)}/\left[360 \times \text{Aperture (Hz)}\right]
\]

Depending on the aperture and device length, the phase accuracy used is either incremental phase accuracy or worst case phase accuracy.

**General Information**

- **Miscellaneous Information**
- **Front Panel**
- **Rear Panel**
### Environment and Dimensions

#### Table 38. Miscellaneous Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>System IF Bandwidth Range</td>
<td>--</td>
<td>1 Hz to 40 kHz, nominal</td>
</tr>
<tr>
<td>CPU</td>
<td>--</td>
<td>Intel® 1.1 GHz Pentium® M with 1 GByte RAM</td>
</tr>
</tbody>
</table>

#### Table 39. Front Panel Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RFC Connectors</strong></td>
<td></td>
</tr>
<tr>
<td>E8362B</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>3.5 mm (male), 50 ohm, (nominal)</td>
</tr>
<tr>
<td>Center Pin Recession</td>
<td>0.002 in. (characteristic)</td>
</tr>
<tr>
<td>E8363/4B</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>2.4 mm (male), 50 ohm, (nominal)</td>
</tr>
<tr>
<td>Center Pin Recession</td>
<td>0.002 in. (characteristic)</td>
</tr>
</tbody>
</table>

**Display**

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>21.3 cm (8.4 in) diagonal color active matrix LCD; 640 (horizontal) X 480 (vertical) resolution; 59.83 Hz vertical refresh rate; 31.41 Hz horizontal refresh rate</td>
</tr>
<tr>
<td>Refresh Rate</td>
<td>Vertical 59.83 Hz; Horizontal 31.41 kHz</td>
</tr>
</tbody>
</table>

**Display Range**

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magnitude</td>
<td>±200 dB (at 20 dB/div), max</td>
</tr>
<tr>
<td>Phase</td>
<td>±500°, max</td>
</tr>
<tr>
<td>Polar</td>
<td>10 pUnits, min 1000 Units, max</td>
</tr>
</tbody>
</table>
| Pixels      | When running the analyzer's built-in Display Test, one or more of the following symptoms indicate a faulty display assembly:  
  - A complete row or column of "stuck on" or "dark" pixels. |
• More than six "stuck on" pixels (but not more than three green)
• More than twelve "dark" pixels (but not more than seven of the same color)
• Two or more consecutive "stuck on" pixels or three or more consecutive "dark" pixels (but no more than one set of two consecutive 'dark' pixels)
• "Stuck on" or "dark" pixels less than 6.5 mm apart (excluding consecutive pixels)

<table>
<thead>
<tr>
<th>Display Resolution</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Magnitude</td>
<td>0.001 dB/div, min</td>
</tr>
<tr>
<td>Phase</td>
<td>0.01°/div, min</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Marker Resolution</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Magnitude</td>
<td>0.001 dB, min</td>
</tr>
<tr>
<td>Phase</td>
<td>0.01°, min</td>
</tr>
<tr>
<td>Polar</td>
<td>0.01 mUnit, min; 0.01°, min</td>
</tr>
</tbody>
</table>

**Table 40. Rear Panel Information**

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz Reference In</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Connector</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Input Frequency</td>
<td>10 MHz ± 10 ppm, typical</td>
</tr>
<tr>
<td>Input Level</td>
<td>-15 dBm to +20 dBm, typical</td>
</tr>
<tr>
<td>Input Impedance</td>
<td>200 W, nom.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz Reference Out</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Connector</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Output Frequency</td>
<td>10 MHz ± 1 ppm, typical</td>
</tr>
<tr>
<td>Signal Type</td>
<td>Sine Wave, typical</td>
</tr>
<tr>
<td>Output Level</td>
<td>+10 dBm ± 4 dB into 50 W, typical</td>
</tr>
<tr>
<td>Output Impedance</td>
<td>50 W, nominal</td>
</tr>
<tr>
<td>Harmonics</td>
<td>&lt;-40 dBc, typical</td>
</tr>
<tr>
<td>Option H08 &amp; H11 Rear Panel Connectors (typical)</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>IF Connectors</strong></td>
<td></td>
</tr>
<tr>
<td>A, R1, R2, B (BNC Connectors)</td>
<td></td>
</tr>
<tr>
<td><strong>IF Connector Input Frequency</strong></td>
<td></td>
</tr>
<tr>
<td>8 1/3 MHz</td>
<td></td>
</tr>
<tr>
<td><strong>Nominal Input Impedance at IF Inputs</strong></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
</tr>
<tr>
<td><strong>RF Damage Level to IF Connector Inputs</strong></td>
<td></td>
</tr>
<tr>
<td>-20.0 dBm</td>
<td></td>
</tr>
<tr>
<td><strong>DC Damage Level to IF Connector Inputs</strong></td>
<td></td>
</tr>
<tr>
<td>25 volts</td>
<td></td>
</tr>
<tr>
<td><strong>0.1 dB Compression Point at IF Inputs</strong></td>
<td></td>
</tr>
<tr>
<td>-27.0 dBm</td>
<td></td>
</tr>
<tr>
<td><strong>Pulse Input Connectors(^1)</strong></td>
<td></td>
</tr>
<tr>
<td>A, R1, R2, B (BNC Connectors)</td>
<td></td>
</tr>
<tr>
<td><strong>Nominal Input Impedance at Pulse Inputs</strong></td>
<td></td>
</tr>
<tr>
<td>1 Kohm</td>
<td></td>
</tr>
<tr>
<td><strong>Minimum IF Gate Width</strong></td>
<td></td>
</tr>
<tr>
<td>20 ns for less than 1 dB deviation from theoretical performance(^2)</td>
<td></td>
</tr>
<tr>
<td><strong>DC Damage Level to Pulse Connector Inputs</strong></td>
<td></td>
</tr>
<tr>
<td>5.5 volts</td>
<td></td>
</tr>
<tr>
<td><strong>Drive Voltage</strong></td>
<td></td>
</tr>
<tr>
<td>TTL (0, +5.0) Volts</td>
<td></td>
</tr>
<tr>
<td><strong>Rear Panel LO Power (Typical)</strong></td>
<td></td>
</tr>
<tr>
<td>1.7 GHz- 20 GHz</td>
<td></td>
</tr>
<tr>
<td>-7 to -16 dBm</td>
<td></td>
</tr>
<tr>
<td><strong>Rear Panel RF Power 8362B (Typical)</strong></td>
<td></td>
</tr>
<tr>
<td>1.7 GHz to 20 GHz</td>
<td></td>
</tr>
<tr>
<td>-5 to -16 dBm (at -5 dBm test port power(^3))</td>
<td></td>
</tr>
<tr>
<td><strong>Rear Panel RF Power 8363B/8364B (Typical)</strong></td>
<td></td>
</tr>
<tr>
<td>1.7 GHz to 10 GHz</td>
<td></td>
</tr>
<tr>
<td>-2 to -12 dBm (at -5 dBm test port power(^3))</td>
<td></td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td></td>
</tr>
<tr>
<td>0 to -8 dBm (at -5 dBm test port power(^3))</td>
<td></td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td></td>
</tr>
<tr>
<td>+5 to -1 dBm (at -5 dBm test port power(^3))</td>
<td></td>
</tr>
</tbody>
</table>
### VGA Video Output

<table>
<thead>
<tr>
<th>Connector</th>
<th>15-pin mini D-Sub; Drives VGA compatible monitors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Devices Supported:</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Resolutions:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flat Panel (TFT)</td>
</tr>
<tr>
<td>Flat Panel (DSTN)</td>
</tr>
<tr>
<td>CRT Monitor</td>
</tr>
</tbody>
</table>

Simultaneous operation of the internal and external displays is allowed, but with 640 X 480 resolution only. If you change resolution, you can only view the external display (internal display will "white out").

### Bias Input Connectors (Option UNL)

<table>
<thead>
<tr>
<th>Bias current</th>
<th>500 mA, maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bias voltage</td>
<td>40 Volts, maximum</td>
</tr>
</tbody>
</table>

### Test Set IO

25-pin D-Sub connector, available for external test set control.

### Aux IO

25-pin D-Sub connector, male, analog and digital IO.

### Handler IO

36-pin parallel I/O port; all input/output signals are default set to negative logic; can be reset to positive logic via GPIB command.

### GPIB

24-pin D-sub (Type D-24), female; compatible with IEEE-488.

### Parallel Port (LPT1)

25-pin D-Sub miniature connector, female; provides connection to printers or any other parallel port peripherals.

### Serial Port (COM 1)

9-pin D-Sub, male; compatible with RS-232
One port on front panel and five ports on rear panel. Universal Serial Bus jack, Type A configuration (4 contacts inline, contact 1 on left); female

<table>
<thead>
<tr>
<th>Contact</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contact 1</td>
<td>Vcc: 4.75 to 5.25 VDC, 500 mA, maximum</td>
</tr>
<tr>
<td>Contact 2</td>
<td>-Data</td>
</tr>
<tr>
<td>Contact 3</td>
<td>+Data</td>
</tr>
<tr>
<td>Contact 4</td>
<td>Ground</td>
</tr>
</tbody>
</table>

LAN

10/100BaseT Ethernet, 8-pin configuration; auto selects between the two data rates

Line Power

<table>
<thead>
<tr>
<th>Frequency, Voltage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>50/60/400 Hz for 100 - 120 V</td>
<td>50/60 Hz for 220 - 240 V</td>
</tr>
<tr>
<td>Power supply is auto switching</td>
<td>Max 350 Watts</td>
</tr>
</tbody>
</table>

1 Pulse input connectors are operational only with Option H08 (Pulse Measurement Capability) enabled.

2 Based on deviation from signal reduction equation: Signal Reduction (dB) = 20log10(Duty_cycle) = 20log10(pulse_width/period). Measured at Pulse Repetition Frequency (PFR) of 1 MHz.

3 Test port power has to be at a high enough level such that the Drop Cal does not occur. If Drop Cal occurs then the power out of the rear panel RF connector will drop by about 15 dB.

Table 41. Analyzer Dimensions and Weight

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
</table>

2864
### Cabinet Dimensions

<table>
<thead>
<tr>
<th>Description</th>
<th>Height</th>
<th>Width</th>
<th>Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excluding front and rear panel hardware and feet</td>
<td>267 mm</td>
<td>426 mm</td>
<td>427 mm</td>
</tr>
<tr>
<td></td>
<td>10.5 in</td>
<td>16.75 in</td>
<td>16.8 in</td>
</tr>
<tr>
<td>As shipped - includes front panel connectors, rear panel bumpers, and feet.</td>
<td>280 mm</td>
<td>435 mm</td>
<td>470 mm</td>
</tr>
<tr>
<td></td>
<td>11.0 in</td>
<td>17.1 in</td>
<td>18.5 in</td>
</tr>
<tr>
<td>As shipped plus handles</td>
<td>280 mm</td>
<td>458 mm</td>
<td>501 mm</td>
</tr>
<tr>
<td></td>
<td>11.0 in</td>
<td>18 in</td>
<td>19.70 in</td>
</tr>
<tr>
<td>As shipped plus rack-mount flanges</td>
<td>280 mm</td>
<td>483 mm</td>
<td>470 mm</td>
</tr>
<tr>
<td></td>
<td>11.0 in</td>
<td>19 in</td>
<td>18.5 in</td>
</tr>
<tr>
<td>As shipped plus handles and flanges</td>
<td>280 mm</td>
<td>483 mm</td>
<td>501 mm</td>
</tr>
<tr>
<td></td>
<td>11.0 in</td>
<td>19 in</td>
<td>19.70 in</td>
</tr>
</tbody>
</table>

### Weight

<table>
<thead>
<tr>
<th>Description</th>
<th>Net</th>
<th>Shipping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Net</td>
<td>E8362B</td>
<td>28.6 kg (63.5 lb), nominal</td>
</tr>
<tr>
<td></td>
<td>E8363/4B</td>
<td>29 kg (64 lb), nominal</td>
</tr>
<tr>
<td>Shipping</td>
<td>E8362B</td>
<td>35.8 kg (79.5 lb), nominal</td>
</tr>
<tr>
<td></td>
<td>E8363/4B</td>
<td>36.3 kg (80 lb), nominal</td>
</tr>
</tbody>
</table>

**Note:** For Regulatory and Environmental information, refer to the PNA Series Installation and Quick Start Guide, located online at http://cp.literature.agilent.com/litweb/pdf/E8356-90001.pdf.

### Measurement Throughput Summary

- **Typical Cycle Time for Measurement Completion**
- **Cycle Time vs IF Bandwidth**
- **Cycle Time vs Number of Points**
- **Data Transfer Time**

**Table 42** Typical Cycle Time\(^a,b\) (ms) for Measurement Completion
<table>
<thead>
<tr>
<th></th>
<th>Number of Points</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>201</td>
</tr>
<tr>
<td><strong>Start 28 GHz, Stop 30 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
</tr>
<tr>
<td>Uncorrected,</td>
<td>12</td>
</tr>
<tr>
<td>1-port cal</td>
<td></td>
</tr>
<tr>
<td>2-Port cal</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 10 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
</tr>
<tr>
<td>Uncorrected,</td>
<td>86</td>
</tr>
<tr>
<td>1-port cal</td>
<td></td>
</tr>
<tr>
<td>2-Port cal</td>
<td>179</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 20 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
</tr>
<tr>
<td>Uncorrected,</td>
<td>126</td>
</tr>
<tr>
<td>1-port cal</td>
<td></td>
</tr>
<tr>
<td>2-Port cal</td>
<td>264</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 40 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
</tr>
<tr>
<td>Uncorrected,</td>
<td>185</td>
</tr>
<tr>
<td>1-port cal</td>
<td></td>
</tr>
<tr>
<td>2-Port cal</td>
<td>382</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 50 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
</tr>
<tr>
<td>Uncorrected,</td>
<td>210</td>
</tr>
<tr>
<td>1-port cal</td>
<td></td>
</tr>
<tr>
<td>2-Port cal</td>
<td>436</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Start 10 MHz, Stop 67 GHz, 35 kHz IF bandwidth</strong></td>
<td></td>
</tr>
<tr>
<td>Uncorrected</td>
<td>244</td>
</tr>
<tr>
<td>1-Port cal</td>
<td></td>
</tr>
<tr>
<td>2-Port cal</td>
<td>502</td>
</tr>
</tbody>
</table>

a Typical performance.

b Includes sweep time, retrace time and band-crossing time. Analyzer display turned off with DISPLAY:ENABLE OFF. Add 21 ms for display on. Data for one trace (S11) measurement.
Table 43. Cycle Time vs IF Bandwidth

Applies to the Preset condition (201 points, correction off) except for the following changes:

- CF = 28 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)

<table>
<thead>
<tr>
<th>IF Bandwidth (Hz)</th>
<th>Cycle Time (ms)(^b)</th>
<th>Cycle Time (ms) Option 080 enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>40,000</td>
<td>11</td>
<td>100</td>
</tr>
<tr>
<td>35,000</td>
<td>12</td>
<td>101</td>
</tr>
<tr>
<td>30,000</td>
<td>13</td>
<td>102</td>
</tr>
<tr>
<td>20,000</td>
<td>16</td>
<td>106</td>
</tr>
<tr>
<td>10,000</td>
<td>30</td>
<td>127</td>
</tr>
<tr>
<td>7000</td>
<td>38</td>
<td>138</td>
</tr>
<tr>
<td>5000</td>
<td>50</td>
<td>152</td>
</tr>
<tr>
<td>3000</td>
<td>74</td>
<td>182</td>
</tr>
<tr>
<td>1000</td>
<td>274</td>
<td>326</td>
</tr>
<tr>
<td>300</td>
<td>694</td>
<td>782</td>
</tr>
<tr>
<td>100</td>
<td>1905</td>
<td>2054</td>
</tr>
<tr>
<td>30</td>
<td>6091</td>
<td>6355</td>
</tr>
<tr>
<td>10</td>
<td>17916</td>
<td>18372</td>
</tr>
</tbody>
</table>

\(^a\) Typical performance.
\(^b\) Cycle time includes sweep and retrace time.

Table 44. Cycle Time vs Number of Points

Applies to the Preset condition (35 kHz IF bandwidth, correction off) except for the following changes:

- CF = 28 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)
<table>
<thead>
<tr>
<th>Number of Points</th>
<th>Cycle Time (ms)(^b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>51</td>
<td>7</td>
</tr>
<tr>
<td>101</td>
<td>9</td>
</tr>
<tr>
<td>201</td>
<td>12</td>
</tr>
<tr>
<td>401</td>
<td>18</td>
</tr>
<tr>
<td>801</td>
<td>30</td>
</tr>
<tr>
<td>1601</td>
<td>55</td>
</tr>
<tr>
<td>16,001</td>
<td>497</td>
</tr>
</tbody>
</table>

\(^a\) Typical performance.  
\(^b\) Cycle time includes sweep and retrace time.

Table 45. Data Transfer Time (ms)\(^a\)

<table>
<thead>
<tr>
<th>Number of Points</th>
<th>201</th>
<th>401</th>
<th>1601</th>
<th>16,001</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCPI over GPIB (program executed on external PC)</td>
<td>32-bit floating point</td>
<td>7</td>
<td>12</td>
<td>43</td>
</tr>
<tr>
<td></td>
<td>64-bit floating point</td>
<td>12</td>
<td>22</td>
<td>84</td>
</tr>
<tr>
<td></td>
<td>ASCII</td>
<td>64</td>
<td>124</td>
<td>489</td>
</tr>
<tr>
<td>SCPI (program executed in the analyzer)</td>
<td>32-bit floating point</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>64-bit floating point</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>ASCII</td>
<td>29</td>
<td>56</td>
<td>222</td>
</tr>
<tr>
<td>COM (program executed in the analyzer)</td>
<td>32-bit floating point</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Variant type</td>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>DCOM over LAN (program executed on external PC)</td>
<td>2868</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32-bit floating point</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>121</td>
</tr>
<tr>
<td>----------------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>-----</td>
</tr>
<tr>
<td>Variant type</td>
<td>3</td>
<td>6</td>
<td>19</td>
<td>939</td>
</tr>
</tbody>
</table>

**Test Set Block Diagrams**

**E836xB - Standard Configuration and Standard Power Range**

**E836xB - Option UNL Standard Configuration with Extended Power Range and Bias - Tees**
E836xB - Option UNL Standard Configuration with Extended Power Range and Bias - Tees, and Option 016, Receiver Attenuators
Test Set with Option 014 Block Diagrams

E836xB - Option 014 – Configurable Test Set and Standard Power Range

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

E836xB - Option 014 – Configurable Test Set and Standard Power Range, and Option 081 Reference Channel Transfer Switch
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

E836xB - Option 014 Configurable Test Set, and Option UNL Extended Power Range and Bias - Tees
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

E836xB - Option 014 Configurable Test Set, and Option UNL Extended Power Range and Bias - Tees, and Option 081 Reference Channel Transfer Switch
### Item Description

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

E836xB - Option 014 Configurable Test Set and Option UNL, Extended Power Range and Bias - Tees and Option 016 Receiver Attenuators
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B  IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

E836xB - Option 014 Configurable Test Set, and Option UNL Extended Power Range and Bias - Tees, and Option 016 Receiver Attenuators, and Option 081 Reference Channel Transfer Switch
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

Last modified:

- **Jan. 10, 2007**: Revised line power, and CPU values; moved regulatory, operating, and non-operating info to the I&QS Guide.
- **Oct. 5, 2006**: Added 350W typical to line power
- **Oct. 28, 2005**: Previous revision
Technical Specifications for the N5230A

Options 020/025, 120/125, 220/225, 420/425, or 520/525 (2-Port PNA)
(Rev. 2007-01-10)

This is a complete list of the N5230A Options 020, 025, 120, 125, 220, 225, 420, 425, 520, 525 network analyzer technical specifications.

- To optimize viewing of uncertainty curves, click the Maximize button.
- To view or print the .pdf version of the specifications, visit our web site at http://www.agilent.com/find/pna, and search for "N5230A Specifications"
- This N5230A document provides technical specifications for the 85056A 2.4 mm, 85052B 3.5 mm, and 85032B Type-N calibration kits and the N4691A, and N4693A ECal modules. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the curves for your calibration kit and PNA setup.

- **Definitions**
- **Corrected System Performance**
  - **System Dynamic Range**
  - **Extended Dynamic Range**
  - **3.5mm Connectors**
  - **2.4mm Connectors**
  - **Type-N Connectors**
- **Uncorrected System Performance**
- **Test Port Output**
- **Test Port Input**
- **Dynamic Accuracy**
- **Group Delay**
- **General Information**
- **Measurement Throughput Summary**
- **Front-panel Jumper Specs (Options 025, 125, 225, 425, 525)**
**Definitions**

All specifications and characteristics apply over a 25 °C ±5 °C range (unless otherwise stated) and 90 minutes after the instrument has been turned on.

**Specification (spec.):** Warranted performance. Specifications include guardbands to account for the expected statistical performance distribution, measurement uncertainties, and changes in performance due to environmental conditions.

**Characteristic (char.):** A performance parameter that the product is expected to meet before it leaves the factory, but that is not verified in the field and is not covered by the product warranty. A characteristic includes the same guardbands as a specification.

**Typical (typ.):** Expected performance of an average unit which does not include guardbands. It is not covered by the product warranty.

**Nominal (nom.):** A general, descriptive term that does not imply a level of performance. It is not covered by the product warranty.

**Calibration:** The process of measuring known standards to characterize a network analyzer's systematic (repeatable) errors.

**Corrected (residual):** Indicates performance after error correction (calibration). It is determined by the quality of calibration standards and how well "known" they are, plus system repeatability, stability, and noise.

**Uncorrected (raw):** Indicates instrument performance without error correction. The uncorrected performance affects the stability of a calibration.

**Standard:** When referring to the analyzer, this includes no options unless noted otherwise.

---

**Corrected System Performance**

The specifications in this section apply for measurements made with the N5230A analyzer with the following conditions:

- 10 Hz IF bandwidth
- No averaging applied to data
- Isolation calibration with an averaging factor of 8

- **System Dynamic Range**
- **Extended Dynamic Range**
- **3.5mm Connectors**
### 2.4mm Connectors

#### Type-N Connectors

<table>
<thead>
<tr>
<th>Table 1. System Dynamic Range¹</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Standard Configuration and Standard Power Range</td>
</tr>
<tr>
<td>3 MHz to 10 MHz</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
</tr>
<tr>
<td>45 MHz to 70 MHz²</td>
</tr>
<tr>
<td>70 MHz to 500 MHz²</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
</tr>
<tr>
<td>2 GHz to 6 GHz</td>
</tr>
<tr>
<td>6 GHz to 8 GHz</td>
</tr>
<tr>
<td>8 GHz to 9 GHz</td>
</tr>
<tr>
<td>9 GHz to 10.5 GHz</td>
</tr>
<tr>
<td>10.5 GHz to 12.5 GHz</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
</tr>
<tr>
<td>20 GHz to 31.25 GHz</td>
</tr>
<tr>
<td>31.25 GHz to 40 GHz</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
</tr>
</tbody>
</table>
Table 1. System Dynamic Range\(^1\) (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB) at Test Port</th>
<th>Typical (dB) at Test Port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Options 025, 125</td>
<td>Option 225</td>
</tr>
<tr>
<td>Configurable Test Set and Extended Power Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 3 MHz(^3)</td>
<td>92(^4)</td>
<td>--</td>
</tr>
<tr>
<td>3 MHz to 10 MHz</td>
<td>112</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>121</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 70 MHz(^2)</td>
<td>121</td>
<td>101</td>
</tr>
<tr>
<td>70 MHz to 500 MHz(^2)</td>
<td>121</td>
<td>105</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>121</td>
<td>110</td>
</tr>
<tr>
<td>2 GHz to 6 GHz</td>
<td>121</td>
<td>110</td>
</tr>
<tr>
<td>6 GHz to 8 GHz</td>
<td>120</td>
<td>110</td>
</tr>
<tr>
<td>8 GHz to 9 GHz</td>
<td>120</td>
<td>110</td>
</tr>
<tr>
<td>9 GHz to 10.5 GHz</td>
<td>116</td>
<td>110</td>
</tr>
<tr>
<td>10.5 GHz to 12.5 GHz</td>
<td>111</td>
<td>110</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>108</td>
<td>108</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>108</td>
</tr>
<tr>
<td>20 GHz to 31.25 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>31.25 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

1. The system dynamic range is calculated as the difference between the noise floor and the specified source maximum output power.
2. The effective dynamic range must take measurement uncertainties and interfering signals into account.
The system dynamic range is calculated as the difference between the noise floor and the specified source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account.

2 May be degraded by 10 dB at particular frequencies (multiples of 5 MHz) below 500 MHz due to spurious receiver residuals. Methods are available to regain the full dynamic range.

3 May be limited by Crosstalk at certain frequencies below 3 MHz.

4 Value and frequency band changed July 2006.

Receiver Dynamic Range technical specifications are not provided in this N5230A specs document.

### Table 2. Extended Dynamic Range

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB) at Direct Receiver Access Input</th>
<th>Typical (dB) at Direct Receiver Access Input</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Option 025, 125</td>
<td>Option 225</td>
</tr>
<tr>
<td>Configurable Test Set and Extended Power Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 3 MHz$^3$</td>
<td>108$^4$</td>
<td>--</td>
</tr>
<tr>
<td>3 MHz to 10 MHz</td>
<td>128</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>137</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 70 MHz$^2$</td>
<td>137</td>
<td>113</td>
</tr>
<tr>
<td>70 MHz to 500 MHz$^2$</td>
<td>137</td>
<td>117</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>137</td>
<td>122</td>
</tr>
<tr>
<td>2 GHz to 6 GHz</td>
<td>137</td>
<td></td>
</tr>
<tr>
<td>6 GHz to 8 GHz</td>
<td>136</td>
<td>122</td>
</tr>
<tr>
<td>8 GHz to 9 GHz</td>
<td>136</td>
<td></td>
</tr>
<tr>
<td>9 GHz to 10.5 GHz</td>
<td>132</td>
<td>122</td>
</tr>
<tr>
<td>10.5 GHz to 12.5 GHz</td>
<td>127</td>
<td>122</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>124</td>
<td>120</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>120</td>
</tr>
<tr>
<td>Frequency Band</td>
<td>Specifications</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>20 GHz to 31.25 GHz</td>
<td>103</td>
<td></td>
</tr>
<tr>
<td>31.25 GHz to 40 GHz</td>
<td>98</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>83</td>
<td></td>
</tr>
</tbody>
</table>

1 The direct receiver access input extended dynamic range is calculated as the difference between the direct receiver access input noise floor and the source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account. This set-up should only be used when the receiver input will never exceed its compression or damage level. When the analyzer is in segment sweep mode, it can have predefined frequency segments which will output a higher power level when the extended dynamic range is required (i.e. devices with high insertion loss), and reduced power when receiver compression or damage may occur (i.e. devices with low insertion loss). The extended range is only available in one-path transmission measurements.

2 May be degraded by 10 dB at particular frequencies (multiples of 5 MHz) below 500 MHz due to spurious receiver residuals. Methods are available to regain the full dynamic range.

3 May be limited by Crosstalk at certain frequencies below 3MHz.

4 Value and frequency band changed July 2006.

Corrected System Performance with 3.5mm Connectors (Tables 3 - 8)
Corrected System Performance with 2.4mm Connectors (Tables 9 - 12)
Corrected System Performance with Type-N Connectors (Tables 13 - 14)

### Table 15. Uncorrected System Performance

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifications</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Options</td>
<td>Options</td>
</tr>
<tr>
<td></td>
<td>020, 025, 120, 125</td>
<td>220, 225</td>
</tr>
<tr>
<td>Directivity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 10 MHz</td>
<td>16 dB</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>28 dB</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>28 dB</td>
<td>24 dB</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>500 MHz to 1 GHz</td>
<td>1 GHz to 2 GHz</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------</td>
<td>----------------</td>
</tr>
<tr>
<td></td>
<td>28 dB</td>
<td>27 dB</td>
</tr>
<tr>
<td></td>
<td>25 dB</td>
<td>27 dB</td>
</tr>
<tr>
<td></td>
<td>25 dB</td>
<td>21 dB</td>
</tr>
<tr>
<td></td>
<td>20 dB</td>
<td>21 dB</td>
</tr>
<tr>
<td></td>
<td>17 dB</td>
<td>21 dB</td>
</tr>
<tr>
<td></td>
<td>17 dB</td>
<td>16 dB</td>
</tr>
<tr>
<td></td>
<td>15 dB</td>
<td>16 dB</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>16 dB</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**Source Match**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>300 kHz to 10 MHz</th>
<th>10 MHz to 45 MHz</th>
<th>45 MHz to 500 MHz</th>
<th>500 MHz to 2 GHz</th>
<th>2 GHz to 3 GHz</th>
<th>3 GHz to 8 GHz</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>18 dB</td>
<td>25 dB</td>
<td>25 dB</td>
<td>21 dB</td>
<td>19 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>20 dB</td>
<td>17 dB</td>
<td>12 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>17 dB</td>
<td>17 dB</td>
<td>12 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>17 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>17 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>17 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>
### Table 15. Uncorrected System Performance (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifications</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>8 GHz to 9 GHz</strong></td>
<td>Options 020, 025, 120, 125</td>
<td>Options 220, 225</td>
</tr>
<tr>
<td>Load Match</td>
<td>Options 020, 025, 120, 125</td>
<td>Options 220, 225</td>
</tr>
<tr>
<td>300 kHz to 10 MHz</td>
<td>17 dB</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>22 dB</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>22 dB</td>
<td>22 dB</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>17 dB</td>
<td>20 dB</td>
</tr>
<tr>
<td>2 GHz to 3 GHz</td>
<td>14 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td>3 GHz to 8 GHz</td>
<td>10 dB</td>
<td>12 dB</td>
</tr>
<tr>
<td>Description</td>
<td>Specifications</td>
<td>Typical</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td></td>
<td>Options 020, 025, 120, 125</td>
<td>Options 220, 225</td>
</tr>
<tr>
<td></td>
<td>Options 420, 425</td>
<td>Options 520, 525</td>
</tr>
<tr>
<td></td>
<td>Options 020, 025, 120, 125</td>
<td>Options 220, 225</td>
</tr>
<tr>
<td></td>
<td>Options 420, 425</td>
<td>Options 520, 525</td>
</tr>
<tr>
<td>Crosstalk&lt;sup&gt;1&lt;/sup&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 8 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>8 GHz to 10.5 GHz</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

*Table 15. Uncorrected System Performance (Continued)*
<table>
<thead>
<tr>
<th></th>
<th>Specifications</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Options 020,</td>
<td>Options 120, 125</td>
</tr>
<tr>
<td></td>
<td>Options 025</td>
<td>Options 220, 225</td>
</tr>
<tr>
<td></td>
<td>Options 420,</td>
<td>Options 520, 525</td>
</tr>
<tr>
<td></td>
<td>Options 425</td>
<td>Options 020, 025, 120, 125</td>
</tr>
<tr>
<td></td>
<td>Options 220,</td>
<td>Options 220, 225</td>
</tr>
<tr>
<td></td>
<td>Options 425</td>
<td>Options 420, 425</td>
</tr>
<tr>
<td></td>
<td>Options 520,</td>
<td>Options 520, 525</td>
</tr>
</tbody>
</table>

Table 16. Test Port Output

---

**Frequency Range**

| N5230A         | 300kHz to 6 GHz | 300kHz to 13.5 GHz | 10 MHz to 20 GHz | 10 MHz to 40 GHz | 10 MHz to 50 GHz | --                     |

**Nominal Power**

Preset power; attenuator switch point 10 dB below nominal power

|                  | 0 dBm       | 0 dBm       | -5 dBm      | -10 dBm      | -15 dBm      | --                     |

**Frequency Resolution**

|                  | 1 Hz        | --                      |

**CW Accuracy**
<table>
<thead>
<tr>
<th>Description</th>
<th>Specifications</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Options 020, 025, 120, 125</td>
<td>Options 220, 225</td>
</tr>
<tr>
<td>Frequency Stability</td>
<td>--</td>
<td>+/-0.05 ppm, -10° to 70° C</td>
</tr>
</tbody>
</table>

**Table 16. Test Port Output** (Continued)

**Power Level Accuracy**

<table>
<thead>
<tr>
<th>Description</th>
<th>Range</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>300 kHz to 10 MHz</td>
<td>+/±1.0 dB</td>
</tr>
<tr>
<td></td>
<td>10 MHz to 45 MHz</td>
<td>+/±1.0 dB</td>
</tr>
<tr>
<td></td>
<td>45 MHz to 6 GHz</td>
<td>+/±1.0 dB</td>
</tr>
<tr>
<td></td>
<td>6 GHz to 8 GHz</td>
<td>+/±1.5 dB</td>
</tr>
<tr>
<td></td>
<td>8 GHz to 9 GHz</td>
<td>+/±1.5 dB</td>
</tr>
<tr>
<td></td>
<td>9 GHz to 10.5 GHz</td>
<td>+/±1.5 dB</td>
</tr>
</tbody>
</table>

Variation from nominal power in range 0
<table>
<thead>
<tr>
<th>Specifications</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Options</td>
</tr>
<tr>
<td></td>
<td>020, 120</td>
</tr>
<tr>
<td><strong>Max Leveled Power</strong></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 10 MHz</td>
<td>10 dBm</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>10 dBm</td>
</tr>
<tr>
<td>45 MHz to 6 GHz</td>
<td>10 dBm</td>
</tr>
<tr>
<td>6 GHz to 9 GHz</td>
<td>8 dBm</td>
</tr>
<tr>
<td>9 GHz to 12.5 GHz</td>
<td>4 dBm</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>2 dBm</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
</tr>
</tbody>
</table>

Table 16. Test Port Output\(^1\) (Continued)
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Low</th>
<th>High</th>
<th>Lineararity</th>
<th>Low</th>
<th>High</th>
<th>Lineararity</th>
<th>Low</th>
<th>High</th>
<th>Lineararity</th>
<th>Low</th>
<th>High</th>
<th>Lineararity</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-5</td>
<td>dBm</td>
<td>--</td>
<td>-8</td>
<td>dBm</td>
<td>--</td>
<td>-5</td>
<td>dBm</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-11</td>
<td>dBm</td>
<td>--</td>
<td>-15</td>
<td>dBm</td>
<td>--</td>
</tr>
</tbody>
</table>

**Power Level Linearity**

Test reference is at the nominal power level

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Lineararity</th>
<th>Lineararity</th>
<th>Lineararity</th>
<th>Lineararity</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 1 MHz</td>
<td>+/- 4.5 dB</td>
<td>+/- 4.5 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>1 MHz to 10 MHz</td>
<td>+/- 1.0 dB</td>
<td>+/- 1.0 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>+/- 2.0 dB</td>
<td>+/- 2.0 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 1 GHz</td>
<td>+/- 2.0 dB</td>
<td>+/- 1.0 dB</td>
<td>+/- 1.0 dB</td>
<td>+/- 1.0 dB</td>
</tr>
<tr>
<td>1 GHz to 12.5 GHz</td>
<td>+/- 1.5 dB</td>
<td>+/- 1.5 dB</td>
<td>+/- 1.0 dB</td>
<td>+/- 1.0 dB</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>+/- 1.5 dB</td>
<td>+/- 1.5 dB</td>
<td>+/- 1.0 dB</td>
<td>+/- 1.0 dB</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
<td>+/- 1.0 dB</td>
<td>+/- 1.0 dB</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
<td>+/- 1.0 dB</td>
<td>+/- 1.0 dB</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>+/- 1.0 dB</td>
</tr>
</tbody>
</table>

**Power Sweep Range (ALC)**

Options as indicated

<table>
<thead>
<tr>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>020, 025, 120, 125</td>
</tr>
<tr>
<td>220 &amp; 225</td>
</tr>
<tr>
<td>420, 425, 520, 525</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Table 16. Test Port Output</strong> (Continued)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifications</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Option 020, 120</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Option 025, 125</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Option 220</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Option 225</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Option 420, 520</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Option 425, 525</td>
</tr>
<tr>
<td><strong>Power Range</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
<td>-30 to +10 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-90 to +9 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Description</td>
<td>Specifications</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------</td>
<td>----------------</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>-30 to +10 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-90 to +9 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +12 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +12 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +9 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +8 dBm</td>
</tr>
<tr>
<td>45 MHz to 6 GHz</td>
<td>--</td>
<td>-30 to +10 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-90 to +9 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +12 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +12 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +8 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +8 dBm</td>
</tr>
<tr>
<td>6 GHz to 9 GHz</td>
<td>--</td>
<td>-30 to +8 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-90 to +8 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +12 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +12 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +8 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +8 dBm</td>
</tr>
<tr>
<td>9 GHz to 12.5 GHz</td>
<td>--</td>
<td>-30 to +4 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-90 to +4 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +12 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +12 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +8 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +8 dBm</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>--</td>
<td>-30 to +2 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-90 to +1 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +7 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +7 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-27 to +5 dBm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +4 dBm</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>--</td>
</tr>
</tbody>
</table>

**Power Settings**

<table>
<thead>
<tr>
<th>Minimum Power Setting</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>-33 dBm</td>
</tr>
<tr>
<td></td>
<td>-93 dBm</td>
</tr>
<tr>
<td></td>
<td>-30 dBm</td>
</tr>
<tr>
<td></td>
<td>-90 dBm</td>
</tr>
<tr>
<td></td>
<td>-30 dBm</td>
</tr>
<tr>
<td></td>
<td>-90 dBm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Maximum Power Setting</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>+20 dBm</td>
</tr>
</tbody>
</table>

**Table 16. Test Port Output**

The table above provides specifications for various frequency ranges and their corresponding power settings. The table includes columns for the frequency range, description, and specifications. The specifications are listed for each frequency range, indicating the typical range of values for power settings.
<table>
<thead>
<tr>
<th>Phase Noise (Nominal power at test port)</th>
<th>--</th>
<th>10 kHz Offset</th>
<th>100 kHz Offset</th>
<th>1 MHz Offset</th>
<th>10 kHz Offset</th>
<th>100 kHz Offset</th>
<th>1 MHz Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
<td>-86 dBc/Hz</td>
<td>-86 dBc/Hz</td>
<td>-95 dBc/Hz</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 1.5 GHz</td>
<td>--</td>
<td>-86 dBc/Hz</td>
<td>-91 dBc/Hz</td>
<td>-95 dBc/Hz</td>
<td>-77 dBc/Hz</td>
<td>-77 dBc/Hz</td>
<td>-89 dBc/Hz</td>
</tr>
<tr>
<td>1.5 GHz to 3.125 GHz</td>
<td>--</td>
<td>-83 dBc/Hz</td>
<td>-91 dBc/Hz</td>
<td>-95 dBc/Hz</td>
<td>-83 dBc/Hz</td>
<td>-91 dBc/Hz</td>
<td>-95 dBc/Hz</td>
</tr>
<tr>
<td>3.125 GHz to 6.25 GHz</td>
<td>--</td>
<td>-77 dBc/Hz</td>
<td>-85 dBc/Hz</td>
<td>-89 dBc/Hz</td>
<td>-77 dBc/Hz</td>
<td>-85 dBc/Hz</td>
<td>-89 dBc/Hz</td>
</tr>
<tr>
<td>6.25 GHz to 12.5 GHz</td>
<td>--</td>
<td>-71 dBc/Hz</td>
<td>-79 dBc/Hz</td>
<td>-83 dBc/Hz</td>
<td>-71 dBc/Hz</td>
<td>-79 dBc/Hz</td>
<td>-83 dBc/Hz</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>--</td>
<td>-65 dBc/Hz</td>
<td>-73 dBc/Hz</td>
<td>-77 dBc/Hz</td>
<td>-65 dBc/Hz</td>
<td>-73 dBc/Hz</td>
<td>-77 dBc/Hz</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-65 dBc/Hz</td>
<td>-73 dBc/Hz</td>
<td>-77 dBc/Hz</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-59 dBc/Hz</td>
<td>-67 dBc/Hz</td>
<td>-71 dBc/Hz</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>-59 dBc/Hz</td>
<td>-67 dBc/Hz</td>
<td>-71 dBc/Hz</td>
</tr>
</tbody>
</table>

Table 16. Test Port Output^1 (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specifications</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>Options 020, 025, 120, 125, 220, 225, 420, 520, 425, 525</td>
<td></td>
</tr>
</tbody>
</table>

Non-Harmonic Spurious (at Nominal Output Power)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 13.5 GHz</td>
<td>-50 dBc for offset frequency &gt; 1 kHz</td>
</tr>
<tr>
<td>Description</td>
<td>Specifications</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 16. Test Port Output\(^1\) (Continued)

<table>
<thead>
<tr>
<th>Harmonics (2nd or 3rd) at Maximum Output Power</th>
<th>Specifications</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
<td>-17 dBc</td>
</tr>
<tr>
<td>10 MHz to 500 MHz</td>
<td>--</td>
<td>-17 dBc</td>
</tr>
<tr>
<td>500 MHz to 1 GHz</td>
<td>--</td>
<td>-17 dBc</td>
</tr>
<tr>
<td>1 GHz to 13.5 GHz</td>
<td>--</td>
<td>-20 dBc</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>-22 dBc</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>-22 dBc</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>-22 dBc</td>
</tr>
</tbody>
</table>

1Performance specified on Port 1 only. Port 2 performance is a characteristic.

2Power level linearity specified on Port 1 only. Port 2 performance is Typical. Test reference is at the nominal power level.
3ALC range starts at maximum leveled power and decreases in power level by the dB amount specified here.

Table 17. Test Port Input

<table>
<thead>
<tr>
<th>Description</th>
<th>Options</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>020, 025, 120, 125</td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
</tr>
<tr>
<td>Options</td>
<td>220, 225</td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
</tr>
<tr>
<td>Options</td>
<td>420, 425</td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
</tr>
<tr>
<td>Options</td>
<td>520, 525</td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
</tr>
<tr>
<td>Options</td>
<td>020, 025, 120, 125</td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
</tr>
<tr>
<td>Options</td>
<td>220, 225</td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
</tr>
<tr>
<td>Options</td>
<td>420, 425</td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
</tr>
<tr>
<td>Options</td>
<td>520, 525</td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
</tr>
</tbody>
</table>

Test Port Noise Floor\(^1\)

<table>
<thead>
<tr>
<th>10 Hz IF Bandwidth (^5)</th>
<th>Options</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 3 MHz</td>
<td>(-83 \text{ dBm}) (^6)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>3 MHz to 10 MHz</td>
<td>(-103 \text{ dBm})</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>(-112 \text{ dBm})</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>45 MHz to 70 MHz</td>
<td>(-112 \text{ dBm})</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>70 MHz to 500 MHz</td>
<td>(-112 \text{ dBm})</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>(-112 \text{ dBm})</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>2 GHz to 4 GHz</td>
<td>(-112 \text{ dBm})</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>4 GHz to 8 GHz</td>
<td>(-112 \text{ dBm})</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>8 GHz to 10.5 GHz</td>
<td>(-112 \text{ dBm})</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>(-107 \text{ dBm})</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;\text{Options}&gt;</td>
<td>&lt;\text{Options}&gt;</td>
<td></td>
</tr>
<tr>
<td>Frequency Range</td>
<td>13.5 GHz to 20 GHz</td>
<td>20 GHz to 31.25 GHz</td>
<td>31.25 GHz to 40 GHz</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------</td>
<td>--------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>&lt;-105 dBm</td>
<td>&lt;-100 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-100 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-95 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-76 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-80 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-90 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-90 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-90 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-90 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-80 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-80 dBm</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>&lt;-80 dBm</td>
</tr>
<tr>
<td>Description</td>
<td>Specification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>---------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Direct Receiver Access Input Noise Floor¹</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Hz IF Bandwidth</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 3 MHz</td>
<td>&lt;-99 dBm⁶</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 MHz to 10 MHz</td>
<td>&lt;119 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>&lt;128 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>45 MHz to 70 MHz</td>
<td>&lt;128 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>70 MHz to 500 MHz</td>
<td>&lt;128 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>&lt;128 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 GHz to 8 GHz</td>
<td>&lt;128 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 GHz to 10.5 GHz</td>
<td>&lt;128 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>&lt;128 dBm</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

¹ (Options 025, 125, 225, 425, and 525 only)
<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>13.5 GHz to 20 GHz</th>
<th>20 GHz to 31.25 GHz</th>
<th>31.25 GHz to 40 GHz</th>
<th>40 GHz to 50 GHz</th>
<th>300 kHz to 3 MHz</th>
<th>3 MHz to 10 MHz</th>
<th>10 MHz to 45 MHz</th>
<th>45 MHz to 70 MHz</th>
<th>70 MHz to 500 MHz</th>
<th>500 MHz to 2 GHz</th>
<th>2 GHz to 8 GHz</th>
<th>8 GHz to 10.5 GHz</th>
<th>10.5 GHz to 13.5 GHz</th>
<th>13.5 GHz to 20 GHz</th>
<th>20 GHz to 31.25 GHz</th>
<th>31.25 GHz to 40 GHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>&lt;-123 dBm</td>
<td>&lt;-117 dBm</td>
<td>&lt;-112 dBm</td>
<td>&lt;-112 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 31.25 GHz</td>
<td>--</td>
<td>--</td>
<td>&lt;-111 dBm</td>
<td>&lt;-111 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>31.25 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
<td>&lt;-106 dBm</td>
<td>&lt;-106 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>&lt;-98 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td><strong>1 kHz IF Bandwidth</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 3 MHz</td>
<td>&lt;-89 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>3 MHz to 10 MHz</td>
<td>&lt;-99 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>&lt;-108 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>&lt;100 dBm</td>
<td>&lt;106 dBm</td>
<td>&lt;106 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 70 MHz</td>
<td>&lt;-108 dBm</td>
<td>&lt;-88 dBm</td>
<td>&lt;-91 dBm</td>
<td>&lt;-91 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>70 MHz to 500 MHz</td>
<td>&lt;-108 dBm</td>
<td>&lt;-92 dBm</td>
<td>&lt;-91 dBm</td>
<td>&lt;-91 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>&lt;-108 dBm</td>
<td>&lt;-97 dBm</td>
<td>&lt;-102 dBm</td>
<td>&lt;-102 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 8 GHz</td>
<td>&lt;-108 dBm</td>
<td>&lt;-97 dBm</td>
<td>&lt;-102 dBm</td>
<td>&lt;-102 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>8 GHz to 10.5 GHz</td>
<td>&lt;-108 dBm</td>
<td>&lt;-97 dBm</td>
<td>&lt;-92 dBm</td>
<td>&lt;-92 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>&lt;-108 dBm</td>
<td>&lt;-97 dBm</td>
<td>&lt;-92 dBm</td>
<td>&lt;-92 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>&lt;-103 dBm</td>
<td>&lt;-97 dBm</td>
<td>&lt;-92 dBm</td>
<td>&lt;-92 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 31.25 GHz</td>
<td>--</td>
<td>--</td>
<td>&lt;-91 dBm</td>
<td>&lt;-91 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>31.25 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
<td>&lt;-86 dBm</td>
<td>&lt;-86 dBm</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>
### Table 17. Test Port Input (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options 220, 225</td>
<td>Options 420, 520</td>
<td>Options 425, 525</td>
</tr>
<tr>
<td><strong>Compression Level</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz(^2)</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>+5 dBm</td>
<td>0.10 dB</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>+5 dBm</td>
<td>0.15 dB</td>
</tr>
<tr>
<td>2 GHz to 8 GHz</td>
<td>+5 dBm</td>
<td>0.21 dB</td>
</tr>
<tr>
<td>8 GHz to 12.5 GHz</td>
<td>+5 dBm</td>
<td>0.21 dB</td>
</tr>
<tr>
<td>12.5 GHz to 20 GHz</td>
<td>+3 dBm</td>
<td>0.20 dB</td>
</tr>
<tr>
<td>20 GHz to 31.25 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>31.25 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>
Table 17. Test Port Input (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options</td>
<td></td>
<td>--</td>
</tr>
<tr>
<td>020, 120, 025, 125</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Compression Level (continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Power</th>
<th>Compression</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>+8 dBm</td>
<td>1.0 dB (0.1 dB at +5 dBm typ.)</td>
<td>-- -- --</td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>+8 dBm</td>
<td>0.35 dB</td>
<td>-- -- --</td>
</tr>
<tr>
<td>50 MHz to 1 GHz</td>
<td>+8 dBm</td>
<td>0.35 dB</td>
<td>-- -- --</td>
</tr>
<tr>
<td>1 GHz to 6 GHz</td>
<td>+8 dBm</td>
<td>0.25 dB</td>
<td>-- -- --</td>
</tr>
<tr>
<td>6 GHz to 8 GHz</td>
<td>+8 dBm</td>
<td>0.25 dB</td>
<td>-- -- --</td>
</tr>
<tr>
<td>8 GHz to 12.5 GHz</td>
<td>+8 dBm</td>
<td>0.30 dB</td>
<td>-- -- --</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>+8 dBm</td>
<td>0.40 dB</td>
<td>-- -- --</td>
</tr>
</tbody>
</table>
### Table 17. Test Port Input (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>Options 020, 120, 025, 125</td>
<td></td>
</tr>
</tbody>
</table>

#### Test Port Compression - 0.1 dB

<table>
<thead>
<tr>
<th>Description</th>
<th>Power</th>
<th>Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>+5 dBm</td>
<td>0.1 dB</td>
</tr>
<tr>
<td>10 MHz to 1 GHz</td>
<td>+9 dBm</td>
<td>0.1 dB</td>
</tr>
<tr>
<td>1 GHz to 12.5 GHz</td>
<td>+10 dBm</td>
<td>0.1 dB</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>+9 dBm</td>
<td>0.1 dB</td>
</tr>
</tbody>
</table>

### Table 17. Test Port Input (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options 220, 225</td>
<td>Options 420, 425</td>
<td>Options 520, 525</td>
</tr>
<tr>
<td>Option 220</td>
<td>Option 225</td>
<td>Options 420, 425</td>
</tr>
</tbody>
</table>

#### Test Port Compression - 0.1 dB

<table>
<thead>
<tr>
<th>Description</th>
<th>Power</th>
<th>Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz²</td>
<td>+10 dBm</td>
<td>negligible</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>+10 dBm</td>
<td>+1 dBm</td>
</tr>
<tr>
<td>Description</td>
<td>Specification</td>
<td>Typical</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------</td>
<td>------------------</td>
</tr>
<tr>
<td></td>
<td><strong>Options</strong></td>
<td><strong>Options</strong></td>
</tr>
<tr>
<td></td>
<td><strong>220, 225</strong></td>
<td><strong>420, 520</strong></td>
</tr>
</tbody>
</table>

**Trace Noise Magnitude**³

1 kHz IF bandwidth, ratioed measurement, nominal power at test port.

<table>
<thead>
<tr>
<th>Description</th>
<th>Options</th>
<th>Options</th>
<th>Options</th>
<th>Options</th>
<th>Options</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>0.004 dB rms</td>
<td>0.015 dB rms</td>
<td>0.015 dB rms</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>0.004 dB rms</td>
<td>0.010 dB rms</td>
<td>0.010 dB rms</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>0.004 dB rms</td>
<td>0.006 dB rms</td>
<td>0.006 dB rms</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

---

Table 17. Test Port Input (Continued)
<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 GHz to 10.5 GHz</td>
<td>0.004 dB rms</td>
<td>0.006 dB rms</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>0.006 dB rms</td>
<td>0.010 dB rms</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>0.006 dB rms</td>
<td>0.010 dB rms</td>
</tr>
<tr>
<td>20 GHz to 31.25 GHz</td>
<td>--</td>
<td>0.010 dB rms</td>
</tr>
<tr>
<td>31.25 GHz to 40 GHz</td>
<td>--</td>
<td>0.020 dB rms</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>0.020 dB rms</td>
</tr>
</tbody>
</table>

**Table 17. Test Port Input (Continued)**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options 020, 120, 025, 125</td>
<td>Options 020, 120, 025, 125</td>
<td></td>
</tr>
</tbody>
</table>

**Trace Noise Magnitude³ (continued)**

**100 kHz IF bandwidth, ratioed measurement, nominal power at test port**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>12 mdB</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 6 GHz</td>
<td>4 mdB</td>
<td>--</td>
</tr>
<tr>
<td>6 GHz to 10.5 GHz</td>
<td>4 mdB</td>
<td>--</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>8 mdB</td>
<td>--</td>
</tr>
</tbody>
</table>

**600 kHz IF bandwidth, ratioed measurement, nominal power at test port**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
<td>20 mdB</td>
</tr>
<tr>
<td>Description</td>
<td>Specification</td>
<td>Typical</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td></td>
<td>Options</td>
<td>Options</td>
</tr>
<tr>
<td></td>
<td>220, 225</td>
<td>420, 520</td>
</tr>
<tr>
<td>Trace Noise Phase&lt;sup&gt;3&lt;/sup&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 10 MHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>0.060° rms</td>
<td>0.100° rms</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>0.060° rms</td>
<td>0.060° rms</td>
</tr>
<tr>
<td>2 GHz to 10.5 GHz</td>
<td>0.060° rms</td>
<td>0.060° rms</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>0.060° rms</td>
<td>0.100° rms</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>0.060° rms</td>
<td>0.100° rms</td>
</tr>
<tr>
<td>20 GHz to 31.25 GHz</td>
<td>--</td>
<td>0.100° rms</td>
</tr>
<tr>
<td>31.25 GHz to 40 GHz</td>
<td>--</td>
<td>0.200° rms</td>
</tr>
</tbody>
</table>

<sup>3</sup> 1 kHz IF bandwidth, ratioed measurement, nominal power at test port.
<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Technical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options 020, 120, 025, 125</td>
<td>Options 020, 120, 025, 125</td>
<td></td>
</tr>
</tbody>
</table>

### Trace Noise Phase³ (continued)

#### 100 kHz IF bandwidth, ratioed measurement, nominal power at test port

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Specification</th>
<th>Technical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>80 mdeg</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 6 GHz</td>
<td>30 mdeg</td>
<td>--</td>
</tr>
<tr>
<td>6 GHz to 10.5 GHz</td>
<td>30 mdeg</td>
<td>--</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>60 mdeg</td>
<td>--</td>
</tr>
</tbody>
</table>

#### 600 kHz IF bandwidth, ratioed measurement, nominal power at test port

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
<td>100 mdeg</td>
</tr>
<tr>
<td>10 MHz to 6 GHz</td>
<td>--</td>
<td>60 mdeg</td>
</tr>
<tr>
<td>6 GHz to 10.5 GHz</td>
<td>--</td>
<td>60 mdeg</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>--</td>
<td>80 mdeg</td>
</tr>
</tbody>
</table>

Table 17. Test Port Input (Continued)
### Reference Level Magnitude

<table>
<thead>
<tr>
<th>Range</th>
<th>+/-200 dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution</td>
<td>0.001dB</td>
</tr>
</tbody>
</table>

### Reference Level Phase

<table>
<thead>
<tr>
<th>Range</th>
<th>+/-500°</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution</td>
<td>0.01°</td>
</tr>
</tbody>
</table>

### Table 17. Test Port Input (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Option 020, 025, 120, 125, 220, 225, 420, 425, 520, 525</th>
<th>Option 220, 225</th>
<th>Options 420, 425, 520, 525</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stability Magnitude&lt;sup&gt;4&lt;/sup&gt;</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Stability 1</td>
<td>Stability 2</td>
<td>Stability 3</td>
<td>Stability 4</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------</td>
<td>-------------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td><strong>Stability Phase 4</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 kHz to 10 MHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 4 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>4 GHz to 8 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>8 GHz to 13.5 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 40 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>40 GHz to 50 GHz</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>
### Table 17. Test Port Input (Continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Options 020, 120</td>
<td>Option 220</td>
</tr>
<tr>
<td><strong>Damage Input Level</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test Port 1 and 2</td>
<td>--</td>
<td>+27 dBm or +/- 16 VDC</td>
</tr>
<tr>
<td>R1, R2 in</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>A, B in</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Coupler Thru</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Coupler Arm</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

1. Total average (rms) noise power calculated as the mean value of a 801 linear magnitude traces expressed in dBm.
2. For Options 225, 420, 425, 520, 525 - Coupler roll-off will reduce compression to a negligible level below 45 MHz.
3. 1 kHz IF BW, ratioed measurement, nominal power at the test port.
4. Stability is defined as a ratio measurement made at the test port.
5. 10Hz IFBW test port noise floor performance is mathematically derived from the 1kHz IFBW noise floor performance. The performance could be limited by crosstalk below 3MHz at certain frequencies. The measurement is defined as a single receiver measurement with loads on the ports at a given CW frequency with power set to the minimum plus 5dB.

### Table 18. Dynamic Accuracy (Specification^a)

Accuracy of the test port input power reading relative to the reference input power level.

**Options 020, 025, 120, 125**

**Dynamic Accuracy, 300 kHz - 1 MHz, Option 020, 025, 120, or 125**
Dynamic Accuracy, 1 MHz-10 MHz, Option 020, 025, 120, or 125

**Magnitude**

Accuracy (dB)

Testport Power (dBm)

**Phase**

Accuracy (degrees)

Testport Power (dBm)

Dynamic Accuracy, 1 MHz-10 MHz, Option 020, 025, 120, or 125
Dynamic Accuracy, 10 MHz - 1 GHz, Option 020, 025, 120, or 125

Magnitude

Phase

Dynamic Accuracy, 10 MHz - 1 GHz, Option 020, 025, 120, or 125
Dynamic Accuracy, 1 - 8 GHz, Option 020, 025, 120, or 125

Magnitude

**N5230A Option 020, 025, 120 or 125**

Accuracy (dB)

Testport Power (dBm)

Phase

**N5230A Option 020, 025, 120 or 125**

Accuracy (degrees)

Testport Power (dBm)
Dynamic Accuracy, 8 - 10.5 GHz, Option 120 or 125
Dynamic Accuracy, 10.5 - 12.5 GHz, Option 120 or 125

Magnitude

Phase

Dynamic Accuracy, 10.5 - 12.5 GHz, Option 120 or 125
Dynamic Accuracy, 12.5 - 13.5 GHz, Option 120 or 125
Options 220, 225

Dynamic Accuracy, 0.045 GHz, Option 220 or 225
**Dynamic Accuracy, 0.500 GHz, Option 220 or 225**

**Phase**

![Phase Graph](image)

**Magnitude**

![Magnitude Graph](image)
Dynamic Accuracy, 0.500 - 2 GHz, Option 220 or 225

Magnitude

Phase
Dynamic Accuracy, 2 - 12.5 GHz, Option 220 or 225
Dynamic Accuracy, 12.5 - 20 GHz, Option 220 or 225

Phase

Magnitude
Options 420, 425, 520, 525

Dynamic Accuracy, 0.045 GHz, Option 420, 425, 520, or 525

Magnitude

Phase

N5230A Options 420 & 520

-10 dBm at 0.045 GHz
-20 dBm at 0.045 GHz
-30 dBm at 0.045 GHz
-40 dBm at 0.045 GHz
Dynamic Accuracy, 0.500 GHz, Option 420, 425, 520, or 525

Magnitude

Phase

Dynamic Accuracy, 0.500 GHz, Option 420, 425, 520, or 525
Dynamic Accuracy, 0.500 - 2 GHz, Option 420, 425, 520, or 525
Dynamic Accuracy, 2 - 8 GHz, Option 420, 425, 520, or 525
Dynamic Accuracy, 8 - 12.5 GHz, Option 420, 425, 520, or 525
Dynamic Accuracy, 12.5 - 20 GHz, Option 420, 425, 520, or 525
Dynamic Accuracy, 20 - 31.25 GHz, Option 420, 425, 520, or 525

Magnitude

Phase

Dynamic Accuracy, 20 - 31.25 GHz, Option 420, 425, 520, or 525
Dynamic Accuracy, 31.25 - 40 GHz, Option 420, 425, 520, or 525
Dynamic Accuracy, 40 - 50 GHz, Option 520 or 525

---

Magnitude

Phase

---

Dynamic Accuracy, 40 - 50 GHz, Option 520 or 525
Dynamic accuracy is verified with the following measurements:

- compression over frequency
- IF linearity at a single frequency of 1.195 GHz using a reference level of -20 dBm for an input power range of 0 to -110 dBm.

Table 19. Test Port Input (Group Delay)\textsuperscript{a}
<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental Information (typ.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aperture (selectable)</td>
<td>(frequency span)/(number of points -1)</td>
<td></td>
</tr>
<tr>
<td>Maximum Aperture</td>
<td>20% of frequency span</td>
<td></td>
</tr>
<tr>
<td>Range</td>
<td>0.5 x (1/minimum aperture)</td>
<td></td>
</tr>
<tr>
<td>Maximum Delay</td>
<td>Limited to measuring no more than 180° of phase change within the minimum aperture.</td>
<td></td>
</tr>
<tr>
<td>Accuracy</td>
<td>See graph below. Char.</td>
<td></td>
</tr>
</tbody>
</table>

The following graph shows characteristic group delay accuracy with full 2-port calibration and a 10 Hz IF bandwidth. Insertion loss is assumed to be < 2 dB and electrical length to be ten meters.

**Group Delay (Typical)**

![Graph showing group delay accuracy with full 2-port calibration and a 10 Hz IF bandwidth. Insertion loss is assumed to be < 2 dB and electrical length to be ten meters.]

In general, the following formula can be used to determine the accuracy, in seconds, of specific group delay measurement:

\[ \pm \text{Phase Accuracy (deg)} / [360 \times \text{Aperture (Hz)}] \]

Depending on the aperture and device length, the phase accuracy used is either incremental phase accuracy or worst case phase accuracy.

*Group delay is computed by measuring the phase change within a specified frequency step (determined by the frequency span and the number of points per sweep).*

**General Information**

- Miscellaneous Information
- Front Panel
Table 20. Miscellaneous Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>System IF Bandwidth Range</td>
<td></td>
<td>1 Hz to 250 kHz, nominal</td>
</tr>
<tr>
<td>CPU</td>
<td></td>
<td>Intel® 1.1 GHz Pentium® M with 1 GByte RAM</td>
</tr>
</tbody>
</table>

Table 21. Front Panel Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF Connectors</td>
<td></td>
</tr>
<tr>
<td>N5230A</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Option 220 or 225: 3.5 mm (male), 50 ohm, (nominal)</td>
</tr>
<tr>
<td></td>
<td>Option 420, 425, 520, or 525: 2.4 mm (male), 50 ohm, (nominal)</td>
</tr>
<tr>
<td>Center Pin Recession</td>
<td>0.002 in. (characteristic)</td>
</tr>
<tr>
<td>Display</td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>21.3 cm (8.4 in) diagonal color active matrix LCD; 640 (horizontal) X 480 (vertical) resolution</td>
</tr>
<tr>
<td>Refresh Rate</td>
<td>Vertical 59.83 Hz; Horizontal 31.41 kHz</td>
</tr>
<tr>
<td>Pixels</td>
<td>When running the analyzer's built-in Display Test, one or more of the following symptoms indicate a faulty display assembly:</td>
</tr>
<tr>
<td></td>
<td>- A complete row or column of &quot;stuck on&quot; or &quot;dark&quot; pixels.</td>
</tr>
<tr>
<td></td>
<td>- More than six &quot;stuck on&quot; pixels (but not more than three green)</td>
</tr>
<tr>
<td></td>
<td>- More than twelve &quot;dark&quot; pixels (but not more than seven of the same color)</td>
</tr>
<tr>
<td></td>
<td>- Two or more consecutive &quot;stuck on&quot; pixels or three or more consecutive &quot;dark&quot; pixels</td>
</tr>
<tr>
<td></td>
<td>- &quot;Stuck on&quot; or &quot;dark&quot; pixels less than 6.5 mm apart (excluding consecutive pixels)</td>
</tr>
<tr>
<td>Display Range</td>
<td></td>
</tr>
<tr>
<td>Magnitude</td>
<td>±500 dB (at 20 dB/div), max</td>
</tr>
<tr>
<td>Phase</td>
<td>±500°, max</td>
</tr>
</tbody>
</table>
| Polar | 10 pUnits, min  
1000 Units, max |
|-------|----------------|

**Display Resolution**

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magnitude</td>
<td>0.001 dB/div, min</td>
</tr>
<tr>
<td>Phase</td>
<td>0.01°/div, min</td>
</tr>
</tbody>
</table>

**Marker Resolution**

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magnitude</td>
<td>0.001 dB, min</td>
</tr>
<tr>
<td>Phase</td>
<td>0.01°, min</td>
</tr>
<tr>
<td>Polar</td>
<td>0.01 mUnit, min; 0.01°,min</td>
</tr>
</tbody>
</table>

**Table 22. Rear Panel Information**

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>10 MHz Reference In</strong></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Input Frequency</td>
<td>10 MHz ± 10 ppm, Typical</td>
</tr>
<tr>
<td>Input Level</td>
<td>-15 dBm to +20 dBm, Typical</td>
</tr>
<tr>
<td>Input Impedance</td>
<td>200 Ω, nom.</td>
</tr>
<tr>
<td><strong>10 MHz Reference Out</strong></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Output Frequency</td>
<td>10 MHz ± 1 ppm, Typical</td>
</tr>
<tr>
<td>Signal Type</td>
<td>Sine Wave, Typical</td>
</tr>
<tr>
<td>Output Level</td>
<td>+10 dBm ± 4 dB into 50 Ω, Typical</td>
</tr>
<tr>
<td>Output Impedance</td>
<td>50 Ω, nominal</td>
</tr>
<tr>
<td>Harmonics</td>
<td>&lt;-40 dBC, Typical</td>
</tr>
<tr>
<td><strong>VGA Video Output</strong></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>15-pin mini D-Sub; Drives VGA compatible monitors</td>
</tr>
<tr>
<td>Devices Supported:</td>
<td></td>
</tr>
<tr>
<td><strong>Resolutions:</strong></td>
<td></td>
</tr>
<tr>
<td>Flat Panel (TFT)</td>
<td>1024 X 768, 800 X 600, 640 X 480</td>
</tr>
<tr>
<td>Flat Panel (DSTN)</td>
<td>800 X 600, 640 X 480</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>CRT Monitor</td>
<td>1280 X 1024, 1024 X 768, 800 X 600, 640 X 480</td>
</tr>
<tr>
<td></td>
<td>Simultaneous operation of the internal and external displays is allowed, but with 640 X 480 resolution only. If you change resolution, you can only view the external display (internal display will &quot;white out&quot;).</td>
</tr>
<tr>
<td>Test Set IO</td>
<td>25-pin D-Sub connector, available for external test set control</td>
</tr>
<tr>
<td>Aux IO</td>
<td>25-pin D-Sub connector, male, analog and digital IO</td>
</tr>
<tr>
<td>Handler IO</td>
<td>36-pin parallel I/O port; all input/output signals are default set to negative logic; can be reset to positive logic via GPIB command</td>
</tr>
<tr>
<td>GPIB</td>
<td>24-pin D-sub (Type D-24), female; compatible with IEEE-488.</td>
</tr>
<tr>
<td>Parallel Port (LPT1)</td>
<td>25-pin D-S sub miniature connector, female; provides connection to printers or any other parallel port peripherals</td>
</tr>
<tr>
<td>Serial Port (COM 1)</td>
<td>9-pin D-Sub, male; compatible with RS-232</td>
</tr>
<tr>
<td>USB Port</td>
<td>One port on front panel and five ports on rear panel. Universal Serial Bus jack, Type A configuration (4 contacts inline, contact 1 on left); female</td>
</tr>
<tr>
<td>Contact 1</td>
<td>Vcc: 4.75 to 5.25 VDC, 500 mA, maximum</td>
</tr>
<tr>
<td>Contact 2</td>
<td>-Data</td>
</tr>
<tr>
<td>Contact 3</td>
<td>+Data</td>
</tr>
<tr>
<td>Contact 4</td>
<td>Ground</td>
</tr>
</tbody>
</table>

**Table 22. Rear Panel Information (Continued)**
<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LAN</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10/100BaseT Ethernet, 8-pin configuration; auto selects between the two data rates</td>
</tr>
<tr>
<td><strong>Line Power</strong></td>
<td></td>
</tr>
<tr>
<td>Frequency, Voltage</td>
<td>50/60/400 Hz for 100 - 120 V</td>
</tr>
<tr>
<td></td>
<td>50/60 Hz for 220 - 240 V</td>
</tr>
<tr>
<td></td>
<td>Power supply is auto switching</td>
</tr>
<tr>
<td>Max</td>
<td>350 Watts</td>
</tr>
</tbody>
</table>

**Note:** Option H08 and Option H11 are not available with the N5230A

### Table 23. Analyzer Dimensions and Weight

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cabinet Dimensions</strong></td>
<td></td>
</tr>
<tr>
<td>Height</td>
<td>Width</td>
</tr>
<tr>
<td>Excluding front and rear panel hardware and feet</td>
<td>267 mm</td>
</tr>
<tr>
<td>As shipped - includes front panel connectors, rear panel bumpers, and feet.</td>
<td>280 mm</td>
</tr>
<tr>
<td>As shipped plus handles</td>
<td>280 mm</td>
</tr>
<tr>
<td>As shipped plus rack-mount flanges</td>
<td>280 mm</td>
</tr>
<tr>
<td>As shipped plus handles and rack-mount flanges</td>
<td>280 mm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Weight</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Net</td>
<td>N5230A 24.9 kg (55 lb), nominal</td>
</tr>
<tr>
<td>Shipping</td>
<td>N5230A 36.3 kg (80 lb), nominal</td>
</tr>
</tbody>
</table>

Measurement Throughput Summary

- **Typical Cycle Time for Measurement Completion**
- **Cycle Time vs IF Bandwidth**
- **Cycle Time vs Number of Points**
- **Data Transfer Time**

### Table 24. Typical Cycle Time\(^a\) (ms) for Measurement Completion

| Description                                                                 | Typical |  |
|---|---|---|---|---|---|
| Number of Points | Typical |  |
| 201 | 401 | 801 | 1601 | 16,001 |

| Start 8 GHz, Stop 18 GHz, 30 kHz IF bandwidth |
|---|---|---|---|---|
| Uncorrected | 97.5 | 102.7 | 103.8 | 108.2 | 683.9 |
| 2-Port cal | 203.7 | 213.5 | 218.5 | 234.6 | 1504.3 |

| Start 10 MHz, Stop 10 GHz, 30 kHz IF bandwidth |
|---|---|---|---|---|
| Uncorrected | 112.6 | 120.6 | 124.8 | 138.2 | 738.4 |
| 2-Port cal | 232.8 | 251.8 | 265.2 | 304.3 | 1623.4 |

| Start 10 MHz, Stop 20 GHz, 30 kHz IF bandwidth |
|---|---|---|---|---|
| Uncorrected | 146 | 199.3 | 210.9 | 217.2 | 753.9 |
| 2-Port cal | 302.3 | 410.5 | 438.7 | 462.5 | 1660.5 |

| Start 8 GHz, Stop 18 GHz, 50 kHz IF bandwidth |
|---|---|---|---|---|
| Uncorrected | 79.1 | 81 | 81.7 | 86.6 | 482 |
| 2-Port cal | 164.5 | 170.3 | 175.3 | 193.5 | 1104.7 |

| Start 10 MHz, Stop 10 GHz, 50 kHz IF bandwidth |
|---|---|---|---|---|
| Uncorrected | 96.8 | 101.7 | 108.8 | 122.2 | 524.6 |
| 2-Port cal | 202.1 | 215.6 | 236.7 | 276.7 | 1198.8 |

| Start 10 MHz, Stop 20 GHz, 50 kHz IF bandwidth |
|---|---|---|---|---|

2946
2-Port cal & 293.6 & 341 & 360 & 389.5 & 1248.8 \\

Includes sweep time, retrace time and band-crossing time. Analyzer display turned off with
DISPLAY:ENABLE OFF. Add 21 ms for display on. Data for one trace (S11) measurement.

**Table 25. (Options 020/025, 120/125, only) Cycle Time vs IF Bandwidth**

Applies to the **Preset condition** (201 points, correction off) except for the following changes:

- CF = 10 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IF Bandwidth (Hz)</strong></td>
<td><strong>Cycle Time (ms)</strong></td>
</tr>
<tr>
<td>600,000</td>
<td>7</td>
</tr>
<tr>
<td>360,000</td>
<td>7</td>
</tr>
<tr>
<td>280,000</td>
<td>7</td>
</tr>
<tr>
<td>200,000</td>
<td>7</td>
</tr>
<tr>
<td>150,000</td>
<td>7</td>
</tr>
<tr>
<td>100,000</td>
<td>7</td>
</tr>
<tr>
<td>70,000</td>
<td>7</td>
</tr>
<tr>
<td>50,000</td>
<td>9</td>
</tr>
<tr>
<td>30,000</td>
<td>11</td>
</tr>
<tr>
<td>20,000</td>
<td>14</td>
</tr>
<tr>
<td>15,000</td>
<td>17</td>
</tr>
<tr>
<td>10,000</td>
<td>28</td>
</tr>
<tr>
<td>7000</td>
<td>37</td>
</tr>
<tr>
<td>5000</td>
<td>48</td>
</tr>
<tr>
<td>Description</td>
<td>Typical</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>IF Bandwidth (Hz)</td>
<td>Cycle Time (ms)(^a)</td>
</tr>
<tr>
<td></td>
<td>Option 080 enabled</td>
</tr>
<tr>
<td>250,000</td>
<td>8.9</td>
</tr>
<tr>
<td>200,000</td>
<td>9.3</td>
</tr>
<tr>
<td>150,000</td>
<td>9.9</td>
</tr>
</tbody>
</table>

\(^a\) Cycle time includes sweep and retrace time.

**Table 26. (Options 220/225, 420/425, 520/525 only) Cycle Time vs IF Bandwidth**

Applies to the Preset condition (201 points, correction off) except for the following changes:

- CF = 10 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)
<table>
<thead>
<tr>
<th>Value</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>100,000</td>
<td>10.5</td>
<td>41.8</td>
</tr>
<tr>
<td>70,000</td>
<td>11.5</td>
<td>43.6</td>
</tr>
<tr>
<td>50,000</td>
<td>12.8</td>
<td>45.4</td>
</tr>
<tr>
<td>30,000</td>
<td>15.4</td>
<td>50</td>
</tr>
<tr>
<td>20,000</td>
<td>18.3</td>
<td>53.9</td>
</tr>
<tr>
<td>15,000</td>
<td>21</td>
<td>57.5</td>
</tr>
<tr>
<td>10,000</td>
<td>27</td>
<td>65.8</td>
</tr>
<tr>
<td>7000</td>
<td>34</td>
<td>75.4</td>
</tr>
<tr>
<td>5000</td>
<td>48.5</td>
<td>93</td>
</tr>
<tr>
<td>3000</td>
<td>72.8</td>
<td>124</td>
</tr>
<tr>
<td>2000</td>
<td>108.8</td>
<td>169</td>
</tr>
<tr>
<td>1500</td>
<td>126.8</td>
<td>187.1</td>
</tr>
<tr>
<td>1000</td>
<td></td>
<td>272.5</td>
</tr>
<tr>
<td>700</td>
<td></td>
<td>357.7</td>
</tr>
<tr>
<td>500</td>
<td></td>
<td>460</td>
</tr>
<tr>
<td>300</td>
<td></td>
<td>697.7</td>
</tr>
<tr>
<td>200</td>
<td></td>
<td>1003.5</td>
</tr>
<tr>
<td>150</td>
<td></td>
<td>1307.8</td>
</tr>
<tr>
<td>100</td>
<td></td>
<td>1917.6</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>6173.8</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>18214.8</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>181699.2</td>
</tr>
</tbody>
</table>

a Cycle time includes sweep and retrace time.
Table 27. *(Options 020/025, 120/125, only)* Cycle Time vs Number of Points

Applies to the *Preset condition* (correction off) except for the following changes:

- CF = 10 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IF Bandwidth (Hz)</strong></td>
<td><strong>Number of Points</strong></td>
</tr>
<tr>
<td>30,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>401</td>
</tr>
<tr>
<td></td>
<td>801</td>
</tr>
<tr>
<td></td>
<td>1,601</td>
</tr>
<tr>
<td></td>
<td>6,401</td>
</tr>
<tr>
<td></td>
<td>16,001</td>
</tr>
<tr>
<td>100,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>401</td>
</tr>
<tr>
<td></td>
<td>801</td>
</tr>
<tr>
<td></td>
<td>1,601</td>
</tr>
<tr>
<td></td>
<td>6,401</td>
</tr>
<tr>
<td></td>
<td>16,001</td>
</tr>
<tr>
<td>600,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
</tbody>
</table>
a Cycle time includes sweep and retrace time.

**Table 28. (Options 220/225, 420/425, 520/525 only) Cycle Time vs Number of Points**

Applies to the [Preset condition](#) (correction off) except for the following changes:

- CF = 10 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF Bandwidth (Hz)</td>
<td>Number of Points</td>
</tr>
<tr>
<td>30,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>401</td>
</tr>
<tr>
<td></td>
<td>801</td>
</tr>
<tr>
<td></td>
<td>1,601</td>
</tr>
<tr>
<td></td>
<td>6,401</td>
</tr>
<tr>
<td></td>
<td>16,001</td>
</tr>
<tr>
<td>50,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Description</td>
<td>Typical</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td><strong>Number of Points</strong></td>
<td></td>
</tr>
<tr>
<td>201</td>
<td>401</td>
</tr>
</tbody>
</table>

**SCPI over GPIB**
(program executed on external PC)

| 32-bit floating point       | 7       | 12     | 43     | 435     |
| 64-bit floating point       | 12      | 22     | 84     | 856     |
| ASCII                       | 64      | 124    | 489    | 5054    |

**SCPI**
(program executed in the analyzer)

| 32-bit floating point       | 1       | 2      | 3      | 30      |

a Cycle time includes sweep and retrace time.

**Table 29. Data Transfer Time (ms)**
<table>
<thead>
<tr>
<th>64-bit floating point</th>
<th>2</th>
<th>2</th>
<th>4</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>29</td>
<td>56</td>
<td>222</td>
<td>2220</td>
</tr>
</tbody>
</table>

**COM (program executed in the analyzer)**

<table>
<thead>
<tr>
<th>32-bit floating point</th>
<th>&lt;0.4</th>
<th>0.4</th>
<th>0.5</th>
<th>1.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variant type</td>
<td>0.7</td>
<td>1</td>
<td>3</td>
<td>32</td>
</tr>
</tbody>
</table>

**DCOM over LAN**
**program executed on external PC**

<table>
<thead>
<tr>
<th>32-bit floating point</th>
<th>&lt;0.8</th>
<th>1</th>
<th>1.5</th>
<th>7.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variant type</td>
<td>1.8</td>
<td>2.7</td>
<td>8.5</td>
<td>80</td>
</tr>
</tbody>
</table>

**Tables 30 - 35** [Front-panel Jumper Specs (Options 025, 125, 225, 425, 525)]

**Test Set Block Diagrams**

**N5230A Option 020, or 120, or 220, or 420, or 520 (Standard Test Set and Standard Power Range)**

**N5230A Option 025, or 125, or 225, or 425, or 525 (Configurable Test Set and Extended Power Range)**
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

Last modified:

Jan. 10, 2007  Updated line power; moved regulatory, operating, and non-operating info to I&QS Guide.
Technical Specifications for the N5230A
Options 140/145/146/240/245/246 (4-Port PNA-L )
Rev. 2007-01-10

This is a complete list of the N5230A Options 140, 145, 146, 240, 245, 246 network analyzer technical specifications.

- To optimize viewing of uncertainty curves, click the Maximize button.
- To view or print the .pdf version of the specifications, visit our web site at www.agilent.com, and search for "Technical Specifications for the N5230A".
- This N5230A document provides technical specifications for the 85052B calibration kit and the N4433A ECal module. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the curves for your calibration kit and PNA setup.

- Definitions
  - Corrected System Performance
    - System Dynamic Range
    - Extended Dynamic Range
    - 3.5mm Connectors
  - Uncorrected System Performance
    - Test Port Output
    - Test Port Input
    - Dynamic Accuracy
    - Group Delay
  - General Information
  - Measurement Throughput Summary
  - Front-panel Jumper Specs (Option 145 or 146 or 245 or 246 only)
  - Test Set Block Diagrams

See Specs for other PNA models
Definitions

All specifications and characteristics apply over a 25 °C ±5 °C range (unless otherwise stated) and 90 minutes after the instrument has been turned on.

**Specification**(spec.): Warranted performance. Specifications include guardbands to account for the expected statistical performance distribution, measurement uncertainties, and changes in performance due to environmental conditions.

**Characteristic**(char.): A performance parameter that the product is expected to meet before it leaves the factory, but that is not verified in the field and is not covered by the product warranty. A characteristic includes the same guardbands as a specification.

**Typical**(typ.): Expected performance of an average unit which does not include guardbands. It is not covered by the product warranty.

**Nominal**(nom.): A general, descriptive term that does not imply a level of performance. It is not covered by the product warranty.

**Calibration**: The process of measuring known standards to characterize a network analyzer's systematic (repeatable) errors.

**Corrected** (residual): Indicates performance after error correction (calibration). It is determined by the quality of calibration standards and how well "known" they are, plus system repeatability, stability, and noise.


**Standard**: When referring to the analyzer, this includes no options unless noted otherwise.

Corrected System Performance

The specifications in this section apply for measurements made with the N5230A analyzer with the following conditions:

- 10 Hz IF bandwidth
- No averaging applied to data
- Isolation calibration with an averaging factor of 8

- **System Dynamic Range**
- **Extended Dynamic Range**
- **3.5mm Connectors**

**Table 1. System Dynamic Range at Test Port**

2956
<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB) at Test Port</th>
<th>Typical (dB) at Test Port</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Standard configuration and standard power range (Options 140 or 240)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 KHz to 10 MHz(^2)</td>
<td>--</td>
<td>108 dB</td>
</tr>
<tr>
<td>10 MHz to 4 GHz(^2)</td>
<td>120 dB</td>
<td>128 dB</td>
</tr>
<tr>
<td>4 GHz to 6 GHz</td>
<td>118 dB</td>
<td>129 dB</td>
</tr>
<tr>
<td>6 GHz to 10.5 GHz</td>
<td>115 dB</td>
<td>127 dB</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>107 dB</td>
<td>119 dB</td>
</tr>
<tr>
<td>13.5 GHz to 15 GHz</td>
<td>107 dB</td>
<td>119 dB</td>
</tr>
<tr>
<td>15 GHz to 20 GHz</td>
<td>103 dB</td>
<td>116 dB</td>
</tr>
<tr>
<td><strong>Configurable test set and extended power range (Options 145 or 245)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Configurable test set, extended power range, and internal second source (Option 146 or 246)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 KHz to 10 MHz(^2)</td>
<td>--</td>
<td>108 dB</td>
</tr>
<tr>
<td>10 MHz to 4 GHz(^2)</td>
<td>120 dB</td>
<td>128 dB</td>
</tr>
<tr>
<td>4 GHz to 6 GHz</td>
<td>118 dB</td>
<td>128 dB</td>
</tr>
<tr>
<td>6 GHz to 10.5 GHz</td>
<td>113 dB</td>
<td>125 dB</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>105 dB</td>
<td>117 dB</td>
</tr>
<tr>
<td>13.5 GHz to 15 GHz</td>
<td>105 dB</td>
<td>117 dB</td>
</tr>
<tr>
<td>15 GHz to 20 GHz</td>
<td>98 dB</td>
<td>115 dB</td>
</tr>
</tbody>
</table>

1 The system dynamic range is calculated as the difference between the noise floor and the specified source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account.
2 May be degraded by 10 dB at particular frequencies (multiples of 5 MHz) below 500 MHz due to spurious receiver residuals. Methods are available to regain the full dynamic range.

**Table 2. Extended Dynamic Range at Direct Receiver Access Input**
Configurable test set and extended power range (Options 145/245)  
Configurable test set, extended power range, and internal second source (Options 146/246)

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Specification (dB)</th>
<th>Typical (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz²</td>
<td>--</td>
<td>124 dB</td>
</tr>
<tr>
<td>10 MHz to 4 GHz²</td>
<td>136 dB</td>
<td>--</td>
</tr>
<tr>
<td>4 GHz to 6 GHz</td>
<td>134 dB</td>
<td>--</td>
</tr>
<tr>
<td>6 GHz to 10.5 GHz</td>
<td>129 dB</td>
<td>--</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>121 dB</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 15 GHz</td>
<td>121 dB</td>
<td>--</td>
</tr>
<tr>
<td>15 GHz to 20 GHz</td>
<td>114 dB</td>
<td>--</td>
</tr>
</tbody>
</table>

1 The direct receiver access input extended dynamic range is calculated as the difference between the direct receiver access input noise floor and the source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account. This set-up should only be used when the receiver input will never exceed its compression or damage level. When the analyzer is in segment sweep mode, it can have predefined frequency segments which will output a higher power level when the extended dynamic range is required (i.e. devices with high insertion loss), and reduced power when receiver compression or damage may occur (i.e. devices with low insertion loss). The extended range is only available in one-path transmission measurements.

2 May be degraded by 10 dB at particular frequencies (multiples of 5 MHz) below 500 MHz due to spurious receiver residuals. Methods are available to regain the full dynamic range.

Corrected System Performance with 3.5mm Connectors (Tables 3 - 4)

Receiver Dynamic Range technical specifications are not provided in this N5230A specs document.

**Table 5. Uncorrected System Performance**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Options 140/</td>
<td>Options 140/</td>
</tr>
<tr>
<td></td>
<td>145/ 146/ 240/</td>
<td>145/ 146/ 240/</td>
</tr>
<tr>
<td></td>
<td>240/ 245/ 246</td>
<td>245/ 246</td>
</tr>
</tbody>
</table>

**Directivity**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>-23 dB</td>
</tr>
<tr>
<td>10 MHz to 1 GHz</td>
<td>-28 dB</td>
<td>--</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Source Match</td>
<td>Load Match</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td>1 GHz to 3 GHz</td>
<td>-25 dB</td>
<td>--</td>
</tr>
<tr>
<td>3 GHz to 5 GHz</td>
<td>-20 dB</td>
<td>--</td>
</tr>
<tr>
<td>5 GHz to 11.5 GHz</td>
<td>-17 dB</td>
<td>--</td>
</tr>
<tr>
<td>11.5 GHz to 13.5 GHz</td>
<td>-15 dB</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>-15 dB</td>
<td>--</td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>-9 dB</td>
</tr>
<tr>
<td>10 MHz to 1 GHz</td>
<td>-12 dB</td>
<td>--</td>
</tr>
<tr>
<td>1 GHz to 3 GHz</td>
<td>-12 dB</td>
<td>--</td>
</tr>
<tr>
<td>3 GHz to 5 GHz</td>
<td>-12 dB</td>
<td>--</td>
</tr>
<tr>
<td>5 GHz to 10.5 GHz</td>
<td>-12 dB</td>
<td>--</td>
</tr>
<tr>
<td>10.5 GHz to 11.5 GHz</td>
<td>-10 dB</td>
<td>--</td>
</tr>
<tr>
<td>11.5 GHz to 13.5 GHz</td>
<td>-8 dB</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>-8 dB</td>
<td>--</td>
</tr>
<tr>
<td>300 KHz to 5 MHz</td>
<td>--</td>
<td>-9 dB</td>
</tr>
</tbody>
</table>

Source Match

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Source Match</th>
<th>Load Match</th>
<th>Crosstalk2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 GHz to 3 GHz</td>
<td>-25 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>3 GHz to 5 GHz</td>
<td>-20 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>5 GHz to 11.5 GHz</td>
<td>-17 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>11.5 GHz to 13.5 GHz</td>
<td>-15 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>-15 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>-9 dB</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 1 GHz</td>
<td>-12 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>1 GHz to 3 GHz</td>
<td>-12 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>3 GHz to 5 GHz</td>
<td>-12 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>5 GHz to 10.5 GHz</td>
<td>-12 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10.5 GHz to 11.5 GHz</td>
<td>-10 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>11.5 GHz to 13.5 GHz</td>
<td>-8 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>-8 dB</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

Load Match

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Source Match</th>
<th>Load Match</th>
<th>Crosstalk2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 GHz to 3 GHz</td>
<td>-25 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>3 GHz to 5 GHz</td>
<td>-20 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>5 GHz to 11.5 GHz</td>
<td>-17 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>11.5 GHz to 13.5 GHz</td>
<td>-15 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>-15 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>-9 dB</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 1 GHz</td>
<td>-20 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>1 GHz to 3 GHz</td>
<td>-20 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>3 GHz to 5 GHz</td>
<td>-18 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>5 GHz to 11.5 GHz</td>
<td>-12 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>11.5 GHz to 13.5 GHz</td>
<td>-7 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 16 GHz</td>
<td>-7 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-7.5 dB</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

Crosstalk²

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Source Match</th>
<th>Load Match</th>
<th>Crosstalk2</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 5 MHz</td>
<td>--</td>
<td>-9 dB</td>
<td>-70 dB</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Spec</td>
<td>Typical</td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>Options 140/ 145/ 146</td>
<td>300 KHz to 13.5 GHz</td>
<td>--</td>
<td></td>
</tr>
<tr>
<td>Options 240/ 245/ 246</td>
<td>300 KHz to 20 GHz</td>
<td>--</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nominal Power</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-5 dBm</td>
<td>-8 dBm</td>
</tr>
<tr>
<td></td>
<td>Preset power; attenuator switch point 10 dB below nominal power</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Frequency Resolution</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 Hz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CW Accuracy</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+/-1 ppm</td>
</tr>
</tbody>
</table>

1 Specifications apply over environmental temperature of 25 °C ±5 °C, with less than 1°C variation from the calibration temperature.

2 Measurement conditions: normalized to a thru, measured with two shorts, 10 Hz IF bandwidth, averaging factor of 8, alternate mode, source power set to the lesser of the maximum power out or the maximum receiver power.
# Frequency Stability

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>+/-0.05 ppm. -10°C to 70°C</td>
</tr>
</tbody>
</table>

## Power Level Accuracy

Variation from nominal power in range 0

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>--</th>
<th>--</th>
<th>+/-1.0 dB</th>
<th>+/-1.0 dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 2 GHz</td>
<td>+/-1.0 dB</td>
<td>+/-1.0 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 GHz to 10.5 GHz</td>
<td>+/-1.5 dB</td>
<td>+/-1.5 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>+/-2.5 dB</td>
<td>+/-2.5 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>+/-2.5 dB</td>
<td>+/-2.5 dB</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Max Leveled Power

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>--</th>
<th>--</th>
<th>+/-5 dBm</th>
<th>+/-5 dBm</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 4 GHz</td>
<td>+8 dBm</td>
<td>+8 dBm</td>
<td>+12 dBm</td>
<td>+11 dBm</td>
</tr>
<tr>
<td>4 GHz to 6 GHz</td>
<td>+6 dBm</td>
<td>+6 dBm</td>
<td>+10 dBm</td>
<td>+9 dBm</td>
</tr>
<tr>
<td>6 GHz to 10.5 GHz</td>
<td>+3 dBm</td>
<td>+1 dBm</td>
<td>+8 dBm</td>
<td>+6 dBm</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>0 dBm</td>
<td>-2 dBm</td>
<td>+5 dBm</td>
<td>+3 dBm</td>
</tr>
<tr>
<td>13.5 GHz to 15 GHz</td>
<td>0 dBm</td>
<td>-2 dBm</td>
<td>+5 dBm</td>
<td>+3 dBm</td>
</tr>
<tr>
<td>15 GHz to 20 GHz</td>
<td>-3 dBm</td>
<td>-8 dBm</td>
<td>+2 dBm</td>
<td>-1 dBm</td>
</tr>
</tbody>
</table>

## Power Level Linearity

Refer to footnote 1.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>--</th>
<th>--</th>
<th>+/-2.0 dB</th>
<th>+/-2.0 dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 1 GHz</td>
<td>+/-2.0 dB</td>
<td>+/-2.0 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Specification</td>
<td>Typical</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------</td>
<td>------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 GHz to 13.5 GHz</td>
<td>+/-1.5 dB</td>
<td>+/-1.5 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>+/-1.5 dB</td>
<td>+/-1.5 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Power Sweep Range (ALC)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALC range starts at maximum leveled power and decreases by the dB amount specified here.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>32 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 4 GHz</td>
<td>33 dB</td>
<td>33 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 GHz to 6 GHz</td>
<td>31 dB</td>
<td>31 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 GHz to 10.5 GHz</td>
<td>28 dB</td>
<td>26 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>25 dB</td>
<td>23 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13.5 GHz to 15 GHz</td>
<td>25 dB</td>
<td>23 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15 GHz to 20 GHz</td>
<td>22 dB</td>
<td>17 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Power Resolution</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.01 dB</td>
<td>0.01 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Power Range</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>-27 to +5 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +5 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 45MHz</td>
<td>--</td>
<td>-27 to +12 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +11 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>45 MHz to 4 GHz</td>
<td>--</td>
<td>-27 to +12 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +11 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 GHz to 6 GHz</td>
<td>--</td>
<td>-27 to +10 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +9 dB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 GHz to 10.5 GHz</td>
<td>--</td>
<td>-27 to +8 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>-87 to +6 dB</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Power Settings

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Minimum Power Setting</th>
<th>Maximum Power Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 15 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>15 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Power Setting</th>
<th>Minimum Power Setting</th>
<th>Maximum Power Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Power Setting</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Maximum Power Setting</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

### Harmonics (2nd or 3rd) at Maximum Output Power

#### In-band Source Harmonics

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Harmonic Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>-17 dBc</td>
</tr>
<tr>
<td>10 MHz to 1 GHz</td>
<td>-17 dBc</td>
</tr>
<tr>
<td>1 GHz to 13.5 GHz</td>
<td>-20 dBc</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>-20 dBc</td>
</tr>
</tbody>
</table>

#### Non-Harmonic Spurious (at Nominal Output Power)

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Harmonic Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 13.5 GHz</td>
<td>-50 dBc for offset frequen 1 KHz</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>-50 dBc for offset frequen 1 KHz</td>
</tr>
</tbody>
</table>

---

1 For Options 140/145/240/245, performance is specified on Port 1 only; Ports 2, 3, and 4 performance is typical. For Options 146/246, performance is specified on Ports 1 and 3 only; Ports 2 and 4 performance is typical. Test reference is at the nominal power level.
### Table 6. Test Port Output (Continued)

<table>
<thead>
<tr>
<th>Phase Noise (Nominal power at test port)</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Options 140/145/146/240/245/246</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>10 kHz Offset</strong></td>
</tr>
<tr>
<td></td>
<td><strong>100 kHz Offset</strong></td>
</tr>
<tr>
<td></td>
<td><strong>1 MHz Offset</strong></td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>-86 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-86 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-95 dBC/Hz</td>
</tr>
<tr>
<td>10 MHz to 1.5 GHz</td>
<td>-86 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-91 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-95 dBC/Hz</td>
</tr>
<tr>
<td>1.5 GHz to 3.125 GHz</td>
<td>-83 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-91 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-95 dBC/Hz</td>
</tr>
<tr>
<td>3.125 GHz to 6.25 GHz</td>
<td>-77 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-85 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-89 dBC/Hz</td>
</tr>
<tr>
<td>6.25 GHz to 12.5 GHz</td>
<td>-71 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-79 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-83 dBC/Hz</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>-65 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-73 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-77 dBC/Hz</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>-65 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-73 dBC/Hz</td>
</tr>
<tr>
<td></td>
<td>-77 dBC/Hz</td>
</tr>
</tbody>
</table>

### Table 7. Test Port Input

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Opt 140/ 145/ 146/ 240/ 245/ 246</strong></td>
<td><strong>Opt 140/ 145/ 146/ 240/ 245/ 246</strong></td>
</tr>
<tr>
<td><strong>Test Port Noise Floor</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total average (rms) noise power</td>
<td></td>
<td></td>
</tr>
<tr>
<td>calculated as the mean value of</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a linear magnitude trace</td>
<td></td>
<td></td>
</tr>
<tr>
<td>expressed in dBm.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>10 Hz IF Bandwidth</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>&lt;-103 dBm</td>
</tr>
<tr>
<td>10 MHz to 500 MHz</td>
<td>&lt;-112 dBm</td>
<td>&lt;-116 dBm</td>
</tr>
<tr>
<td>500 MHz to 4 GHz</td>
<td>&lt;-112 dBm</td>
<td>&lt;-120 dBm</td>
</tr>
<tr>
<td>4 GHz to 10.5 GHz</td>
<td>&lt;-112 dBm</td>
<td>&lt;-119 dBm</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Direct Receiver Access Input Noise Floor (Options 145/146/245/246)</td>
<td>Total average (rms) noise power calculated as the mean value of a linear magnitude trace expressed in dBm.</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>&lt;-107 dBm</td>
<td>&lt;-114 dBm</td>
</tr>
<tr>
<td>13.5 GHz to 15 GHz</td>
<td>&lt;-107 dBm</td>
<td>&lt;-114 dBm</td>
</tr>
<tr>
<td>15 GHz to 20 GHz</td>
<td>&lt;-106 dBm</td>
<td>&lt;-114 dBm</td>
</tr>
<tr>
<td><strong>1 KH z IF Bandwidth</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>&lt;-83 dBm</td>
</tr>
<tr>
<td>10 MHz to 500 MHz</td>
<td>&lt;-92 dBm</td>
<td>&lt;-96 dBm</td>
</tr>
<tr>
<td>500 MHz to 4 GHz</td>
<td>&lt;-92 dBm</td>
<td>&lt;-100 dBm</td>
</tr>
<tr>
<td>4 GHz to 10.5 GHz</td>
<td>&lt;-92 dBm</td>
<td>&lt;-99 dBm</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>&lt;-87 dBm</td>
<td>&lt;-94 dBm</td>
</tr>
<tr>
<td>13.5 GHz to 15 GHz</td>
<td>&lt;-87 dBm</td>
<td>&lt;-94 dBm</td>
</tr>
<tr>
<td>15 GHz to 20 GHz</td>
<td>&lt;-86 dBm</td>
<td>&lt;-94 dBm</td>
</tr>
<tr>
<td><strong>10 Hz IF Bandwidth</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>&lt;-119 dBm</td>
</tr>
<tr>
<td>10 MHz to 500 MHz</td>
<td>&lt;-128 dBm</td>
<td>&lt;-132 dBm</td>
</tr>
<tr>
<td>500 MHz to 4 GHz</td>
<td>&lt;-128 dBm</td>
<td>&lt;-136 dBm</td>
</tr>
<tr>
<td>4 GHz to 10.5 GHz</td>
<td>&lt;-128 dBm</td>
<td>&lt;-135 dBm</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>&lt;-123 dBm</td>
<td>&lt;-130 dBm</td>
</tr>
<tr>
<td>13.5 GHz to 15 GHz</td>
<td>&lt;-123 dBm</td>
<td>&lt;-130 dBm</td>
</tr>
<tr>
<td>15 GHz to 20 GHz</td>
<td>&lt;-122 dBm</td>
<td>&lt;-130 dBm</td>
</tr>
<tr>
<td><strong>1 KH z IF Bandwidth</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>&lt;-99 dBm</td>
</tr>
<tr>
<td>10 MHz to 500 MHz</td>
<td>&lt;-108 dBm</td>
<td>&lt;-112 dBm</td>
</tr>
<tr>
<td>500 MHz to 4 GHz</td>
<td>&lt;-108 dBm</td>
<td>&lt;-116 dBm</td>
</tr>
<tr>
<td>Description</td>
<td>Specification</td>
<td>Typical</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td></td>
<td>Opt 140/ 145/ 146/ 240/ 245/ 246</td>
<td>Opt 140/ 145/ 146/ 240/ 245/ 246</td>
</tr>
</tbody>
</table>

**Compression Level (at +8 dBm except as noted)**

<table>
<thead>
<tr>
<th></th>
<th>Power</th>
<th>Compression</th>
<th>Power</th>
<th>Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>--</td>
<td>+5 dBm</td>
<td>0.10 dB</td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>+8 dBm</td>
<td>0.35 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>50 MHz to 1 GHz</td>
<td>+8 dBm</td>
<td>0.35 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>1 GHz to 8 GHz</td>
<td>+8 dBm</td>
<td>0.25 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>8 GHz to 12.5 GHz</td>
<td>+8 dBm</td>
<td>0.30 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>12.5 GHz to 13.5 GHz</td>
<td>+8 dBm</td>
<td>0.55 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>+8 dBm</td>
<td>0.55 dB</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**Test Port Compression - 0.1 dB**

<table>
<thead>
<tr>
<th></th>
<th>Power</th>
<th>Compression</th>
<th>Power</th>
<th>Compression</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>--</td>
<td>+5 dBm</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 1 GHz</td>
<td>--</td>
<td>--</td>
<td>+9 dBm</td>
<td>--</td>
</tr>
<tr>
<td>1 GHz to 12.5 GHz</td>
<td>--</td>
<td>--</td>
<td>+10 dBm</td>
<td>--</td>
</tr>
</tbody>
</table>

Table 7. Test Port Input (Continued)
<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Opt 140/240</td>
<td>Opt 145/245</td>
</tr>
<tr>
<td></td>
<td>Opt 140/240</td>
<td>Opt 145/245</td>
</tr>
<tr>
<td>Trace Noise Magnitude</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ratioed measurement, nominal power at test port.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**100 KHz IF bandwidth**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 10.5 GHz</td>
<td>.006 dB rms</td>
<td>.008 dB rms</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>.010 dB rms</td>
<td>.014 dB rms</td>
</tr>
<tr>
<td>10.5 GHz to 20 GHz</td>
<td>.010 dB rms</td>
<td>.014 dB rms</td>
</tr>
</tbody>
</table>

**600 KHz IF bandwidth**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 10.5 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**100 KHz IF bandwidth**

Measured at Maximum Specified Power

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 2 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>Phase Error (deg rms)</td>
<td>Phase Error (deg rms)</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>2 GHz to 10.5 GHz</td>
<td>0.002 dB rms</td>
<td>0.003 dB rms</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>0.006 dB rms</td>
<td>0.009 dB rms</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>0.006 dB rms</td>
<td>0.009 dB rms</td>
</tr>
</tbody>
</table>

**Trace Noise Phase**

Ratioed measurement, nominal power at test port.

**100 KHz IF bandwidth**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Phase Error (deg rms)</th>
<th>Phase Error (deg rms)</th>
<th>Phase Error (deg rms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>0.110 deg rms</td>
<td>0.180 deg rms</td>
<td>0.280 deg rms</td>
</tr>
<tr>
<td>10 MHz to 10.5 GHz</td>
<td>0.050 deg rms</td>
<td>0.070 deg rms</td>
<td>0.125 deg rms</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>0.080 deg rms</td>
<td>0.100 deg rms</td>
<td>0.250 deg rms</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>0.080 deg rms</td>
<td>0.100 deg rms</td>
<td>0.250 deg rms</td>
</tr>
</tbody>
</table>

**600 KHz IF bandwidth**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Phase Error (deg rms)</th>
<th>Phase Error (deg rms)</th>
<th>Phase Error (deg rms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>0.110 deg rms</td>
<td>0.180 deg rms</td>
<td>0.300 deg rms</td>
</tr>
<tr>
<td>10 MHz to 10.5 GHz</td>
<td>0.080 deg rms</td>
<td>0.100 deg rms</td>
<td>0.200 deg rms</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>0.120 deg rms</td>
<td>0.160 deg rms</td>
<td>0.430 deg rms</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>0.120 deg rms</td>
<td>0.160 deg rms</td>
<td>0.430 deg rms</td>
</tr>
</tbody>
</table>

**100 KHz IF bandwidth**

Measured at Maximum Specified Power

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Phase Error (deg rms)</th>
<th>Phase Error (deg rms)</th>
<th>Phase Error (deg rms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>0.040 deg rms</td>
<td>0.050 deg rms</td>
<td>0.075 deg rms</td>
</tr>
<tr>
<td>Frequency Range</td>
<td>°C</td>
<td>°C</td>
<td>°C</td>
</tr>
<tr>
<td>----------------------</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>10 MHz to 2 GHz</td>
<td>--</td>
<td>--</td>
<td>0.007 deg rms</td>
</tr>
<tr>
<td>2 GHz to 10.5 GHz</td>
<td>--</td>
<td>--</td>
<td>0.012 deg rms</td>
</tr>
<tr>
<td>10.5 GHz to 13.5 GHz</td>
<td>--</td>
<td>--</td>
<td>0.040 deg rms</td>
</tr>
<tr>
<td>13.5 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
<td>0.040 deg rms</td>
</tr>
</tbody>
</table>

**Stability Magnitude**

Stability is defined as a ratio measurement made at the test port.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>°C</th>
<th>°C</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 2 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 4 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>4 GHz to 13.5 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>13.5 GHz to 16 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>16 GHz to 19 GHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>19 GHz to 20 GHz</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**Stability Phase**

Stability is defined as a ratio measurement made at the test port.

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>°C</th>
<th>°C</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 KHz to 10 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>10 MHz to 45 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Description</td>
<td>Specification</td>
<td>Typical</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td></td>
<td>Options 140/240</td>
<td>Opt 145/146/245</td>
</tr>
<tr>
<td><strong>Reference Level Magnitude</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Range</td>
<td>+/-200 dB</td>
<td>+/-200 dB</td>
</tr>
<tr>
<td>Resolution</td>
<td>0.001dB</td>
<td>0.001dB</td>
</tr>
<tr>
<td><strong>Reference Level Phase</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Range</td>
<td>+/-500&amp;&amp;176</td>
<td>+/-500&amp;&amp;176</td>
</tr>
<tr>
<td>Resolution</td>
<td>0.01&amp;&amp;176</td>
<td>0.01&amp;&amp;176</td>
</tr>
</tbody>
</table>
### Table 7. Test Port Input (Continued)

<table>
<thead>
<tr>
<th>Damage Input Level</th>
<th>Opt 140/ 240</th>
<th>Opt 145/ 146/ 245/ 246</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Port 1,2,3, and 4</td>
<td>+27 dBm or +/-16 VDC</td>
<td>+27 dBm or +/-16 VDC</td>
</tr>
<tr>
<td>Receivers R,A,B,C,D</td>
<td>--</td>
<td>+15 dBm or +/-16 VDC</td>
</tr>
<tr>
<td>Source out (reference)</td>
<td>--</td>
<td>+27 dBm or +/-16 VDC</td>
</tr>
<tr>
<td>Source out (test ports)</td>
<td>--</td>
<td>+27 dBm or +/-16 VDC</td>
</tr>
<tr>
<td>Coupler Thru</td>
<td>--</td>
<td>+27 dBm or +/-16 VDC</td>
</tr>
<tr>
<td>Coupler Arm</td>
<td>--</td>
<td>+15 dBm or +/-0 VDC</td>
</tr>
</tbody>
</table>

### Table 8. Dynamic Accuracy (Specification a)

Accuracy of the test port input power reading relative to the reference input power level.

**Dynamic Accuracy, 0.045 GHz**

![Magnitude Graph](image)

-10 dBm at 0.045 GHz
-20 dBm at 0.045 GHz
-30 dBm at 0.045 GHz
-40 dBm at 0.045 GHz
Dynamic Accuracy, 0.500 GHz

Phase

Dynamic Accuracy, 0.500 GHz

Magnitude
Dynamic Accuracy, 1-2 GHz

Magnitude

Phase
Dynamic Accuracy, 2 - 12.5 GHz

Magnitude

Phase
Dynamic Accuracy, 12.5 - 13.5 GHz

**Phase**

![Phase Graph](image)

**Magnitude**

![Magnitude Graph](image)
Dynamic Accuracy, 13.5-20 GHz
Dynamic accuracy is verified with the following measurements:

- compression over frequency
- IF linearity at a single frequency of 1.195 GHz using a reference level of -20 dBm for an input power range of 0 to -110 dBm.

### Table 9. Test Port Input (Group Delay)  

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information (typ.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aperture (selectable)</td>
<td>(frequency span)/(number of points -1)</td>
</tr>
<tr>
<td>Maximum Aperture</td>
<td>20% of frequency span</td>
</tr>
<tr>
<td>Range</td>
<td>0.5 x (1/minimum aperture)</td>
</tr>
<tr>
<td>Maximum Delay</td>
<td>Limited to measuring no more than 180° of phase change within the minimum aperture.</td>
</tr>
<tr>
<td>Accuracy</td>
<td>See graph below. Char.</td>
</tr>
</tbody>
</table>

The following graph shows characteristic group delay accuracy with full 2-port calibration and a 10 Hz IF bandwidth. Insertion loss is assumed to be < 2 dB and electrical length to be ten meters.

For any $S_{ij}$ Group Delay measurement, $S_{ii} = 0$, $S_{ij} = 1$, $S_{ji} = 0$, $S_{kl} = 0$ for all $k \neq $ $i$ $j$
In general, the following formula can be used to determine the accuracy, in seconds, of specific group delay measurement:

\[ \pm \frac{\text{Phase Accuracy (deg)}}{360 \times \text{Aperture (Hz)}} \]

Depending on the aperture and device length, the phase accuracy used is either incremental phase accuracy or worst case phase accuracy.

Group delay is computed by measuring the phase change within a specified frequency step (determined by the frequency span and the number of points per sweep).

### General Information

- **Miscellaneous Information**
- **Front Panel**
- **Rear Panel**
- **Environment and Dimensions**

#### Table 10. Miscellaneous Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>System IF Bandwidth Range</td>
<td>1 Hz to 600 kHz, nominal</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel® 1.1 GHz Pentium® M with 1 GByte RAM</td>
</tr>
</tbody>
</table>

#### Table 11. Front Panel Information
<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RF Connectors</strong></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Options 140/ 145/ 146/ 240/ 245/ 246: 3.5 mm (male), 50 ohm, (nominal)</td>
</tr>
<tr>
<td>Center Pin Recession</td>
<td>0.002 in. (characteristic)</td>
</tr>
<tr>
<td><strong>Display</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>The PNA display must remain in the 16 bit color setting in order to comply with international emissions regulations.</td>
</tr>
<tr>
<td>Size</td>
<td>21.3 cm (8.4 in) diagonal color active matrix LCD; 640 (horizontal) X 480 (vertical) resolution</td>
</tr>
<tr>
<td>Refresh Rate</td>
<td>Vertical 59.83 Hz; Horizontal 31.41 kHz</td>
</tr>
<tr>
<td>Pixels</td>
<td>When running the analyzer's built-in Display Test, one or more of the following symptoms indicate a faulty display assembly:</td>
</tr>
<tr>
<td></td>
<td>• A complete row or column of &quot;stuck on&quot; or &quot;dark&quot; pixels.</td>
</tr>
<tr>
<td></td>
<td>• More than six &quot;stuck on&quot; pixels (but not more than three green)</td>
</tr>
<tr>
<td></td>
<td>• More than twelve &quot;dark&quot; pixels (but not more than seven of the same color)</td>
</tr>
<tr>
<td></td>
<td>• Two or more consecutive &quot;stuck on&quot; pixels or three or more consecutive &quot;dark&quot; pixels (but no more than one set of two consecutive &quot;dark&quot; pixels)</td>
</tr>
<tr>
<td></td>
<td>• &quot;Stuck on&quot; or &quot;dark&quot; pixels less than 6.5 mm apart (excluding consecutive pixels)</td>
</tr>
<tr>
<td><strong>Display Range</strong></td>
<td></td>
</tr>
<tr>
<td>Magnitude</td>
<td>±500 dB (at 20 dB/div), max</td>
</tr>
<tr>
<td>Phase</td>
<td>±500°, max</td>
</tr>
<tr>
<td>Polar</td>
<td>10 pUnits, min</td>
</tr>
<tr>
<td></td>
<td>1000 Units, max</td>
</tr>
<tr>
<td><strong>Display Resolution</strong></td>
<td></td>
</tr>
<tr>
<td>Magnitude</td>
<td>0.001 dB/div, min</td>
</tr>
<tr>
<td>Phase</td>
<td>0.01°/div, min</td>
</tr>
<tr>
<td><strong>Marker Resolution</strong></td>
<td></td>
</tr>
<tr>
<td>Magnitude</td>
<td>0.001 dB, min</td>
</tr>
<tr>
<td>Phase</td>
<td>0.01°, min</td>
</tr>
<tr>
<td>Polar</td>
<td>0.01 mUnit, min; 0.01°, min</td>
</tr>
</tbody>
</table>
### Table 12. Rear Panel Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>10 MHz Reference In</strong></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Input Frequency</td>
<td>10 MHz ± 10 ppm, Typical</td>
</tr>
<tr>
<td>Input Level</td>
<td>-15 dBm to +20 dBm, Typical</td>
</tr>
<tr>
<td>Input Impedance</td>
<td>200 Ω, nom.</td>
</tr>
</tbody>
</table>

| **10 MHz Reference Out**     |                          |
| Connector                    | BNC, female              |
| Output Frequency             | 10 MHz ± 1 ppm, Typical  |
| Signal Type                  | Sine Wave, Typical       |
| Output Level                 | +10 dBm ± 4 dB into 50 Ω, Typical |
| Output Impedance             | 50 Ω, nominal            |
| Harmonics                    | < -40 dBc, Typical       |

| **VGA Video Output**         |                          |
| Connector                    | 15-pin mini D-Sub; Drives VGA compatible monitors |

<table>
<thead>
<tr>
<th>Devices Supported</th>
<th>Resolutions:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flat Panel (TFT)</td>
<td>1024 X 768, 800 X 600, 640 X 480</td>
</tr>
<tr>
<td>Flat Panel (DSTN)</td>
<td>800 X 600, 640 X 480</td>
</tr>
<tr>
<td>CRT Monitor</td>
<td>1280 X 1024, 1024 X 768, 800 X 600, 640 X 480</td>
</tr>
</tbody>
</table>

Simultaneous operation of the internal and external displays is allowed, but with 640 X 480 resolution only. If you change resolution, you can only view the external display (internal display will "white out").

<table>
<thead>
<tr>
<th><strong>Test Set IO</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Set IO</td>
<td>25-pin D-Sub connector, female, available for external test set control</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Aux IO</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Aux IO</td>
<td>25-pin D-Sub connector, male, analog and digital IO</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Handler IO</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Handler IO</td>
<td>36-pin parallel I/O port; all input/output signals are default set to negative logic; can be reset to positive logic via GPIB command female.</td>
</tr>
</tbody>
</table>
**GPIB (two ports - dedicated controller and dedicated talker/listener)**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>24-pin D-sub (Type D-24), female; compatible with IEEE-488.</td>
</tr>
</tbody>
</table>

**Parallel Port (LPT1)**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>25-pin D-Sub miniature connector, female; provides connection to printers or any other parallel port peripherals</td>
</tr>
</tbody>
</table>

**Serial Port (COM 1)**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>9-pin D-Sub, male; compatible with RS-232</td>
</tr>
</tbody>
</table>

**USB Port**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One port on front panel and five ports on rear panel. Universal Serial Bus jack, Type A configuration (4 contacts inline, contact 1 on left); female</td>
</tr>
<tr>
<td>Contact 1</td>
<td>Vcc: 4.75 to 5.25 VDC, 500 mA, maximum</td>
</tr>
<tr>
<td>Contact 2</td>
<td>-Data</td>
</tr>
<tr>
<td>Contact 3</td>
<td>+Data</td>
</tr>
<tr>
<td>Contact 4</td>
<td>Ground</td>
</tr>
</tbody>
</table>

**LAN**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10/100BaseT Ethernet, 8-pin configuration; auto selects between the two data rates</td>
</tr>
</tbody>
</table>

**Line Power**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency, Voltage</td>
<td>50/60/400 Hz for 100 - 120 V</td>
</tr>
<tr>
<td></td>
<td>50/60 Hz for 220 - 240 V</td>
</tr>
<tr>
<td></td>
<td>Power supply is auto switching</td>
</tr>
<tr>
<td>Max</td>
<td>350 Watts</td>
</tr>
</tbody>
</table>

**Note**: Option H08 and Option H11 are not available with the N5230A network analyzer.

**Table 13. Analyzer Dimensions and Weight**

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
</table>

2981
### Measurement Throughput Summary

- **Typical Cycle Time for Measurement Completion**
- **Cycle Time vs IF Bandwidth**
- **Cycle Time vs Number of Points**
- **Data Transfer Time**

#### Table 14. Typical Cycle Time \(^a\) (ms) for Measurement Completion

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Points</td>
<td></td>
</tr>
<tr>
<td>201</td>
<td>401</td>
</tr>
<tr>
<td>Start 8 GHz, Stop 18 GHz, 600 kHz IF bandwidth</td>
<td></td>
</tr>
<tr>
<td>Uncorrected</td>
<td>21.148</td>
</tr>
<tr>
<td>4-Port cal</td>
<td>74.597</td>
</tr>
<tr>
<td>Start 300 kHz, Stop 10 GHz, 600 kHz IF bandwidth</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Uncorrected</td>
<td>19.814</td>
</tr>
<tr>
<td>4-Port cal</td>
<td>69.752</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start 300 kHz, Stop 20 GHz, 600 kHz IF bandwidth</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncorrected</td>
<td>32.575</td>
<td>34.7</td>
<td>39.237</td>
<td>43.155</td>
<td>69.625</td>
</tr>
<tr>
<td>4-Port cal</td>
<td>121.254</td>
<td>133.626</td>
<td>157.506</td>
<td>179.223</td>
<td>487.779</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start 8 GHz, Stop 18 GHz, 100 kHz IF bandwidth</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncorrected</td>
<td>38.083</td>
<td>51.816</td>
<td>55.488</td>
<td>56.36</td>
<td>184.154</td>
</tr>
<tr>
<td>4-Port cal</td>
<td>143.271</td>
<td>201.814</td>
<td>215.056</td>
<td>230.133</td>
<td>934.161</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start 300 kHz, Stop 10 GHz, 100 kHz IF bandwidth</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncorrected</td>
<td>37.03</td>
<td>42.532</td>
<td>45.122</td>
<td>46.729</td>
<td>198.683</td>
</tr>
<tr>
<td>4-Port cal</td>
<td>137.431</td>
<td>162.37</td>
<td>194.13</td>
<td>192.182</td>
<td>906.768</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start 300 kHz, Stop 20 GHz, 100 kHz IF bandwidth</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncorrected</td>
<td>44.98</td>
<td>69.408</td>
<td>87.161</td>
<td>92.475</td>
<td>198.792</td>
</tr>
<tr>
<td>4-Port cal</td>
<td>169.041</td>
<td>268.877</td>
<td>343.898</td>
<td>369.526</td>
<td>914.963</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start 8 GHz, Stop 18 GHz, 50 kHz IF bandwidth</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncorrected</td>
<td>42.171</td>
<td>70.09</td>
<td>88.702</td>
<td>90.981</td>
<td>371.611</td>
</tr>
<tr>
<td>4-Port cal</td>
<td>157.107</td>
<td>271.791</td>
<td>351.517</td>
<td>368.02</td>
<td>1532.609</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start 300 kHz, Stop 10 GHz, 50 kHz IF bandwidth</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncorrected</td>
<td>43.713</td>
<td>61.41</td>
<td>66.878</td>
<td>69.373</td>
<td>385.04</td>
</tr>
<tr>
<td>4-Port cal</td>
<td>163.58</td>
<td>238.267</td>
<td>259.687</td>
<td>279.816</td>
<td>1580.761</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start 300 kHz, Stop 20 GHz, 50 kHz IF bandwidth</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncorrected</td>
<td>48.673</td>
<td>80.798</td>
<td>124.605</td>
<td>147.303</td>
<td>388.46</td>
</tr>
<tr>
<td>4-Port cal</td>
<td>184.429</td>
<td>313.392</td>
<td>493.142</td>
<td>587.548</td>
<td>1587.839</td>
</tr>
</tbody>
</table>

a Includes sweep time, retrace time and band-crossing time. Analyzer display turned off with DISPLAY:ENABLE OFF. Add 21 ms for display on. Data for one trace (S_{11}) measurement.
### Table 15. Cycle Time vs IF Bandwidth

Applies to the Preset condition (201 points, correction off) except for the following changes:

- CF = 10 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IF Bandwidth (Hz)</strong></td>
<td><strong>Cycle Time (ms)(^a)</strong></td>
</tr>
<tr>
<td>600,000</td>
<td>3.13</td>
</tr>
<tr>
<td>360,000</td>
<td>3.21</td>
</tr>
<tr>
<td>280,000</td>
<td>3.17</td>
</tr>
<tr>
<td>200,000</td>
<td>3.17</td>
</tr>
<tr>
<td>150,000</td>
<td>3.19</td>
</tr>
<tr>
<td>100,000</td>
<td>4.05</td>
</tr>
<tr>
<td>70,000</td>
<td>4.99</td>
</tr>
<tr>
<td>50,000</td>
<td>6.41</td>
</tr>
<tr>
<td>30,000</td>
<td>8.78</td>
</tr>
<tr>
<td>20,000</td>
<td>12.07</td>
</tr>
<tr>
<td>15,000</td>
<td>14.91</td>
</tr>
<tr>
<td>10,000</td>
<td>26.02</td>
</tr>
<tr>
<td>7000</td>
<td>34.54</td>
</tr>
<tr>
<td>5000</td>
<td>45.87</td>
</tr>
<tr>
<td>3000</td>
<td>69.91</td>
</tr>
<tr>
<td>2000</td>
<td>99.69</td>
</tr>
<tr>
<td>1500</td>
<td>128.18</td>
</tr>
<tr>
<td>1000</td>
<td>215.62</td>
</tr>
<tr>
<td>Number of Points</td>
<td>Cycle Time (ms)</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>1</td>
<td>0.00000</td>
</tr>
<tr>
<td>10</td>
<td>0.00000</td>
</tr>
<tr>
<td>30</td>
<td>0.00000</td>
</tr>
<tr>
<td>100</td>
<td>0.00000</td>
</tr>
<tr>
<td>200</td>
<td>0.00000</td>
</tr>
<tr>
<td>300</td>
<td>0.00000</td>
</tr>
<tr>
<td>500</td>
<td>0.00007</td>
</tr>
<tr>
<td>700</td>
<td>0.00006</td>
</tr>
</tbody>
</table>

a Cycle time includes sweep and retrace time.

**Table 16.** *Cycle Time vs Number of Points*

Applies to the Preset condition (correction off) except for the following changes:

- CF = 10 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IF Bandwidth (Hz)</strong></td>
<td><strong>Number of Points</strong></td>
</tr>
<tr>
<td>30,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>401</td>
</tr>
<tr>
<td></td>
<td>801</td>
</tr>
<tr>
<td></td>
<td>1,601</td>
</tr>
<tr>
<td></td>
<td>6,401</td>
</tr>
<tr>
<td></td>
<td>16,001</td>
</tr>
<tr>
<td>100,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-------</td>
</tr>
<tr>
<td>11</td>
<td>2.82</td>
</tr>
<tr>
<td>51</td>
<td>2.86</td>
</tr>
<tr>
<td>101</td>
<td>2.96</td>
</tr>
<tr>
<td>201</td>
<td>4.02</td>
</tr>
<tr>
<td>401</td>
<td>6.23</td>
</tr>
<tr>
<td>801</td>
<td>10.65</td>
</tr>
<tr>
<td>1,601</td>
<td>19.49</td>
</tr>
<tr>
<td>6,401</td>
<td>70.96</td>
</tr>
<tr>
<td>16,001</td>
<td>173.78</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>600,000</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2.84</td>
</tr>
<tr>
<td>11</td>
<td>2.84</td>
</tr>
<tr>
<td>51</td>
<td>2.87</td>
</tr>
<tr>
<td>101</td>
<td>3.03</td>
</tr>
<tr>
<td>201</td>
<td>3.14</td>
</tr>
<tr>
<td>401</td>
<td>3.51</td>
</tr>
<tr>
<td>801</td>
<td>4.22</td>
</tr>
<tr>
<td>1,601</td>
<td>6.22</td>
</tr>
<tr>
<td>6,401</td>
<td>19.35</td>
</tr>
<tr>
<td>16,001</td>
<td>45.12</td>
</tr>
</tbody>
</table>

a Cycle time includes sweep and retrace time.

*Table 17. Data Transfer Time (ms)*
<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number of Points</strong></td>
<td></td>
</tr>
<tr>
<td>201</td>
<td>401</td>
</tr>
</tbody>
</table>

**SCPI over GPIB**  
(program executed on external PC)

| 32-bit floating point | 6 | 10 | 33 | 313 |
| 64-bit floating point | 10 | 18 | 65 | 622 |
| ASCII | 53 | 105 | 406 | 4032 |

**SCPI over SiCL/LAN or TCP/IP Socket**  
(program executed in the analyzer)

| 32-bit floating point | 1 | 2 | 2.5 | 7 |
| 64-bit floating point | 2 | 2 | 3 | 10 |
| ASCII | 11 | 20 | 73 | 720 |

**COM**  
(program executed in the analyzer)

| 32-bit floating point | <0.2 | 0.2 | 0.3 | 0.9 |
| Variant type | 0.6 | 1 | 3.2 | 32 |

**DCOM over LAN**  
(program executed on external PC)

| 32-bit floating point | <0.8 | 1 | 1.6 | 7.5 |
| Variant type | 1.9 | 3 | 8.9 | 82 |

**Tables 18 - 23** Front-panel Jumper Specs (Option 145/146/245/246)

**Test Set Block Diagrams**

**N5230A Option 140/ 240 (Standard Test Set and Standard Power Range)**

2987
N5230A Option 145/ 245 (Configurable Test Set and Extended Power Range)

N5230A Option 146/ 246 (Configurable Test Set and Extended Power Range; Second Source)
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>RCVR R IN</td>
<td>h</td>
<td>CPLR THRU</td>
<td>o</td>
<td>RCVR D IN</td>
</tr>
<tr>
<td>b</td>
<td>SOURCE OUT</td>
<td>i</td>
<td>CPLR ARM</td>
<td>p</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>RCVR B IN</td>
<td>q</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>RCVR C IN</td>
<td>r</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>e</td>
<td>CPLR ARM</td>
<td>l</td>
<td>CPLR ARM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>RCVR A IN</td>
<td>m</td>
<td>CPLR THRU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>g</td>
<td>SOURCE OUT</td>
<td>n</td>
<td>SOURCE OUT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Last modified:

- Jan. 10, 2007: Updated line power, moved regulatory, operating, and non-operating info to I&QS Guide.
- Nov. 28, 2006: Updated for new model
- Nov. 6, 2006: Added display note
- Oct. 5, 2006: Added 350W typical to line power
- July 10, 2006: Previous version
This is a complete list of the technical specifications for the N5242A PNA-X network analyzer with the following options:

**Option 029**, adds hardware and firmware for high-accuracy noise figure measurements, utilizing source correction techniques. It requires use as an impedance tuner, and a 346-series noise source for calibration. See the block diagram.

**Option 200**, 2-port standard test set (includes six front-panel access loops) and power range. See the block diagram.

**Option 219**, adds 2-port extended power range, source and receiver attenuators, and bias-tees (requires Option 200). See the block diagram.

**Option 224**, adds an internal second source, a combiner, and mechanical switches to the 2-port analyzer (requires Option 200, 219, and 224). See the block diagram.

**Option 400**, 4-port standard test set (includes twelve front-panel access loops), power range, and an internal second source (Option 080 recommended). See the block diagram.

**Option 419**, adds 4-port extended power range, source and receiver attenuators, and bias-tees (requires Option 400). See the block diagram.

**Option 423**, adds an internal combiner, and mechanical switches to the 4-port analyzer (requires Option 400, 419, and 080). See the block diagram.

---

**Definitions**

**Corrected System Performance**

**Uncorrected System Performance**

**Test Port Input**

**Dynamic Accuracy**

**General Information**

**Measurement Throughput Summary**

**Specifications: Front-Panel Jumpers**

**Specifications: N5242A Option 029**

**Uncorrected System Performance, N5242A Option 029**

**Test Set Block Diagrams**

---

**Definitions**

All specifications and characteristics apply over a 25 °C ±5 °C range (unless otherwise stated) and 90 minutes after the instrument has been turned on.

**Specification (spec.):** Warranted performance. Specifications include guardbands to account for the expected statistical performance distribution, measurement uncertainties, and changes in performance due to environmental conditions.

**Characteristic (char.):** A performance parameter that the product is expected to meet before it leaves the factory, but
characteristic includes the same guardbands as a specification.

**Typical (typ.):** Expected performance of an average unit which does not include guardbands. It is not covered by the warranty.

**Nominal (nom.):** A general, descriptive term that does not imply a level of performance. It is not covered by the warranty.

**Calibration:** The process of measuring known standards to characterize a network analyzer's systematic (repeatable) errors.

**Corrected (residual):** Indicates performance after error correction (calibration). It is determined by the quality of calibration standards and how well they are known, plus system repeatability, stability, and noise.

**Uncorrected (raw):** Indicates instrument performance without error correction. The uncorrected performance affects the stability of a calibration.

**Standard:** When referring to the analyzer, this includes no options unless noted otherwise.
**Corrected System Performance**

The specifications in this section apply for measurements made with the N5242A analyzer with the following conditions:

- 10 Hz IF bandwidth
- No averaging applied to data
- Isolation calibration with an averaging factor of 8

**Table 1a. System Dynamic Range at Test Port**

Option 200 or 400

<table>
<thead>
<tr>
<th>Description</th>
<th>Port 1 or 3</th>
<th>Port 2 or 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>93</td>
<td>93</td>
</tr>
<tr>
<td>50 MHz to 100 MHz</td>
<td>103</td>
<td>103</td>
</tr>
<tr>
<td>100 MHz to 500 MHz</td>
<td>117</td>
<td>117</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>124</td>
<td>127</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>127</td>
<td>127</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>127</td>
<td>127</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>127</td>
<td>124</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>122</td>
<td>117</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>112</td>
<td>109</td>
</tr>
</tbody>
</table>

1. The system dynamic range is calculated as the difference between the noise floor and the specified source maximum output pov into account.
2. Either port can be used as the source port. Any other port can be used as the receiver port.
3. May typically be degraded at particular frequencies below 500 MHz due to spurious receiver residuals.

**Table 1b. System Dynamic Range at Test Port**

Option 219 or 419
The system dynamic range is calculated as the difference between the noise floor and the specified source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account.

Either port can be used as the source port. Any other port can be used as the receiver port.

May typically be degraded at particular frequencies below 500 M Hz due to spurious receiver residuals.

Table 1c. System Dynamic Range at Test Port

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB) at Test Port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Port 1 or 3 ²</td>
</tr>
<tr>
<td>10 M Hz to 50 M Hz³</td>
<td>93</td>
</tr>
<tr>
<td>50 M Hz to 100 M Hz³</td>
<td>103</td>
</tr>
<tr>
<td>100 M Hz to 500 M Hz³</td>
<td>117</td>
</tr>
<tr>
<td>500 M Hz to 3.2 GHz</td>
<td>124</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>127</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>126</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>124</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>118</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>110</td>
</tr>
</tbody>
</table>

1 The system dynamic range is calculated as the difference between the noise floor and the specified source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account.

2 Either port can be used as the source port. Any other port can be used as the receiver port.

3 May typically be degraded at particular frequencies below 500 M Hz due to spurious receiver residuals.

Table 1c. System Dynamic Range at Test Port

Option 224

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB) at Test Port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Source 2, Out 1</td>
</tr>
<tr>
<td>10 M Hz to 50 M Hz²</td>
<td>98</td>
</tr>
<tr>
<td>50 M Hz to 100 M Hz²</td>
<td>108</td>
</tr>
<tr>
<td>100 M Hz to 500 M Hz²</td>
<td>122</td>
</tr>
<tr>
<td>500 M Hz to 3.2 GHz</td>
<td>128</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>132</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>130</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>129</td>
</tr>
</tbody>
</table>
The system dynamic range is calculated as the difference between the noise floor and the specified source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account.

Either port can be used as the source port. Any other port can be used as the receiver port.

May typically be degraded at particular frequencies below 500 MHz due to spurious receiver residuals.

Table 1d. System Dynamic Range at Test Port

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB) at Test Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1 or 3</td>
<td>Port 2 or 4</td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>93</td>
</tr>
<tr>
<td>50 MHz to 100 MHz</td>
<td>103</td>
</tr>
<tr>
<td>100 MHz to 500 MHz</td>
<td>117</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>124</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>127</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>126</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>124</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>117</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>107</td>
</tr>
</tbody>
</table>

The system dynamic range is calculated as the difference between the noise floor and the specified source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account.

Either port can be used as the source port. Any other port can be used as the receiver port.

May typically be degraded at particular frequencies below 500 MHz due to spurious receiver residuals.
### Table 2b. Extended Dynamic Range at Direct Receiver Access Input

**Option 219 or 419**

<table>
<thead>
<tr>
<th>Description</th>
<th>Port 1 or 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>128</td>
</tr>
<tr>
<td>50 MHz to 100 MHz</td>
<td>115</td>
</tr>
<tr>
<td>100 MHz to 500 MHz</td>
<td>129</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>136</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>139</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>139</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>139</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>134</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>124</td>
</tr>
</tbody>
</table>

1 The direct receiver access input extended dynamic range is calculated as the difference between the direct receiver access input noise floor and the source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account. This set-up should only be used when the receiver input will never exceed its maximum receiver input. When the analyzer is in segment sweep mode, it can have predefined frequency segments which will output a higher power level when the extended dynamic range is required (i.e. devices with high insertion loss), and reduced power when the maximum receiver input level will occur (i.e. devices with low insertion loss). The extended range is only available in one-path transmission measurements.

2 Either port can be used as the source port. Any other port can be used as the receiver port.

3 May typically be degraded at particular frequencies below 500 MHz due to spurious receiver residuals.
The direct receiver access input extended dynamic range is calculated as the difference between the direct receiver access input noise floor and the source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account. This set-up should only be used when the receiver input will never exceed its maximum output power. When the analyzer is in segment sweep mode, it can have predefined frequency segments which will output a higher power level when the extended dynamic range is required (i.e. devices with high insertion loss), and reduced power when receiver compression or damage may occur (i.e. devices with low insertion loss). The extended range is only available in one-path transmission measurements.

Either port can be used as the source port. Any other port can be used as the receiver port.

May typically be degraded at particular frequencies below 500 MHz due to spurious receiver residuals.

Table 2c. Extended Dynamic Range at Direct Receiver Access Input

<table>
<thead>
<tr>
<th>Description</th>
<th>Source 2, Out 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>133</td>
</tr>
<tr>
<td>50 MHz to 100 MHz</td>
<td>120</td>
</tr>
<tr>
<td>100 MHz to 500 MHz</td>
<td>134</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>140</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>144</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>142</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>141</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>135</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>126</td>
</tr>
</tbody>
</table>

The direct receiver access input extended dynamic range is calculated as the difference between the direct receiver access input noise floor and the source maximum output power. This set-up should only be used when the receiver input will never exceed its maximum output power. When the analyzer is in segment sweep mode, it can have predefined frequency segments which will output a higher power level when the extended dynamic range is required (i.e. devices with high insertion loss), and reduced power when receiver compression or damage may occur (i.e. devices with low insertion loss). The extended range is only available in one-path transmission measurements.

May typically be degraded at particular frequencies below 500 MHz due to spurious receiver residuals.

Table 2d. Extended Dynamic Range at Direct Receiver Access Input

<table>
<thead>
<tr>
<th>Description</th>
<th>Source 2, Out 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>133</td>
</tr>
<tr>
<td>50 MHz to 100 MHz</td>
<td>120</td>
</tr>
<tr>
<td>100 MHz to 500 MHz</td>
<td>134</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>140</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>144</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>142</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>141</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>135</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>126</td>
</tr>
<tr>
<td>Description</td>
<td>Port 1 or 3</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>10 MHz to 50 MHz²</td>
<td>128</td>
</tr>
<tr>
<td>50 MHz to 100 MHz²</td>
<td>115</td>
</tr>
<tr>
<td>100 MHz to 500 MHz³</td>
<td>129</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>136</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>139</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>138</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>136</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>129</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>119</td>
</tr>
</tbody>
</table>

1 The direct receiver access input extended dynamic range is calculated as the difference between the direct receiver access input noise floor and the source maximum output power. The effective dynamic range must take measurement uncertainties and interfering signals into account. This set-up should only be used when the receiver input will never exceed its compression or damage level. When the analyzer is in segment sweep mode, it can have predefined frequency segments which will output a higher power level when the extended dynamic range is required (i.e. devices with high insertion loss), and reduced power when receiver compression or damage may occur (i.e. devices with low insertion loss). The extended range is only available in one-path transmission measurements.

2 Either port can be used as the source port. Any other port can be used as the receiver port.

3 May typically be degraded at particular frequencies below 500 MHz due to spurious receiver residuals.

Receiver Dynamic Range technical specifications are not provided in this N5242A specs document.

**N5242A Corrected System Performance with 3.5mm Connectors**

**All Options**

Note: For any Sii reflection measurement:

- $S_{jj} = 0$.

For any Sij transmission measurement:

- $S_{ji} = S_{ij}$ when $S_{ij} \leq 1$
- $S_{ji} = 1/S_{ij}$ when $S_{ij} > 1$
- $S_{kk} = 0$ for all $k$

**Table 3. 85052B Calibration Kit**

**N5242A All Options**
Applies to the N5242A Option 200 or 219 or 224 or 400 or 419 or 423 analyzers, 85052B (3.5mm) calibration kit, 85131F flexible test port cable set, and a full 2-port calibration. Also applies to the following condition:

Environmental temperature 23° ±3 °C, with < 1 °C deviation from calibration temperature

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 MHz to 500 MHz</td>
</tr>
<tr>
<td></td>
<td>500 MHz</td>
</tr>
<tr>
<td>Directivity</td>
<td>48</td>
</tr>
<tr>
<td>Source Match</td>
<td>40</td>
</tr>
<tr>
<td>Load Match</td>
<td>48</td>
</tr>
<tr>
<td>Reflection Tracking 1</td>
<td>±0.003</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
<tr>
<td>Transmission Tracking 1</td>
<td>±0.017</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
</tbody>
</table>

1 Temperature deviation is a characteristic value.

**Transmission Uncertainty (Specifications)**

**Magnitude**

![Graph showing transmission uncertainty for various frequency ranges]
Reflection Uncertainty (Specifications)

**Phase**

N5242A All Options Full Two Port Cal Using 85052B

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz
- 20 GHz to 26.5 GHz

![Phase Graph]

**Magnitude**

N5242A All Options with 85052B

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz
- 20 GHz to 26.5 GHz

![Magnitude Graph]
Table 4. N4433A 4-Port Electronic Calibration Module

N5242A All Options

**Note**: Uncertainty curves for the N4433A are created using a 2-port calibration. Multiport uncertainties are not supported at this time.

Applies to the N5242A Option 200 or 219 or 224 or 400 or 419 or 423 analyzers, N4433A (3.5mm) electronic calibration module, 85131F flexible test port cable set, and a full 2-port calibration. Also applies to the following condition:

Environmental temperature 23° ±3 °C, with < 1 °C deviation from calibration temperature

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 MHz to 500 MHz</td>
</tr>
<tr>
<td>Directivity</td>
<td>52</td>
</tr>
<tr>
<td>Source Match</td>
<td>42</td>
</tr>
<tr>
<td>Load Match</td>
<td>41</td>
</tr>
<tr>
<td>Reflection Tracking</td>
<td>±0.060</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
<tr>
<td>Transmission Tracking</td>
<td>±0.063</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
</tbody>
</table>

1 Temperature deviation is a characteristic value.
Reflection Uncertainty (Specifications)

**Magnitude**

N5242A All Options with N4433A

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz

S21 = S12 = 0
Source Power = -5 dBm

**Phase**

N5242A All Options with N4433A

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz

S21 = S12 = 0
Source Power = -5 dBm
Table 5. N4691B 2- Port Electronic Calibration Module

N5242A All Options

Applies to the N5242A Option 200 or 219 or 224 or 400 or 419 or 423 analyzers, N4691B (3.5mm) electronic calibration module, 85131F flexible test port cable set, and a full 2-port calibration. Also applies to the following condition:

Environmental temperature 23° ±3 °C, with < 1 °C deviation from calibration temperature

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 MHz to 500 MHz</td>
</tr>
<tr>
<td>Directivity</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>500 MHz to 2 GHz</td>
</tr>
<tr>
<td>Source Match</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td>2 GHz to 20 GHz</td>
</tr>
<tr>
<td>Load Match</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>20 GHz to 26.5 GHz</td>
</tr>
<tr>
<td>Reflection Tracking</td>
<td>±0.050 ±0.010/°C</td>
</tr>
<tr>
<td></td>
<td>±0.020 ±0.010/°C</td>
</tr>
<tr>
<td>Transmission Tracking</td>
<td>±0.056 ±0.010/°C</td>
</tr>
<tr>
<td></td>
<td>±0.022 ±0.010/°C</td>
</tr>
</tbody>
</table>

1 Temperature deviation is a characteristic value.
Transmission Uncertainty (Specifications)

**Magnitude**

- N5242A, All Options Full Two Port Cal Using N4691B

**Phase**

- N5242A, All Options Full Two Port Cal Using N4691B

---

Graphs showing uncertainty in transmission coefficient for different frequency ranges.
Reflection Uncertainty (Specifications)

This N5242A document does not present specifications for the 85052C or 85052D Calibration Kit. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the data and curves for the 85052C or the 85052D Calibration Kit.
Uncorrected System Performance

Table 6. Error Terms

Ports 1, 2, 3, 4. All Options. If Option 029 is included, see section Specifications: N5242A Option 029.

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Directivity (dB)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>16</td>
<td>23</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>24</td>
<td>28</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>24</td>
<td>32</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>23</td>
<td>25</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>16</td>
<td>22</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>16</td>
<td>22</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>16</td>
<td>22</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>16</td>
<td>22</td>
</tr>
<tr>
<td><strong>Source Match (dB)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>11</td>
<td>14</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>18</td>
<td>28</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>18</td>
<td>22</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>14</td>
<td>18</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>12</td>
<td>16</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>10</td>
<td>14</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>8</td>
<td>12</td>
</tr>
</tbody>
</table>
Table 6. (Continued) Error Terms

<table>
<thead>
<tr>
<th>Error Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Term 1</td>
<td>Explanation 1</td>
</tr>
<tr>
<td>Term 2</td>
<td>Explanation 2</td>
</tr>
<tr>
<td>Term 3</td>
<td>Explanation 3</td>
</tr>
</tbody>
</table>

1. Note: Additional notes or conditions may be provided in the context.
<table>
<thead>
<tr>
<th>Load Match (dB)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>11</td>
<td>18</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>17</td>
<td>25</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>17</td>
<td>22</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>13</td>
<td>17</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>9</td>
<td>14</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>9</td>
<td>14</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>8</td>
<td>13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transmission Tracking³ (dB)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reflection Tracking (dB)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 6. (Continued) Error Terms

<table>
<thead>
<tr>
<th>Crosstalk⁴ (dB)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>-84</td>
</tr>
<tr>
<td>50 MHz to 100 MHz</td>
<td></td>
<td>-90</td>
</tr>
<tr>
<td>100 MHz to 500 MHz</td>
<td></td>
<td>-110</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td></td>
<td>-120</td>
</tr>
<tr>
<td>3.2 GHz to 20 GHz</td>
<td></td>
<td>-122</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td></td>
<td>-117</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td></td>
<td>-114</td>
</tr>
</tbody>
</table>

¹ Specifications apply over environmental temperature of 25 °C ±5 °C, with less than 1°C variation from the calibration temperature.

³ Cable loss not included.

⁴ Measurement conditions: normalized to a thru, measured with shorts on all ports, 10 Hz IF bandwidth, averaging factor of 8, alternate power.

Test Port Output

Table 7. Frequency Information

All Options

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
<th>Typical (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency Range</td>
<td>10 MHz to 26.5 GHz</td>
<td>--</td>
</tr>
<tr>
<td>Frequency Resolution</td>
<td>1 Hz</td>
<td>--</td>
</tr>
<tr>
<td>Frequency Accuracy</td>
<td>+/- 1 ppm</td>
<td>--</td>
</tr>
</tbody>
</table>
| Frequency Stability       | --                 | +/- 0.05 ppm, -10° to 70° C
                          |                    | +/- 0.1 ppm/ yr maximum |

Table 8a. Maximum Leveled Power, Option 200 or 400
<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dBm)</th>
<th>Specification (dBm)</th>
<th>Specification (dBm)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Port 1 or 3(^1)</td>
<td>Port 1 or 3(^1)</td>
<td>Port 2 or 4(^1)</td>
</tr>
<tr>
<td></td>
<td>Filtered Mode(^2)</td>
<td>Hi Pwr Mode(^2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See Figure 2 (Opt 200) or Figure 5 (Opt 400)</td>
<td>See Figure 3 (Opt 200) or Figure 6 (Opt 400)</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>8</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>10</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>10</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>13</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>13</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>13</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>12</td>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>5</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

\(^1\) Either port can be used as the source port.

\(^2\) In Filtered Mode, the signal path goes through filters to minimize harmonics below 3.2 GHz. In Hi Pwr Mode, the signal bypasses the filters to maximize output.
Figure 1. Block Diagram, N5242A Option 200

Figure 2. Path Configuration Diagram, N5242A Option 200, Port 1 Filtered Mode
Figure 3. Path Configuration Diagram, N5242A Option 200, Port 1 Hi Pwr Mode

Figure 4. Block Diagram, N5242A Option 400
Figure 5. Path Configuration Diagram, N5242A Option 400, Port 1 or 3 Filtered Mode
Figure 6. Path Configuration Diagram, N5242A Option 400, Port 1 or 3 Hi Pwr Mode

Port 1 or 3 Filtered Mode
### Table 8b. Maximum Leveled Power, Option 219 or 419

If Option 029 is included, see section Specifications: N5242A Option 029.

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dBm)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Port 1 or 3&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>Filtered Mode&lt;sup&gt;2&lt;/sup&gt;</td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>8</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>10</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>10</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>13</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>12</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>10</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>8</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>3</td>
</tr>
</tbody>
</table>

<sup>1</sup> Either port can be used as the source port.

<sup>2</sup> In Filtered Mode, the signal path goes through filters to minimize harmonics below 3.2 GHz. In Hi Pwr Mode, the signal bypasses the filters.

#### Figure 7. Block Diagram, N5242A Option 219

![Block Diagram](image)

**Note:** The path configuration drawing for Option 219 is identical to the path configuration drawings for Option 200, which are shown in...
Figure 8. Block Diagram, N5242A Option 419

Note: The path configuration drawing for Option 419 is identical to the path configuration drawings for Option 400, which are shown in

Table 8c. Maximum Leveled Power, Option 224

If Option 029 is included, see section Specifications: N5242A Option 029.

<table>
<thead>
<tr>
<th>Description</th>
<th>Port 1 Filtered Mode ¹</th>
<th>Port 1 Hi Pwr Mode ¹</th>
<th>Port 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>7</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>8</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>8</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>13</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>12</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>10</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>7</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>0</td>
<td>0</td>
<td>-2</td>
</tr>
</tbody>
</table>

¹ In Filtered Mode, the signal path goes through filters to minimize harmonics below 3.2 GHz. In Hi Pwr Mode, the signal bypasses the filters.
Table 8d. Maximum Leveled Power, Option 224

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dBm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source 2, Out 1 Filtered Mode</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>9</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>11</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>10</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>18</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>16</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>15</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>13</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>7</td>
</tr>
<tr>
<td>Source 2, Out 1 Hi Pwr Mode</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>18</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>18</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>14</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>18</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>16</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>15</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>13</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>7</td>
</tr>
<tr>
<td>Source 2, Out 2</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>13</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>18</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>17</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>22</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>22</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>21</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>22</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>22</td>
</tr>
</tbody>
</table>

1 In Filtered Mode, the signal path goes through filters to minimize harmonics below 3.2 GHz. In Hi Pwr Mode, the signal bypasses these filters.

Figure 9. Block Diagram: N5242A Option 224
**Figure 10.** Path Configuration Diagram, N5242A Option 224

**Table 8e. Maximum Leveled Power, Option 423**

If Option 029 is included, see section Specifications: N5242A Option 029.

<table>
<thead>
<tr>
<th>Description</th>
<th>Port 1 or 3 Specification (dBm)</th>
<th>Port 1 or 3 Specification (dBm)</th>
<th>Port 2 or 4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Port 1 or 3 Filtered Mode¹</td>
<td>Port 1 or 3 Hi Pwr Mode¹</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>7</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>8</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>8</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>13</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>12</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>10</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>7</td>
<td>7</td>
<td>5</td>
</tr>
</tbody>
</table>

¹ In Filtered Mode, the signal path goes through filters to minimize harmonics below 3.2 GHz. In Hi Pwr Mode, the signal bypasses the filters to maximize output power.
In Filtered Mode, the signal path goes through filters to minimize harmonics below 3.2 GHz. In Hi Pwr Mode, the signal bypasses the
**Table 8f. Maximum Leveled Power, Option 224 or 423**

If Option 029 is included, see section Specifications: N5242A Option 029.

<table>
<thead>
<tr>
<th>Description</th>
<th>Source 1, Port 1 Combine Mode</th>
<th>Source 1, Port 1 Combine Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Filtered Mode</td>
<td>Hi Pwr Mode</td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>7</td>
<td>17</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>9</td>
<td>17</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

1 In Filtered Mode, the signal path goes through filters to minimize harmonics below 3.2 GHz. In Hi Pwr Mode, the signal bypasses the filters.

**Figure 11.** Block Diagram: N5242A Option 423
**Table 9a. Power Level Accuracy**

**All Options**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ports 1, 2, 3, 4(^1)</td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>+/-1.0</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>+/-1.0</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>+/-1.0</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>+/-1.0</td>
</tr>
<tr>
<td>10 GHz to 13 GHz</td>
<td>+/-1.2</td>
</tr>
<tr>
<td>13 GHz to 18 GHz</td>
<td>+/-2.0</td>
</tr>
<tr>
<td>18 GHz to 26.5 GHz</td>
<td>+/-2.5</td>
</tr>
</tbody>
</table>

\(^1\) Any port can be used as the source port. Source in filtered mode where applicable.
### Table 9b. Power Level Linearity

#### All Options

<table>
<thead>
<tr>
<th>Description</th>
<th>Port 1 or 3&lt;sup&gt;1&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-25dBm = P&lt;-20dBm</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>+/- 2.0</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>+/- 1.0</td>
</tr>
</tbody>
</table>

<sup>1</sup> Either port can be used as the source port. Source in filtered mode.

### Table 9c. (Continued) Power Level Linearity

#### All Options

<table>
<thead>
<tr>
<th>Description</th>
<th>Port 2 or 4&lt;sup&gt;1&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-25dBm = P&lt;-20dBm</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>+/- 5.0</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>+/- 4.0</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>+/- 2.5</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>+/- 2.0</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>+/- 1.0</td>
</tr>
</tbody>
</table>
Either port can be used as the source port.
**Table 9d. Power Level Linearity**

Option 224

<table>
<thead>
<tr>
<th>Description</th>
<th>Source 2, Out 1&lt;sup&gt;1&lt;/sup&gt; P = -15dBm</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>+/- 1.0</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>+/- 1.0</td>
</tr>
</tbody>
</table>

<sup>1</sup> Source in filtered mode.

**Table 10a. Power Sweep Range**

Option 200 or 400

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Port 1 or 3&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>38</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>38</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>35</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>38</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>38</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>38</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>37</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>30</td>
</tr>
</tbody>
</table>

<sup>1</sup> Either port can be used as the source port. Source in filtered mode where applicable.
### Table 10b. Power Sweep Range

**Option 219 or 419**

<table>
<thead>
<tr>
<th>Description</th>
<th>Port 1 or 3</th>
<th>Port 2 or 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>38</td>
<td>38</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>38</td>
<td>38</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>35</td>
<td>38</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>38</td>
<td>38</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>37</td>
<td>36</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>35</td>
<td>33</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>33</td>
<td>32</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>28</td>
<td>24</td>
</tr>
</tbody>
</table>

1 Either port can be used as the source port. Source in filtered mode where applicable.

### Table 10c. Power Sweep Range

**Option 224 or 423**

<table>
<thead>
<tr>
<th>Description</th>
<th>Port 1 or 3</th>
<th>Port 2 or 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>38</td>
<td>38</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>38</td>
<td>38</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>35</td>
<td>38</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>38</td>
<td>38</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>37</td>
<td>35</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>35</td>
<td>32</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>32</td>
<td>30</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>25</td>
<td>23</td>
</tr>
</tbody>
</table>

1 Either port can be used as the source port. Source in filtered mode where applicable.
### Table 10d. Power Sweep Range

**Option 224**

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
<th>Typical (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source 2(^1) Out 1</td>
<td>Source 2(^1) Out 2</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>33</td>
<td>28</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>33</td>
<td>32</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>29</td>
<td>29</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>33</td>
<td>33</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>31</td>
<td>31</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>30</td>
<td>28</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>28</td>
<td>27</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>22</td>
<td>20</td>
</tr>
</tbody>
</table>

\(^1\) Source in filtered mode where applicable.

### Table 11. Nominal Power (Preset Power)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dBm)</th>
<th>Option 200 or 400</th>
<th>Option 219 or 224 or 419 or 423</th>
<th>Option 224</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ports 1, 2, 3, 4(^1)</td>
<td>Source 2, Out 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 26.5 GHz</td>
<td>0</td>
<td>-5</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

\(^1\) Any port can be used as the source port. Any other port can be used as the receiver port.
Table 12. Power Resolution and Maximum/Minimum Settable Power

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification at Test Port</th>
<th>All Options</th>
<th>All Options</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>Power Resolution</td>
<td>0.01 dB</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Maximum Settable Power</td>
<td>--</td>
<td>30 dBm</td>
<td>--</td>
</tr>
<tr>
<td>Minimum Settable Power</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

1 Any port can be used as the source port.
Table 13. Harmonics at Max Specified Power

All Options

(See Tables 8a - 8f Maximum Leveled Power)

<table>
<thead>
<tr>
<th>Description</th>
<th>Port 1 or 3 $^1$, $^2$ Source 2 Out 1 $^3$</th>
<th>Port 2 or 4 $^1$ Source 2 Out 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2nd Harmonics $^4$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-51</td>
<td>-13</td>
</tr>
<tr>
<td>50 MHz to 2 GHz</td>
<td>-51</td>
<td>-13</td>
</tr>
<tr>
<td>2 GHz to 3.2 GHz</td>
<td>-60</td>
<td>-21</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-60</td>
<td>-21</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>-60</td>
<td>-21</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-60</td>
<td>-21</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-60</td>
<td>-21</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-60</td>
<td>-21</td>
</tr>
</tbody>
</table>

| 3rd Harmonics $^4$   |                                          |                                 |
| 10 MHz to 50 MHz     | -51                                      | -13                             |
| 50 MHz to 2 GHz      | -51                                      | -13                             |
| 2 GHz to 3.2 GHz     | -60                                      | -21                             |
| 3.2 GHz to 10 GHz    | -60                                      | -21                             |
| 10 GHz to 16 GHz     | -60                                      | -21                             |
| 16 GHz to 20 GHz     | -60                                      | -21                             |
| 20 GHz to 24 GHz     | -60                                      | -21                             |
| 24 GHz to 26.5 GHz   | -60                                      | -21                             |
Table 13. Harmonics at Max Specified Power (continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical (dBc) at Test Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/2 and 1/4 Sub-Harmonics 4</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-73</td>
</tr>
<tr>
<td>50 MHz to 2 GHz</td>
<td>-73</td>
</tr>
<tr>
<td>2 GHz to 3.2 GHz</td>
<td>-73</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-66</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>-66</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-66</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-61</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-61</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-63</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>-63</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-63</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-52</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-52</td>
</tr>
</tbody>
</table>

1 Any port can be used as the source port.
2 < 3.2 GHz Filtered Mode
3 At port 1 max specified power.
4 Listed frequency is fundamental frequency; test at max specified power

Table 14. Non-Harmonic Spurs at nominal power

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical (dBc) at Test Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ports 1, 2, 3, 4</td>
<td></td>
</tr>
<tr>
<td>Source 2 Out 1,</td>
<td></td>
</tr>
<tr>
<td>Source 2 Out 2</td>
<td></td>
</tr>
</tbody>
</table>

Offset frequency = 30 kHz to 5 MHz

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical (dBc) at Test Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 500 MHz</td>
<td>-50</td>
</tr>
<tr>
<td>500 MHz to 1 GHz</td>
<td>-60</td>
</tr>
<tr>
<td>1 GHz to 2 GHz</td>
<td>-60</td>
</tr>
<tr>
<td>2 GHz to 4 GHz</td>
<td>-57</td>
</tr>
<tr>
<td>4 GHz to 8 GHz</td>
<td>-51</td>
</tr>
<tr>
<td>8 GHz to 16 GHz</td>
<td>-45</td>
</tr>
<tr>
<td>16 GHz to 24 GHz</td>
<td>-39</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-33</td>
</tr>
<tr>
<td>Description</td>
<td>Ports 1, 2</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td></td>
<td>1 kHz Offset</td>
</tr>
<tr>
<td>10 MHz to 500 MHz</td>
<td>-85</td>
</tr>
<tr>
<td>500 MHz to 1 GHz</td>
<td>-105</td>
</tr>
<tr>
<td>1 GHz to 2 GHz</td>
<td>-100</td>
</tr>
<tr>
<td>2 GHz to 4 GHz</td>
<td>-95</td>
</tr>
<tr>
<td>4 GHz to 8 GHz</td>
<td>-89</td>
</tr>
<tr>
<td>8 GHz to 16 GHz</td>
<td>-83</td>
</tr>
<tr>
<td>16 GHz to 26.5 GHz</td>
<td>-78</td>
</tr>
</tbody>
</table>
### Test Port Input

Ports 1, 2, 3, 4. All Options. If Option 029 is included, see section Specifications: N5242A Option 029.

#### Table 16. Test Port Noise Floor\(^1\) (dBm)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 Hz IFBW</td>
<td>-80</td>
<td>-87</td>
</tr>
<tr>
<td>10 M Hz to 50 M Hz(^2)</td>
<td>-90</td>
<td>-95</td>
</tr>
<tr>
<td>50 M Hz to 100 M Hz(^2)</td>
<td>-104</td>
<td>-110</td>
</tr>
<tr>
<td>100 M Hz to 500 M Hz(^2)</td>
<td>-114</td>
<td>-117</td>
</tr>
<tr>
<td>500 M Hz to 2 GHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 GHz to 20 GHz</td>
<td>-114</td>
<td>-117</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-110</td>
<td>-115</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-107</td>
<td>-113</td>
</tr>
</tbody>
</table>

\(^1\) Total average (rms) noise power calculated as the mean value of a linear magnitude trace expressed in dBm.

\(^2\) May typically be degraded at particular frequencies below 500 M Hz due to spurious receiver residuals.

#### Table 17. Direct Receiver Access Input Noise Floor\(^1\) (dBm)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 M Hz to 50 M Hz(^2)</td>
<td>--</td>
<td>-130</td>
</tr>
<tr>
<td>50 M Hz to 100 M Hz(^2)</td>
<td>--</td>
<td>-128</td>
</tr>
<tr>
<td>100 M Hz to 500 M Hz(^2)</td>
<td>--</td>
<td>-132</td>
</tr>
<tr>
<td>500 M Hz to 2 GHz</td>
<td>--</td>
<td>-133</td>
</tr>
<tr>
<td>2 GHz to 20 GHz</td>
<td>--</td>
<td>-129</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>-122</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>-119</td>
</tr>
</tbody>
</table>

\(^1\) Total average (rms) noise power calculated as the mean value of a linear magnitude trace expressed in dBm.

\(^2\) May typically be degraded at particular frequencies below 500 M Hz due to spurious receiver residuals.
Table 18. Test Port Compression at 0.1 dB (dBm)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>50 M Hz to 500 M Hz</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>500 M Hz to 3.2 GHz</td>
<td>--</td>
<td>13</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>--</td>
<td>13</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>--</td>
<td>13</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>--</td>
<td>12</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>10.5</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>10</td>
</tr>
</tbody>
</table>

Test port receiver compression at specified input levels below 500 MHz is negligible due to coupler roll off in this frequency range.

Table 19. Test Port Compression @ 8 dBm Test Port Power (dB)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz(^1)</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>50 M Hz to 500 M Hz(^1)</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>500 M Hz to 3.2 GHz</td>
<td>&lt;0.17</td>
<td>--</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>&lt;0.17</td>
<td>--</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>&lt;0.17</td>
<td>--</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>&lt;0.23</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>&lt;0.23</td>
<td>--</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>&lt;0.29</td>
<td>--</td>
</tr>
</tbody>
</table>

\(^1\) Test port receiver compression at specified input levels below 500 MHz is negligible due to coupler roll off in this frequency range.
### Table 20. Trace Noise Magnitude (dB rms)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ratioed measurement, nominal power at test port.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### 1 kHz IFBW

<table>
<thead>
<tr>
<th>Range</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 100 MHz</td>
<td>0.007</td>
<td>0.0039</td>
</tr>
<tr>
<td>100 MHz to 13.5 GHz</td>
<td>0.002</td>
<td>0.0005</td>
</tr>
<tr>
<td>13.5 GHz to 16 GHz</td>
<td>0.002</td>
<td>0.0005</td>
</tr>
<tr>
<td>16 GHz to 22.5 GHz</td>
<td>0.002</td>
<td>0.0006</td>
</tr>
<tr>
<td>22.5 GHz to 24 GHz</td>
<td>0.003</td>
<td>0.0014</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>0.005</td>
<td>0.0020</td>
</tr>
</tbody>
</table>

#### 100 kHz IFBW

<table>
<thead>
<tr>
<th>Range</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 100 MHz</td>
<td>--</td>
<td>0.040</td>
</tr>
<tr>
<td>100 MHz to 13.5 GHz</td>
<td>--</td>
<td>0.005</td>
</tr>
<tr>
<td>13.5 GHz to 16 GHz</td>
<td>--</td>
<td>0.005</td>
</tr>
<tr>
<td>16 GHz to 22.5 GHz</td>
<td>--</td>
<td>0.005</td>
</tr>
<tr>
<td>22.5 GHz to 24 GHz</td>
<td>--</td>
<td>0.008</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>0.008</td>
</tr>
</tbody>
</table>

#### 600 kHz IFBW

<table>
<thead>
<tr>
<th>Range</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 100 MHz</td>
<td>--</td>
<td>0.140</td>
</tr>
<tr>
<td>100 MHz to 13.5 GHz</td>
<td>--</td>
<td>0.011</td>
</tr>
<tr>
<td>13.5 GHz to 16 GHz</td>
<td>--</td>
<td>0.011</td>
</tr>
<tr>
<td>16 GHz to 22.5 GHz</td>
<td>--</td>
<td>0.012</td>
</tr>
<tr>
<td>22.5 GHz to 24 GHz</td>
<td>--</td>
<td>0.020</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>0.020</td>
</tr>
</tbody>
</table>
### Table 21. Trace Noise Phase (deg rms)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ratioed measurement, nominal power at test port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>1 kHz IFBW</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 100 MHz</td>
<td>0.051</td>
<td>0.0261</td>
</tr>
<tr>
<td>100 MHz to 13.5 GHz</td>
<td>0.015</td>
<td>0.0041</td>
</tr>
<tr>
<td>13.5 GHz to 16 GHz</td>
<td>0.042</td>
<td>0.0124</td>
</tr>
<tr>
<td>16 GHz to 22.5 GHz</td>
<td>0.042</td>
<td>0.0135</td>
</tr>
<tr>
<td>22.5 GHz to 26.5 GHz</td>
<td>0.054</td>
<td>0.0225</td>
</tr>
<tr>
<td><strong>100 kHz IFBW</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 100 MHz</td>
<td>--</td>
<td>0.266</td>
</tr>
<tr>
<td>100 MHz to 13.5 GHz</td>
<td>--</td>
<td>0.030</td>
</tr>
<tr>
<td>13.5 GHz to 16 GHz</td>
<td>--</td>
<td>0.030</td>
</tr>
<tr>
<td>16 GHz to 22.5 GHz</td>
<td>--</td>
<td>0.033</td>
</tr>
<tr>
<td>22.5 GHz to 26.5 GHz</td>
<td>--</td>
<td>0.057</td>
</tr>
<tr>
<td><strong>600 kHz IFBW</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 100 MHz</td>
<td>--</td>
<td>1.053</td>
</tr>
<tr>
<td>100 MHz to 13.5 GHz</td>
<td>--</td>
<td>0.075</td>
</tr>
<tr>
<td>13.5 GHz to 16 GHz</td>
<td>--</td>
<td>0.075</td>
</tr>
<tr>
<td>16 GHz to 22.5 GHz</td>
<td>--</td>
<td>0.082</td>
</tr>
<tr>
<td>22.5 GHz to 26.5 GHz</td>
<td>--</td>
<td>0.139</td>
</tr>
</tbody>
</table>

### Table 22. Reference Level Magnitude

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>+/- 500 dB</td>
<td>--</td>
</tr>
<tr>
<td>Resolution</td>
<td>0.001 dB</td>
<td>--</td>
</tr>
</tbody>
</table>
### Table 23. Reference Level Phase

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>+/- 500°</td>
<td>--</td>
</tr>
<tr>
<td>Resolution</td>
<td>0.01°</td>
<td>--</td>
</tr>
</tbody>
</table>

### Table 24. Stability Magnitude (dB/ °C)

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stability is defined as a ratio measurement made at the test port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>0.01</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>--</td>
<td>0.01</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>--</td>
<td>0.01</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>--</td>
<td>0.02</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>--</td>
<td>0.02</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>--</td>
<td>0.03</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>0.03</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>0.04</td>
</tr>
</tbody>
</table>

### Table 25. Stability Phase (°/ °C)

- Stability is defined as a ratio measurement made at the test port.
Stability is defined as a ratio measurement made at the test port.

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>0.29</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>--</td>
<td>0.06</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>--</td>
<td>0.07</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>--</td>
<td>0.13</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>--</td>
<td>0.13</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>--</td>
<td>0.40</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>0.54</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>0.56</td>
</tr>
</tbody>
</table>
Table 26. Damage Input Level

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Port 1 or 2 or 3 or 4</td>
<td>&gt; +30 dBm RF, 40 VDC</td>
<td>Test Port 1 or 2 or 3 or 4</td>
</tr>
<tr>
<td>(Option 224 only) Source 2 Out 1 or</td>
<td>&gt; +30 dBm RF, 0 VDC</td>
<td>(Option 224 only) Source 2 Out 1</td>
</tr>
<tr>
<td>Source 2 Out 2</td>
<td></td>
<td>or Source 2 Out 2</td>
</tr>
</tbody>
</table>
Dynamic Accuracy

Table 27. Dynamic Accuracy (Specification)

Accuracy of the test port input power reading relative to the reference input power level.

Dynamic Accuracy, 0.010 GHz

Magnitude

Phase
Dynamic Accuracy, 0.050 GHz

Phase

Magnitude
Dynamic Accuracy, 0.500 GHz

Phase

Magnitude
Dynamic Accuracy, 16 - 20 GHz
Dynamic Accuracy, 20 - 26.5 GHz

**Phase**

![Phase Graph](image1)

**Magnitude**

![Magnitude Graph](image2)
Dynamic accuracy is verified with the following measurements:

Compression over frequency

IF linearity at a single frequency of 1.195 GHz using a reference level of -20 dBm for an input power range of 0 to -120 dBm.
Table 28. Test Port Input (Group Delay)\(^a\)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aperture (selectable)</td>
<td>((\text{frequency span}) / (\text{number of points -1}))</td>
</tr>
<tr>
<td>Maximum Aperture</td>
<td>20% of frequency span</td>
</tr>
<tr>
<td>Range</td>
<td>(0.5 \times (1/\text{minimum aperture}))</td>
</tr>
<tr>
<td>Maximum Delay</td>
<td>Limited to measuring no more than 180° of phase change</td>
</tr>
<tr>
<td>Accuracy</td>
<td>See graph below. Char.</td>
</tr>
</tbody>
</table>

The following graph shows characteristic group delay accuracy with full 2-port calibration and a 10 Hz IF bandwidth. Insertion loss is:

For any \(S_{ij}\) Group Delay measurement, \(S_{ii} = 0, S_{ij} = 1, S_{ji} = 0, S_{kl} = 0\) for all \(kl\) i j

**Group Delay (Typical)**

![Graph of group delay accuracy](image)

In general, the following formula can be used to determine the accuracy, in seconds, of specific group delay measurement:

\[
\pm \text{Phase Accuracy (deg)} / [360 \times \text{Aperture (Hz)}]
\]

Depending on the aperture and device length, the phase accuracy used is either incremental phase accuracy or worst-case phase accuracy.

\(^a\) Group delay is computed by measuring the phase change within a specified frequency step (determined by the frequency span and th
General Information

Miscellaneous Information
Front Panel
Rear Panel
Environment and Dimensions

Table 29. Miscellaneous Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Supplemental Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>System IF Bandwidth Range</td>
<td>1 Hz to 5 MHz, nominal</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel® 1.6 GHz Pentium® M</td>
</tr>
</tbody>
</table>

Table 30. Front Panel Information

All Options

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RF Connectors</strong></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Option 200 or 219 or 224 or 400 or 419 or 423: 3.5 mm (male), 50 ohm, (nominal)</td>
</tr>
<tr>
<td>Center Pin Recession</td>
<td>0.002 in. (characteristic)</td>
</tr>
<tr>
<td><strong>USB 2.0 Ports</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Master (4 ports)</strong></td>
<td></td>
</tr>
<tr>
<td>Standard</td>
<td>Compatible with USB 2.0</td>
</tr>
<tr>
<td>Connector</td>
<td>USB Type-A female</td>
</tr>
<tr>
<td><strong>Display</strong></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>26.3 cm (10.4 in) diagonal color active matrix LCD; 1024 (horizontal) X 768 (vertical) resolution</td>
</tr>
<tr>
<td>Refresh Rate</td>
<td>Vertical 60 Hz; Horizontal 46.08 kHz</td>
</tr>
<tr>
<td>Pixels</td>
<td>A display is considered faulty if:</td>
</tr>
<tr>
<td></td>
<td>More than 0.002% of the total pixels have a constant blue, green, red, or black appearance that will not change.</td>
</tr>
<tr>
<td></td>
<td>Three or more consecutive pixels have a constant blue, green, red, or black appearance that will not change.</td>
</tr>
</tbody>
</table>
appearance that will not change.
### Table 30. (Continued) Front Panel Information

<table>
<thead>
<tr>
<th>Display Range</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Magnitude</strong></td>
<td>+/-2500 dB (at 500 dB/ div), max</td>
</tr>
<tr>
<td><strong>Phase</strong></td>
<td>+/-2500° (at 500 dB/ div), max</td>
</tr>
<tr>
<td><strong>Polar</strong></td>
<td>10 pUnits, min</td>
</tr>
<tr>
<td></td>
<td>10,000 Units, max</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Display Resolution</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Magnitude</strong></td>
<td>0.001 dB/ div, min</td>
</tr>
<tr>
<td><strong>Phase</strong></td>
<td>0.01°/ div, min</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Marker Resolution</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Magnitude</strong></td>
<td>0.001 dB, min</td>
</tr>
<tr>
<td><strong>Phase</strong></td>
<td>0.01°, min</td>
</tr>
<tr>
<td><strong>Polar</strong></td>
<td>10 pUnit, min</td>
</tr>
</tbody>
</table>
### Table 31. Rear Panel Information

#### All Options

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>10 MHz Reference In</strong></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Input Frequency</td>
<td>10 MHz ± 10 ppm, typical</td>
</tr>
<tr>
<td>Input Level</td>
<td>-15 dBm to +20 dBm, typical</td>
</tr>
<tr>
<td>Input Impedance</td>
<td>200 Ω, nom.</td>
</tr>
<tr>
<td><strong>10 MHz Reference Out</strong></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>BNC, female</td>
</tr>
<tr>
<td>Output Frequency</td>
<td>10 MHz ± 1 ppm, typical</td>
</tr>
<tr>
<td>Signal Type</td>
<td>Sine Wave, typical</td>
</tr>
<tr>
<td>Output Level</td>
<td>+10 dBm ± 4 dB into 50</td>
</tr>
<tr>
<td>Output Impedance</td>
<td>50 Ω, nominal</td>
</tr>
<tr>
<td>Harmonics</td>
<td>&lt;-40 dBc, typical</td>
</tr>
<tr>
<td><strong>External IF Inputs</strong></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Allows use of external IF signals from remote mixers, bypassing the PNA's first converters</td>
</tr>
<tr>
<td>Connectors</td>
<td>SMA (female); A, B, C, D, R (4-port); A, B, R1, R2 (2-port)</td>
</tr>
<tr>
<td>Input Frequency</td>
<td></td>
</tr>
<tr>
<td>Normal IF path:</td>
<td></td>
</tr>
<tr>
<td>RF &lt; 53 MHz: IF = 2.535211 MHz</td>
<td></td>
</tr>
<tr>
<td>RF &gt;= 53 MHz: IF = 7.605634 MHz</td>
<td></td>
</tr>
<tr>
<td>Narrowband IF path:</td>
<td>IF = 10.70 MHz</td>
</tr>
<tr>
<td>Input Impedance</td>
<td>50</td>
</tr>
<tr>
<td>RF Damage Level</td>
<td></td>
</tr>
<tr>
<td>DC Damage Level</td>
<td>5.5 VDC</td>
</tr>
<tr>
<td>0.1 dB Compression Point</td>
<td></td>
</tr>
</tbody>
</table>
Table 31. (Continued) R ear P anel I nformation

<table>
<thead>
<tr>
<th>Pulse Inputs (IF Gates)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Function</strong></td>
<td>Internal receiver gates used for point-in-pulse and pulse-profile measurements</td>
</tr>
<tr>
<td><strong>Connectors</strong></td>
<td>15-pin mini D-sub</td>
</tr>
<tr>
<td><strong>Input Impedance</strong></td>
<td>1 K Ohm</td>
</tr>
<tr>
<td><strong>Minimum Pulse Width, Source Modulators</strong></td>
<td>33 ns</td>
</tr>
<tr>
<td><strong>Minimum Pulse Width, Receiver Gates</strong></td>
<td>20 ns</td>
</tr>
<tr>
<td><strong>DC Damage Level</strong></td>
<td>5.5 VDC</td>
</tr>
<tr>
<td><strong>Drive Voltage</strong></td>
<td>0 V (off), +3.3 V (on), nominal</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RF Pulse Modulator Input (Source Modulator)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>On/Off Ratio</strong></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 3.2 GHz</td>
<td>-64</td>
</tr>
<tr>
<td>3.2 GHz to 26.5 GHz</td>
<td>-80</td>
</tr>
<tr>
<td><strong>Pulse Period</strong></td>
<td></td>
</tr>
<tr>
<td>Minimum</td>
<td>33 ns</td>
</tr>
<tr>
<td>Maximum</td>
<td>70 s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>External Test Set Driver</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Function</strong></td>
<td>Used for driving remote mixers</td>
</tr>
<tr>
<td><strong>Connections</strong></td>
<td>SMA (female)</td>
</tr>
<tr>
<td><strong>RF, LO Output Frequency Range</strong></td>
<td>1.7 to 26.5 GHz</td>
</tr>
<tr>
<td>Description</td>
<td>Typical (dBm)</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td></td>
<td>Upper Limit (dBm)</td>
</tr>
<tr>
<td><strong>Test Set Drivers (Continued)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Rear Panel LO Power</strong></td>
<td></td>
</tr>
<tr>
<td>Full frequency range: 12.535 MHz to 26.5 GHz</td>
<td></td>
</tr>
<tr>
<td>1.7 GHz to 18 GHz</td>
<td>0</td>
</tr>
<tr>
<td>18 GHz to 22.5 GHz</td>
<td>2</td>
</tr>
<tr>
<td>22.5 GHz to 26.5 GHz</td>
<td>6</td>
</tr>
<tr>
<td><strong>Rear Panel RF Power</strong></td>
<td></td>
</tr>
<tr>
<td>3.2 GHz to 20 GHz</td>
<td>-3</td>
</tr>
<tr>
<td>20 GHz to 26.5 GHz</td>
<td>-8</td>
</tr>
</tbody>
</table>
Table 31. (Continued) Rear Panel Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VGA Video Output</strong></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>15-pin mini D-Sub; Drives VGA compatible monitors</td>
</tr>
<tr>
<td><strong>Devices Supported:</strong></td>
<td></td>
</tr>
<tr>
<td>Resolutions:</td>
<td></td>
</tr>
<tr>
<td>Flat Panel (TFT)</td>
<td>1024 X 768, 800 X 600, 640 X 480</td>
</tr>
<tr>
<td>Flat Panel (DSTN)</td>
<td>800 X 600, 640 X 480</td>
</tr>
<tr>
<td>CRT Monitor</td>
<td>1280 X 1024, 1024 X 768, 800 X 600, 640 X 480</td>
</tr>
<tr>
<td>Simultaneous operation of the internal and external displays is allowed, but with 640 X 480 resolution only. If you change resolution, you can only view the external display (internal display will &quot;white out&quot;).</td>
<td></td>
</tr>
<tr>
<td><strong>Bias Tee Inputs</strong></td>
<td></td>
</tr>
<tr>
<td>Connectors</td>
<td>BNC(f) for ports 1, 2, 3 and 4</td>
</tr>
<tr>
<td>Fuse</td>
<td>500 mA</td>
</tr>
<tr>
<td>Maximum Bias Current</td>
<td>+/-200 mA</td>
</tr>
<tr>
<td>Maximum Bias Voltage</td>
<td>+/-40 VDC</td>
</tr>
<tr>
<td>Trigger Inputs/ Outputs</td>
<td>BNC(f), TTL/ CMOS compatible</td>
</tr>
<tr>
<td>Test Set IO</td>
<td>25-pin D-Sub connector, available for external test set control.</td>
</tr>
<tr>
<td>Power IO</td>
<td>9-pin D-Sub, female; analog and digital IO</td>
</tr>
<tr>
<td>Handler IO</td>
<td>36-pin parallel I/O port; all input/output signals are default set to neg</td>
</tr>
<tr>
<td><strong>Table 31. (Continued) Rear Panel Information</strong></td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>GPIB</strong> (two ports - dedicated controller and dedicated talker/listener)</td>
<td>24-pin D-sub (Type D-24), female; compatible with IEEE-488.</td>
</tr>
<tr>
<td><strong>Parallel Port (LPT1)</strong></td>
<td>25-pin D-Sub miniature connector, female; provides connection to printers or any other parallel port peripherals</td>
</tr>
<tr>
<td><strong>Serial Port (COM 1)</strong></td>
<td>9-pin D-Sub, male; compatible with RS-232</td>
</tr>
<tr>
<td><strong>USB Port</strong></td>
<td>Four ports on front panel (all Host) and five ports (four Host and one Device) on rear panel. Type A configuration (eight Host) and Type B configuration (one Device), USB 2.0 compatible.</td>
</tr>
<tr>
<td><strong>LAN</strong></td>
<td>10/100BaseT Ethernet, 8-pin configuration; auto selects between the two data rates</td>
</tr>
</tbody>
</table>

**Line Power**

<table>
<thead>
<tr>
<th><strong>Frequency, Voltage</strong></th>
<th>50/60 Hz for 100 240 VAC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Power supply is auto switching</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Max</strong></td>
<td>450 watts</td>
</tr>
</tbody>
</table>

**Note**: Option H11 is not available with the N5242A network analyzer.
## Table 32. Analyzer Dimensions and Weight

<table>
<thead>
<tr>
<th>Cabinet Dimensions</th>
<th>Height</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excluding front and rear panel hardware and feet</td>
<td>266 mm</td>
<td>4x:</td>
</tr>
<tr>
<td></td>
<td>10.5 in</td>
<td>17x:</td>
</tr>
<tr>
<td>Excluding front and rear panel hardware and feet.</td>
<td>266 mm</td>
<td>48:</td>
</tr>
<tr>
<td>Including rack-mount flanges.</td>
<td>10.5 in</td>
<td>15:</td>
</tr>
<tr>
<td></td>
<td>EIA RU&lt;sup&gt;1&lt;/sup&gt; = 6</td>
<td></td>
</tr>
<tr>
<td>As shipped - including front panel connectors, rear</td>
<td>277 mm</td>
<td>4:</td>
</tr>
<tr>
<td>panel bumpers, and feet.</td>
<td>10.9 in</td>
<td>17:</td>
</tr>
<tr>
<td>As shipped including rack-mount flanges</td>
<td>277 mm</td>
<td>46:</td>
</tr>
<tr>
<td></td>
<td>10.9 in</td>
<td>15:</td>
</tr>
</tbody>
</table>

### Weight

<table>
<thead>
<tr>
<th></th>
<th>Option 200 or 219 or 224</th>
<th>Option 400 or 419 or 423</th>
</tr>
</thead>
<tbody>
<tr>
<td>Net</td>
<td>27 kg (60 lb), nominal</td>
<td>37 kg (82 lb), nominal</td>
</tr>
<tr>
<td>Shipping</td>
<td>43 kg (95 lb), nominal</td>
<td>53 kg (117 lb), nominal</td>
</tr>
</tbody>
</table>

<sup>1</sup> Network analyzer feet removed.

**Note:** For Regulatory and Environmental information, refer to the PNA Series Installation and Quick Start Guide, located online at [http/](http/).
**Measurement Throughput Summary**

Typical Cycle Time for Measurement Completion

Cycle Time vs. IF Bandwidth

Cycle Time vs. Number of Points

Data Transfer Time

**Table 33. Typical Cycle Time\(^a\) (ms) for Measurement Completion**

<table>
<thead>
<tr>
<th>All Options</th>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of Points</td>
<td></td>
</tr>
<tr>
<td></td>
<td>201</td>
<td>401</td>
</tr>
<tr>
<td>Start 9 GHz, Stop 10 GHz, 600 kHz IF bandwidth</td>
<td>Uncorrected</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>2-Port cal</td>
<td>20</td>
</tr>
<tr>
<td>Start 9 GHz, Stop 10 GHz, 10 kHz IF bandwidth</td>
<td>Uncorrected</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>2-Port cal</td>
<td>80</td>
</tr>
<tr>
<td>Start 9 GHz, Stop 10 GHz, 1 kHz IF bandwidth</td>
<td>Uncorrected</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td>2-Port cal</td>
<td>460</td>
</tr>
<tr>
<td>Start 10 GHz, Stop 20 GHz, 600 kHz IF bandwidth</td>
<td>Uncorrected</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>2-Port cal</td>
<td>62</td>
</tr>
<tr>
<td>Start 10 GHz, Stop 20 GHz, 10 kHz IF bandwidth</td>
<td>Uncorrected</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>2-Port cal</td>
<td>149</td>
</tr>
<tr>
<td>Start 10 GHz, Stop 20 GHz, 1 kHz IF bandwidth</td>
<td>Uncorrected</td>
<td>236</td>
</tr>
<tr>
<td>2-Port Cal</td>
<td>400</td>
<td>926</td>
</tr>
</tbody>
</table>
Table 33. (Continued) Typical Cycle Time (ms) for Measurement Completion

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Start 10 MHz, Stop 26.5 GHz, 600 kHz IF bandwidth</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Uncorrected</td>
<td>59</td>
</tr>
<tr>
<td>2-Port cal</td>
<td>125</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start 10 MHz, Stop 26.5 GHz, 10 kHz IF bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncorrected</td>
</tr>
<tr>
<td>2-Port cal</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Start 10 MHz, Stop 26.5 GHz, 1 kHz IF bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncorrected</td>
</tr>
<tr>
<td>2-Port cal</td>
</tr>
</tbody>
</table>

\(^a\) Includes sweep time, retrace time and band-crossing time. Analyzer display turned off with DISPLAY:ENABLE OFF. Add 21 ms for display on. Data for one trace (S\(_{11}\)) measurement.

Note: Option H08 and Option H11 are not available with the N5242A network analyzer.
Table 34. Cycle Time vs. IF Bandwidth

Applies to the Preset condition (201 points, correction off) except for the following changes:

- CF = 10 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)

<table>
<thead>
<tr>
<th>IF Bandwidth (Hz)</th>
<th>Cycle Time (ms)</th>
<th>Trace Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>600,000</td>
<td>5.00</td>
<td>0.009</td>
</tr>
<tr>
<td>100,000</td>
<td>6.84</td>
<td>0.003</td>
</tr>
<tr>
<td>30,000</td>
<td>11.6</td>
<td>0.002</td>
</tr>
<tr>
<td>10,000</td>
<td>29.0</td>
<td>0.001</td>
</tr>
<tr>
<td>3,000</td>
<td>71.8</td>
<td>0.0007</td>
</tr>
<tr>
<td>1,000</td>
<td>222</td>
<td>0.0004</td>
</tr>
<tr>
<td>300</td>
<td>640</td>
<td>0.0003</td>
</tr>
<tr>
<td>100</td>
<td>1826</td>
<td>0.0002</td>
</tr>
<tr>
<td>30</td>
<td>5982</td>
<td>&lt;0.0002</td>
</tr>
<tr>
<td>10</td>
<td>17830</td>
<td>&lt;0.0002</td>
</tr>
<tr>
<td>3</td>
<td>60000</td>
<td>&lt;0.0002</td>
</tr>
</tbody>
</table>

a Cycle time includes sweep and retrace time.
Table 35. Cycle Time vs. Number of Points

Applies to the Preset condition (correction off) except for the following changes:

- CF = 10 GHz
- Span = 100 MHz
- Display off (add 21 ms for display on)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF Bandwidth (Hz)</td>
<td>Number of Points</td>
</tr>
<tr>
<td>1,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>401</td>
</tr>
<tr>
<td></td>
<td>801</td>
</tr>
<tr>
<td></td>
<td>1,601</td>
</tr>
<tr>
<td></td>
<td>6,401</td>
</tr>
<tr>
<td></td>
<td>16,001</td>
</tr>
<tr>
<td>10,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>401</td>
</tr>
<tr>
<td></td>
<td>801</td>
</tr>
<tr>
<td></td>
<td>1,601</td>
</tr>
<tr>
<td></td>
<td>6,401</td>
</tr>
<tr>
<td></td>
<td>16,001</td>
</tr>
</tbody>
</table>
Table 35. (Continued) Cycle Time V s. Number of Points

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF Bandwidth (Hz)</td>
<td>Number of Points</td>
</tr>
<tr>
<td>30,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>401</td>
</tr>
<tr>
<td></td>
<td>801</td>
</tr>
<tr>
<td></td>
<td>1,601</td>
</tr>
<tr>
<td></td>
<td>6,401</td>
</tr>
<tr>
<td></td>
<td>16,001</td>
</tr>
<tr>
<td>600,000</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>201</td>
</tr>
<tr>
<td></td>
<td>401</td>
</tr>
<tr>
<td></td>
<td>801</td>
</tr>
<tr>
<td></td>
<td>1,601</td>
</tr>
<tr>
<td></td>
<td>6,401</td>
</tr>
<tr>
<td></td>
<td>16,001</td>
</tr>
</tbody>
</table>

¹ Cycle time includes sweep and retrace time.
Table 36. Data Transfer Time (ms)

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Performance</th>
<th>Number of Points</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Number of Points</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>201</td>
</tr>
<tr>
<td></td>
<td></td>
<td>401</td>
</tr>
<tr>
<td><strong>SCPI over GPIB</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Program executed on external PC(^2))</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32-bit floating point</td>
<td>5.6</td>
<td>10.5</td>
</tr>
<tr>
<td>64-bit floating point</td>
<td>10.5</td>
<td>20.3</td>
</tr>
<tr>
<td>ASCII</td>
<td>46</td>
<td>92.5</td>
</tr>
<tr>
<td><strong>SCPI over SICL/ LAN or TCP/IP Socket</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Program executed in the analyzer)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32-bit floating point</td>
<td>0.18</td>
<td>0.21</td>
</tr>
<tr>
<td>64-bit floating point</td>
<td>0.22</td>
<td>0.28</td>
</tr>
<tr>
<td>ASCII</td>
<td>6.3</td>
<td>12.3</td>
</tr>
<tr>
<td><strong>COM(^3)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Program executed in the analyzer)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32-bit floating point</td>
<td>&lt;0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>Variant type</td>
<td>0.75</td>
<td>1.2</td>
</tr>
<tr>
<td><strong>DCOM over LAN(^3)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Program executed on external PC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32-bit floating point</td>
<td>&lt;1.0</td>
<td>1.2</td>
</tr>
<tr>
<td>Variant type</td>
<td>2.7</td>
<td>4.5</td>
</tr>
</tbody>
</table>

1 Measured with the analyzer display off. Values will increase slightly if the analyzer display is on.
2 Measured when using the SCPI command DISPlay: VISible OFF.
3 Values are for real and imaginary pairs, with the analyzer display off.

**Note**: Specifications for Recall & Sweep Speed are not provided for the N5242A analyzers.
Specifications: Front-Panel Jumpers

Model N5242A (PNA-X)

Note: All PNA-X options have the following front-panel jumpers for each port.

- Measurement Receiver Inputs
- Reference Receiver Inputs
- Reference Outputs (Source Out)
- Source Outputs
- Coupler Inputs
Table 37. Measurement Receiver Inputs
(Rcvr A IN, Rcvr B IN, Rcvr C IN, Rcvr D IN) @ 0.1dB Typical Compression

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>All Options</td>
</tr>
<tr>
<td><strong>Maximum Input Level</strong></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz$^1$</td>
<td>--</td>
</tr>
<tr>
<td>50 MHz to 500 MHz$^1$</td>
<td>--</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-2 dBm</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-2 dBm</td>
</tr>
<tr>
<td>10 GHZ to 16 GHz</td>
<td>-2 dBm</td>
</tr>
<tr>
<td>16 GHZ to 20 GHz</td>
<td>-2.5 dBm</td>
</tr>
<tr>
<td>20 GHZ to 24 GHz</td>
<td>-4 dBm</td>
</tr>
<tr>
<td>24 GHZ to 26.5 GHz</td>
<td>-4 dBm</td>
</tr>
<tr>
<td><strong>Damage Level</strong></td>
<td></td>
</tr>
<tr>
<td>N5242A</td>
<td>+15 dBm</td>
</tr>
<tr>
<td><strong>Maximum DC Level</strong></td>
<td></td>
</tr>
<tr>
<td>N5242A</td>
<td>0 V</td>
</tr>
</tbody>
</table>

$^1$ Test port receiver compression at specified input levels below 500 MHz is negligible due to coupler roll off in this frequency range.
Table 38. Reference Receiver Input
(RCVR R1 IN) @ Max Specified Output Power

<table>
<thead>
<tr>
<th>Description</th>
<th>Option 200 or 400 Filtered Mode</th>
<th>Option 200 or 400 Hi Pwr Mode</th>
<th>Option 224 or 423 Filtered Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Input Level</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-8 dBm</td>
<td>-3 dBm</td>
<td>-9 dBm</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>-6 dBm</td>
<td>-3 dBm</td>
<td>-7 dBm</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-6 dBm</td>
<td>-6 dBm</td>
<td>-7 dBm</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-3 dBm</td>
<td>-3 dBm</td>
<td>-2 dBm</td>
</tr>
<tr>
<td>10 GHZ to 16 GHZ</td>
<td>-4 dBm</td>
<td>-4 dBm</td>
<td>-3 dBm</td>
</tr>
<tr>
<td>16 GHZ to 20 GHZ</td>
<td>-5 dBm</td>
<td>-5 dBm</td>
<td>-6 dBm</td>
</tr>
<tr>
<td>20 GHZ to 24 GHZ</td>
<td>-7 dBm</td>
<td>-7 dBm</td>
<td>-9 dBm</td>
</tr>
<tr>
<td>24 GHZ to 26.5 GHZ</td>
<td>-16 dBm</td>
<td>-16 dBm</td>
<td>-18 dBm</td>
</tr>
<tr>
<td>Damage Level</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>+15 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maximum DC Level</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>+/-7 V</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 39. Reference Receiver Input

(RCVR R2 IN, RCVR R3 IN, RCVR R4 IN) @ Max Specified Output Power

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical Options</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Option 400</td>
</tr>
<tr>
<td></td>
<td>RCVR R3 IN</td>
</tr>
<tr>
<td></td>
<td>Filtered Mode</td>
</tr>
<tr>
<td>Maximum Input Level</td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-6 dBm</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>-4 dBm</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-4 dBm</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>0 dBm</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>1 dBm</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>1 dBm</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>0 dBm</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-8 dBm</td>
</tr>
</tbody>
</table>

Damage Level

+15 dBm

Maximum DC Level

+/- 15 V
Table 39. (Continued) Reference Receiver Input

(RCVR R2 IN, RCVR R3 IN, RCVR R4 IN) @ Max Specified Output Power

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Option 423</td>
</tr>
<tr>
<td></td>
<td>RCVR R3 IN</td>
</tr>
<tr>
<td></td>
<td>Filtered Mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Maximum Input Level</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-7 dBm</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>-6 dBm</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-5 dBm</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>1 dBm</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>1 dBm</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>0 dBm</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-2 dBm</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-10 dBm</td>
</tr>
</tbody>
</table>

| Damage Level              | +15 dBm          |

<p>| Maximum DC Level          | +/-15 V          |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Option 200 or 400</td>
</tr>
<tr>
<td></td>
<td>Filtered Mode</td>
</tr>
</tbody>
</table>

**Maximum Input Level**

<table>
<thead>
<tr>
<th>Range</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-8 dBm</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>-6 dBm</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-6 dBm</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-3 dBm</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>-4 dBm</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-5 dBm</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-7 dBm</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-16 dBm</td>
</tr>
</tbody>
</table>

**Damage Level**

+15 dBm

**Maximum DC Level**

+/-7 V
### Table 41. Reference Output

*(REF 2 SOURCE OUT, REF 3 SOURCE OUT, REF 4 SOURCE OUT) @ Max Specified Output Power*

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
<th>Option 400 REF 3 Source Out Filtered Mode</th>
<th>Option 400 REF 3 Source Out Hi Pwr Mode</th>
<th>Option 200 or 400 REF 2 Source Out REF 4 Source Out Filtered Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Maximum Input Level</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-6 dBm</td>
<td>-1 dBm</td>
<td>-1 dBm</td>
<td></td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>-4 dBm</td>
<td>-1 dBm</td>
<td>-1 dBm</td>
<td></td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-4 dBm</td>
<td>-4 dBm</td>
<td>0 dBm</td>
<td></td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>0 dBm</td>
<td>0 dBm</td>
<td>0 dBm</td>
<td></td>
</tr>
<tr>
<td>10 GHZ to 16 GHZ</td>
<td>1 dBm</td>
<td>1 dBm</td>
<td>0 dBm</td>
<td></td>
</tr>
<tr>
<td>16 GHZ to 20 GHz</td>
<td>1 dBm</td>
<td>1 dBm</td>
<td>-3 dBm</td>
<td></td>
</tr>
<tr>
<td>20 GHZ to 24 GHz</td>
<td>0 dBm</td>
<td>0 dBm</td>
<td>-6 dBm</td>
<td></td>
</tr>
<tr>
<td>24 GHZ to 26.5 GHz</td>
<td>-8 dBm</td>
<td>-8 dBm</td>
<td>-12 dBm</td>
<td></td>
</tr>
<tr>
<td><strong>Damage Level</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>+15 dBm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Maximum DC Level</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 V</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 41. (Continued) Reference Output

(REF 2 SOURCE OUT, REF 3 SOURCE OUT, REF 4 SOURCE OUT) @ Max Specified Output Power

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Option 423</td>
</tr>
<tr>
<td></td>
<td>REF 3</td>
</tr>
<tr>
<td></td>
<td>Source Out</td>
</tr>
<tr>
<td></td>
<td>Filtered Mode</td>
</tr>
</tbody>
</table>

Maximum Input Level

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-7 dBm</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>-6 dBm</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-5 dBm</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>1 dBm</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>1 dBm</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>0 dBm</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-2 dBm</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-10 dBm</td>
</tr>
</tbody>
</table>

Damage Level

<table>
<thead>
<tr>
<th></th>
<th>+15 dBm</th>
</tr>
</thead>
</table>

Maximum DC Level

<table>
<thead>
<tr>
<th></th>
<th>0 V</th>
</tr>
</thead>
</table>
Table 42. Source Outputs
(PORT 1 SOURCE OUT, PORT 2 SOURCE OUT, PORT 3 SOURCE OUT, PORT 4 SOURCE OUT) @ Max Specified Output Power

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
<th>Filtered Mode</th>
<th>Hi Pwr Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Option 200 or 400</td>
<td>Port 1 Source Out</td>
<td>Port 1 Source Out</td>
</tr>
<tr>
<td>Maximum Input Level</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 50 MHz</td>
<td>8 dBm</td>
<td>13 dBm</td>
<td>13 dBm</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>10 dBm</td>
<td>13 dBm</td>
<td>13 dBm</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>11 dBm</td>
<td>11 dBm</td>
<td>13 dBm</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>14 dBm</td>
<td>14 dBm</td>
<td>14 dBm</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>14 dBm</td>
<td>14 dBm</td>
<td>14 dBm</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>14 dBm</td>
<td>14 dBm</td>
<td>11 dBm</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>13 dBm</td>
<td>13 dBm</td>
<td>9 dBm</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>7 dBm</td>
<td>7 dBm</td>
<td>4 dBm</td>
</tr>
<tr>
<td>Damage Level</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>+30 dBm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maximum DC Level</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 V</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 42. (Continued) Source Outputs
(PORT 1 SOURCE OUT, PORT 2 SOURCE OUT, PORT 3 SOURCE OUT, PORT 4 SOURCE OUT) @ Max Specified Output Power

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Option 224 or 423 Port 1 Source Out Port 3 Source Out Filtered Mode</td>
</tr>
</tbody>
</table>

#### Maximum Input Level

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Typical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>7 dBm</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>8 dBm</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>9 dBm</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>14 dBm</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>14 dBm</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>12 dBm</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>9 dBm</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>2 dBm</td>
</tr>
</tbody>
</table>

#### Damage Level

<table>
<thead>
<tr>
<th>Damage Level</th>
<th>Typical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>+30 dBm</td>
<td></td>
</tr>
</tbody>
</table>

#### Maximum DC Level

<table>
<thead>
<tr>
<th>DC Level</th>
<th>Typical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 V</td>
<td></td>
</tr>
</tbody>
</table>
Table 43. Coupler Inputs
(PORT 1 CPLR THRU, PORT 2 CPLR THRU, PORT 3 CPLR THRU, PORT 4 CPLR THRU) Insertion Loss of Coupler Thru

<table>
<thead>
<tr>
<th>Description</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Option 200 or 400</td>
</tr>
</tbody>
</table>

**Maximum Input Level**

<table>
<thead>
<tr>
<th>Frequency Range</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td></td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td></td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td></td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td></td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td></td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td></td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td></td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td></td>
</tr>
</tbody>
</table>

**Damage Level**

| N5242A | +30 dBm |

**Maximum DC Level**

| N5242A | +/- 40 V |
**Specifications: N5242A Option 029**

General specifications apply to Option 029 with the port 1 noise tuner switch set to the internal position except as stated below. By default, the switch is set to the external position to ensure functional compatibility with existing PNA instruments. Refer to the Noise Figure Application for information on setting the switch position.

**Corrected System Performance with 3.5mm Connectors**

Note: For any Sii reflection measurement:

- $S_{jj} = 0$.

For any Sij transmission measurement:

- $S_{ji} = S_{ij}$ when $S_{ij} \leq 1$
- $S_{ji} = 1/S_{ij}$ when $S_{ij} > 1$
- $S_{kk} = 0$ for all $k$

**Table 44. 85052B Calibration Kit**

**N5242A Option 029**

Applies to the N5242A analyzer with Option 029, an 85052B (3.5mm) calibration kit, an 85131F flexible test port cable set, and a full 2-port calibration. S-parameter measurements were made with the analyzer in the noise figure configuration. Environmental temperature 23° ±3 °C, with < 1 °C deviation from calibration temperature

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 MHz to 500 MHz</td>
</tr>
<tr>
<td><strong>Directivity</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>48</td>
</tr>
<tr>
<td><strong>Source Match</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>40</td>
</tr>
<tr>
<td><strong>Load Match</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>48</td>
</tr>
<tr>
<td><strong>Reflection Tracking</strong></td>
<td>±0.003</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
<tr>
<td><strong>S21 Transmission Tracking</strong></td>
<td>±0.033</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
<tr>
<td><strong>S12 Transmission Tracking</strong></td>
<td>±0.044</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
</tbody>
</table>

1 Temperature deviation is a characteristic value.
Specifications: N5242A Option 029 (continued)

S₂¹ Transmission Uncertainty (Specifications)

S₂¹ Magnitude

S₂¹ Phase
Specifications: N5242A Option 029 (continued)

S₁₂ Transmission Uncertainty (Specifications)

S₁₂ Magnitude

![Graph showing S₁₂ Magnitude uncertainty vs. Transmission Coefficient (dB)]

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz
- 20 GHz to 26.5 GHz

- S₁₁ = S₂₂ = 0
- Source Power = -11 dBm

S₁₂ Phase

![Graph showing S₁₂ Phase uncertainty vs. Transmission Coefficient (dB)]

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz
- 20 GHz to 26.5 GHz

- S₁₁ = S₂₂ = 0
- Source Power = -11 dBm
Specifications: N5242A Option 029 (continued)

Reflection Uncertainty (Specifications)

Magnitude

Phase

N5242A Option 029 with 85052B

Source Power = -5 dBm

S21 = S12 = 0
Specifications: N5242A Option 029 (continued)

Table 45. N4433A 4-Port Electronic Calibration Module

N5242A Option 029

Note: Uncertainty curves for the N4433A are created using a 2-port calibration. Multiport uncertainties are not supported at this time.

Applies to the N5242A analyzer with Option 029, an N4433A (3.5mm) electronic calibration module, an 85131F flexible test port cable set, and a full 2-port calibration. S-parameter measurements were made with the analyzer in the noise figure configuration. Environmental temperature 23° ±3 °C, with < 1 °C deviation from calibration temperature.

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 M Hz to 500 M Hz</td>
</tr>
<tr>
<td>Directivity</td>
<td>52</td>
</tr>
<tr>
<td>Source Match</td>
<td>42</td>
</tr>
<tr>
<td>Load Match</td>
<td>40</td>
</tr>
<tr>
<td>Reflection Tracking&lt;sup&gt;1&lt;/sup&gt;</td>
<td>±0.060</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
<tr>
<td>S&lt;sub&gt;21&lt;/sub&gt; Transmission Tracking&lt;sup&gt;1&lt;/sup&gt;</td>
<td>±0.065</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
<tr>
<td>S&lt;sub&gt;12&lt;/sub&gt; Transmission Tracking&lt;sup&gt;1&lt;/sup&gt;</td>
<td>±0.069</td>
</tr>
<tr>
<td></td>
<td>+0.010/ °C</td>
</tr>
</tbody>
</table>

<sup>1</sup> Temperature deviation is a characteristic value.
Specifications: N5242A Option 029 (continued)

S₂¹ Transmission Uncertainty (Specifications)

S₂¹ Magnitude

S₂¹ Phase
Specifications: N5242A Option 029 (continued)

S\(_{12}\) Transmission Uncertainty (Specifications)

**S\(_{12}\) Magnitude**

N5242A Option 029 Full Two Port Cal Using N4433A

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz

Uncertainty (dB)

Transmission Coefficient (dB)

S\(_{11}\) = S\(_{22}\) = 0
Source Power = -11 dBm

**S\(_{12}\) Phase**

N5242A Option 029 Full Two Port Cal Using N4433A

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz

Uncertainty (degrees)

Transmission Coefficient (dB)

S\(_{11}\) = S\(_{22}\) = 0
Source Power = -11 dBm
Specifications: N5242A Option 029 (continued)

Reflection Uncertainty (Specifications)

**Magnitude**

N5242A Option 029 with N4433A

0.05

0.04

0.03

0.02

0.01

0

0.0

0.2

0.4

0.6

0.8

1

Reflection Coefficient (linear)

Uncertainty (linear)

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz

S21 = S12 = 0
Source Power = -5 dBm

**Phase**

N5242A Option 029 with N4433A

10

8

6

4

2

0

0

0.2

0.4

0.6

0.8

1

Reflection Coefficient (linear)

Uncertainty (deg)

- 50 MHz to 500 MHz
- 500 MHz to 2 GHz
- 2 GHz to 20 GHz

S21 = S12 = 0
Source Power = -5 dBm
Specifications: N5242A Option 029 (continued)

Table 46. N4691B 2- Port Electronic Calibration Module

N5242A Option 029

Applies to the N5242A analyzer with Option 029, an N4691B (3.5mm) electronic calibration module, an 85131F flexit were made with the analyzer in the noise figure configuration. Environmental temperature 23° ±3 °C, with <1 °C de

<table>
<thead>
<tr>
<th>Description</th>
<th>Specification (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50 MHz to 500 MHz</td>
</tr>
<tr>
<td>Directivity</td>
<td>46</td>
</tr>
<tr>
<td>Source Match</td>
<td>41</td>
</tr>
<tr>
<td>Load Match</td>
<td>40</td>
</tr>
<tr>
<td>Reflection Tracking 1</td>
<td>±0.050 (+0.010/ °C)</td>
</tr>
<tr>
<td>S21 Transmission Tracking 1</td>
<td>±0.060 (+0.010/ °C)</td>
</tr>
<tr>
<td>S12 Transmission Tracking 1</td>
<td>±0.068 (+0.010/ °C)</td>
</tr>
</tbody>
</table>

1 Temperature deviation is a characteristic value.
Transmission Uncertainty (Specifications)

### S21 Magnitude

- **N5242A Option 029 Full Two Port Cal Using N4691B**

![Graph showing S21 magnitude uncertainty](image)

- **Uncertainty (dB)**
- **Transmission Coefficient (dB)**
- **50 MHz to 500 MHz**
- **500 MHz to 2 GHz**
- **2 GHz to 20 GHz**
- **20 GHz to 26.5 GHz**

### S21 Phase

- **N5242A Option 029 Full Two Port Cal**

![Graph showing S21 phase uncertainty](image)

- **Uncertainty (degrees)**
- **Transmission Coefficient**
- **50 MHz to 500 MHz**
- **500 MHz to 2 GHz**
- **2 GHz to 20 GHz**
- **20 GHz to 26.5 GHz**

*S11 = S22 = 0  
Source Power = -11 dBm*
Specifications: N5242A Option 029 (continued)

Transmission Uncertainty (Specifications)

S12 Magnitude

![Graph showing transmission uncertainty magnitude for different frequency ranges.]

S12 Phase

![Graph showing transmission uncertainty phase for different frequency ranges.]

Specifications: N5242A Option 029 (continued)

Reflection Uncertainty (Specifications)
This N5242A document does not present specifications for the 85052C or 85052D Calibration Kit. Please download our free Uncertainty Calculator from http://www.agilent.com/find/na_calculator to generate the data and curves for the 85052C or the 85052D Calibration Kit.
Uncorrected System Performance, N5242A Option 029

Noise Error Terms: N5242A Option 029

Table 47. Noise State Directivity, Port 1 and Port 2\(^1\) (dB)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 M Hz</td>
<td>-16</td>
<td>-22</td>
</tr>
<tr>
<td>50 M Hz to 500 M Hz</td>
<td>-24</td>
<td>-28</td>
</tr>
<tr>
<td>500 M Hz to 3.2 GHz</td>
<td>-24</td>
<td>-32</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-23</td>
<td>-25</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>-16</td>
<td>-20</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-15</td>
<td>-20</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-15</td>
<td>-20</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-15</td>
<td>-20</td>
</tr>
</tbody>
</table>

\(^1\) Analyzer test set switches are set in accordance with the Figure 22 block diagram, but with a jumper replacing the ECal module.

Table 48. Noise State Load Match, Port 1\(^1\) (dB)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 M Hz</td>
<td>-11</td>
<td>-18</td>
</tr>
<tr>
<td>50 M Hz to 500 M Hz</td>
<td>-17</td>
<td>-24</td>
</tr>
<tr>
<td>500 M Hz to 3.2 GHz</td>
<td>-15</td>
<td>-19</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-10</td>
<td>-15</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>-9</td>
<td>-15</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-8</td>
<td>-13</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-7</td>
<td>-13</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-7</td>
<td>-11</td>
</tr>
</tbody>
</table>

\(^1\) Analyzer test set switches are set in accordance with the Figure 22 block diagram, but with a jumper replacing the ECal module.
Specifications: N5242A Option 029 (continued)

Table 49. Noise State Load Match, Port 2\(^1\) (dB)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-9</td>
<td>-12</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>-13</td>
<td>-15</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-9</td>
<td>-12</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-5.5</td>
<td>-7.5</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>-5.5</td>
<td>-7.5</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-5.5</td>
<td>-7.5</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-5.5</td>
<td>-7.5</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-5.5</td>
<td>-7.5</td>
</tr>
</tbody>
</table>

\(^1\) Analyzer test set switches are set in accordance with the Figure 22 block diagram, but with a jumper replacing the ECal module.

Table 50. Noise State Reflection Tracking (\(S_{11}\) and \(S_{22}\)), Min/Max (dB)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>+/- 1.5</td>
</tr>
</tbody>
</table>
## Specifications: N5242A Option 029 (continued)

### Table 51. Noise State Transmission Tracking, Min/Max (dB)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>+/-1.5</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>--</td>
<td>+/-1.5</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>--</td>
<td>+/-1.5</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>--</td>
<td>+/-1.5</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>--</td>
<td>+/-1.5</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>--</td>
<td>+/-1.5</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>+/-1.5</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>+/-1.5</td>
</tr>
</tbody>
</table>

### Table 52. Noise State Source Match, Port 1\(^1\) (dB)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-9</td>
<td>-13</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>-18</td>
<td>-28</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-17</td>
<td>-22</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-12</td>
<td>-18</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>-11</td>
<td>-16</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-9</td>
<td>-13</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-8</td>
<td>-13</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-7</td>
<td>-12</td>
</tr>
</tbody>
</table>

\(^1\) Analyzer test set switches are set in accordance with the Figure 22 block diagram, but with a jumper replacing the ECal module.
## Specifications: N5242A Option 029 (continued)

### Table 53. Noise State Source Match, Port 2\(^1\) (dB)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>-9</td>
<td>-12</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>-13</td>
<td>-15</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>-9</td>
<td>-12</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>-6</td>
<td>-7</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>-6</td>
<td>-8</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>-7</td>
<td>-9</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>-6</td>
<td>-9</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-6</td>
<td>-9</td>
</tr>
</tbody>
</table>

\(^1\) Analyzer test set switches are set in accordance with the Figure 22 block diagram, but with a jumper replacing the ECal module.

### Test Port Input: Option 029

### Table 54. Receiver Noise Figure, Port 2 (dB)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>800 kHz, 2 MHz, 4 MHz, 8 MHz BW, High Gain Setting(^1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 200 MHz</td>
<td>9.0</td>
<td>--</td>
</tr>
<tr>
<td>200 MHz to 2 GHz</td>
<td>12.0</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 20 GHz</td>
<td>14.5</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 26.5 GHz</td>
<td>17.0</td>
<td>--</td>
</tr>
</tbody>
</table>

\(^1\) 24 MHz BW, High Gain Setting

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 MHz to 200 MHz</td>
<td>9.0</td>
<td>--</td>
</tr>
<tr>
<td>200 MHz to 2 GHz</td>
<td>12.0</td>
<td>--</td>
</tr>
<tr>
<td>2 GHz to 20 GHz</td>
<td>14.5</td>
<td>--</td>
</tr>
<tr>
<td>20 GHz to 26.5 GHz</td>
<td>18.5</td>
<td>--</td>
</tr>
</tbody>
</table>
1 Using Option 029 noise receivers.
## Specifications: N5242A Option 029 (continued)

### Table 55. Noise Jitter\(^1,2\) (dB)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>4 MHz BW, Low Gain Setting</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 15 MHz</td>
<td>0.30</td>
<td>0.15</td>
</tr>
<tr>
<td>15 MHz to 3 GHz</td>
<td>0.10</td>
<td>0.07</td>
</tr>
<tr>
<td>3 GHz to 26.5 GHz</td>
<td>0.11</td>
<td>0.07</td>
</tr>
<tr>
<td><strong>4 MHz BW, Medium Gain Setting</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 15 MHz</td>
<td>0.30</td>
<td>0.15</td>
</tr>
<tr>
<td>15 MHz to 3 GHz</td>
<td>0.10</td>
<td>0.07</td>
</tr>
<tr>
<td>3 GHz to 26.5 GHz</td>
<td>0.10</td>
<td>0.07</td>
</tr>
<tr>
<td><strong>4 MHz BW, High Gain Setting</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 MHz to 15 MHz</td>
<td>0.10</td>
<td>0.07</td>
</tr>
<tr>
<td>15 MHz to 3 GHz</td>
<td>0.10</td>
<td>0.07</td>
</tr>
<tr>
<td>3 GHz to 26.5 GHz</td>
<td>0.10</td>
<td>0.07</td>
</tr>
</tbody>
</table>

1 201 points, 1 noise average  
2 May typically be degraded at frequencies below 500 MHz due to spurious noise receiver residuals.

### Table 56. Noise Receiver Linearity (dB)

---

3095
<table>
<thead>
<tr>
<th>Power Range</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>4 MHz BW, Low Gain Setting</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-34 dBm to -64 dBm</td>
<td>+/-0.05 dB</td>
<td>--</td>
</tr>
<tr>
<td>-64 dBm to -70 dBm</td>
<td>+/-0.10 dB</td>
<td>--</td>
</tr>
<tr>
<td><strong>4 MHz BW, Medium Gain Setting</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-48 dBm to -76 dBm</td>
<td>+/-0.05 dB</td>
<td>--</td>
</tr>
<tr>
<td>-76 dBm to -86 dBm</td>
<td>+/-0.10 dB</td>
<td></td>
</tr>
<tr>
<td><strong>4 MHz BW, High Gain Setting</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-58 dBm to -84 dBm</td>
<td>+/-0.05 dB</td>
<td>--</td>
</tr>
<tr>
<td>-84 dBm to -92 dBm</td>
<td>+/-0.10 dB</td>
<td>--</td>
</tr>
</tbody>
</table>
### Specifications: N5242A Option 029 (continued)

#### Table 57. Noise Receiver Input Range

<table>
<thead>
<tr>
<th>Power Range</th>
<th>Specification</th>
<th>High Gain Setting</th>
<th>Medium Gain Setting</th>
<th>Low Gain Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>500 MHz to 3 GHz</td>
<td>Max DUT NF + Gain (dB)</td>
<td>32 dB</td>
<td>44 dB</td>
<td>55 dB</td>
</tr>
<tr>
<td></td>
<td>Max Input Power (dBm) for &lt;0.1 dB Compression&lt;sup&gt;1&lt;/sup&gt;</td>
<td>&lt;=-57 dBm</td>
<td>&lt;=-45 dBm</td>
<td>&lt;=-34 dBm</td>
</tr>
<tr>
<td></td>
<td>Max Operating Input Power (dBm) Typical</td>
<td>-37 dBm Typical</td>
<td>-33 dBm Typical</td>
<td>-26 dBm Typical</td>
</tr>
<tr>
<td></td>
<td>Limit BW for full NF + Gain</td>
<td>400 MHz</td>
<td>400 MHz</td>
<td>400 MHz</td>
</tr>
<tr>
<td>3 GHz to 26.5 GHz</td>
<td>Max DUT NF + Gain (dB)</td>
<td>46 dB</td>
<td>57 dB</td>
<td>68 dB</td>
</tr>
<tr>
<td></td>
<td>Max Input Power (dBm) for &lt;0.1 dB Compression&lt;sup&gt;1&lt;/sup&gt;</td>
<td>&lt;=-43 dBm</td>
<td>&lt;=-32 dBm</td>
<td>&lt;=-21 dBm</td>
</tr>
<tr>
<td></td>
<td>Max Operating Input Power (dBm) Typical</td>
<td>-23 dBm</td>
<td>-20 dBm</td>
<td>-13 dBm</td>
</tr>
<tr>
<td></td>
<td>Limit BW for full NF + Gain</td>
<td>400 MHz</td>
<td>400 MHz</td>
<td>400 MHz</td>
</tr>
</tbody>
</table>

<sup>1</sup> Derived from 0.25 dB CW compression specification and -5 dB offset, derived from exponential model for device compression. Refer to test port 2.

#### Test Port Output: Option 029

#### Table 58. Max Leveled Power, Port 1 Filtered Mode - Option 219/029, 419/029 (dBm)
<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>12</td>
<td>17</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>3</td>
<td>8</td>
</tr>
</tbody>
</table>

**Specifications: N5242A Option 029 (continued)**

**Table 59. Max Leveled Power, Port 1 High Power Mode - Option 219/029, 419/029 (dBm)**

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>12</td>
<td>18</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>12</td>
<td>19</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>12</td>
<td>17</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>3</td>
<td>8</td>
</tr>
</tbody>
</table>

**Table 60. Max Leveled Power, Port 1 Filtered Mode - Option 224/029, 423/029 (dBm)**
<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>13</td>
<td>19</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>
### Table 61. Max Leveled Power, Port 1 High Power Mode - Option 224/029, 423/029 (dBm)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>12</td>
<td>18</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>13</td>
<td>20</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>13</td>
<td>19</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>

### Table 62. Max Leveled Power, Port 1 Source 1, Combine Mode, Filtered Mode - Option 224/029, 423/029 (dBm)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>6</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>--</td>
<td>9</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>--</td>
<td>9</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>--</td>
<td>15</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>--</td>
<td>11</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>--</td>
<td>8</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>5</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>0</td>
</tr>
</tbody>
</table>
### Table 63. Max Leveled Power, Port 1 Source 1, Combine Mode, High Power Mode - Option 224/029, 423/029 (dBm)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>16</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>--</td>
<td>17</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>--</td>
<td>10</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>--</td>
<td>15</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>--</td>
<td>11</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>--</td>
<td>8</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>5</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 64. Max Leveled Power, Port 1 Source 2, Combine Mode, Filtered Mode - Option 224/029, 423/029 (dBm)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>-8</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>--</td>
<td>-5</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>--</td>
<td>-5</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>--</td>
<td>2</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>--</td>
<td>-2</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>--</td>
<td>-4</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>-7</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>-11</td>
</tr>
</tbody>
</table>
Table 65. Max Leveled Power, Port 1 Source 2, Combine Mode, High Power Mode - Option 224/029, 423/029 (dBm)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>--</td>
<td>2</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>--</td>
<td>4</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>--</td>
<td>-4</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>--</td>
<td>2</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>--</td>
<td>-2</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>--</td>
<td>-4</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>--</td>
<td>-7</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>--</td>
<td>-11</td>
</tr>
</tbody>
</table>

Table 66. Max Leveled Power, Port 2 - Option 219/029, 419/029 (dBm)

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>13</td>
<td>17</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>13</td>
<td>19</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>13</td>
<td>18</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>13</td>
<td>17</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-2</td>
<td>3</td>
</tr>
</tbody>
</table>

Specifications: N5242A Option 029 (continued)

Table 67. Max Leveled Power, Port 2 - Option 224/029, 423/029 (dBm)

3102
<table>
<thead>
<tr>
<th>Frequency</th>
<th>Specification</th>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 50 MHz</td>
<td>13</td>
<td>17</td>
</tr>
<tr>
<td>50 MHz to 500 MHz</td>
<td>13</td>
<td>20</td>
</tr>
<tr>
<td>500 MHz to 3.2 GHz</td>
<td>12</td>
<td>16</td>
</tr>
<tr>
<td>3.2 GHz to 10 GHz</td>
<td>13</td>
<td>17</td>
</tr>
<tr>
<td>10 GHz to 16 GHz</td>
<td>10</td>
<td>14</td>
</tr>
<tr>
<td>16 GHz to 20 GHz</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>20 GHz to 24 GHz</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>24 GHz to 26.5 GHz</td>
<td>-2</td>
<td>4</td>
</tr>
</tbody>
</table>
Test Set Block Diagrams

NOTE: For best readability, use a color printer for printing the following graphics.

Figure 13. 2-Port N5242A Base Unit (Option 200)

Figure 14. 2-Port N5242A (Option 219)
**Figure 15.** 2-Port N5242A (Option 224)

**Figure 16.** 4-Port N5242A Base Unit (Option 400)
Figure 17. 4-Port N5242A (Option 419)

Figure 18. 4-Port N5242A (Option 423)
Figure 19. 2-Port N5242A (Option 219, 224), Showing J-Designators for Rear Panel Connectors

Figure 20. 4-Port N5242A (Option 419, 423), Showing J-Designators for Rear Panel Connectors
**Figure 21.** Receiver Block Diagram

**Figure 22.** 2-Port PNA-X with Noise Figure Hardware (shown with Options 200, 219, 224, & 029).

On 4-Port analyzers (Option 419 or 423) with Option 029, ports 1 and 2 are the same as shown here.

At test port 1 front panel loops, a noise tuner switch connects the noise tuner (ECal module) in series with Source 1, providing several different input impedances.

At test port 2, a noise tuner switch and a coupler route the RF signal from the DUT output to two noise receivers. The appropriate receiver is automatically selected as required for the frequency being measured.
<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6-Mar-08</td>
<td>Provided more resolution for RP IF input.</td>
</tr>
<tr>
<td>6-Mar-08</td>
<td>Show full freq range for RP LO output.</td>
</tr>
</tbody>
</table>
Typical System Performance for the N5250A
(Rev. 2004-08-02)

- This is a complete list of the N5250A network analyzer typical system performance.
- To view or print the .pdf version of this document, visit our web site at http://www.agilent.com, type 5988-9620EN in the Quick Search box, then click GO.

See Specs for other PNA models

Definitions

Typical: Expected performance of an average unit which does not include guardbands. It is not covered by the product warranty.

Standard: When referring to the analyzer, this includes no options unless noted otherwise.

Unlike the lengthy specifications documents for other PNA models, this document presents typical system performance for the following categories only:

- System Dynamic Range
- Test Port Power
- Noise Floor
- Test Port Damage Level
- Option H11 Rear Panel Connectors

Typical System Performance

Table 1. System Dynamic Range
<table>
<thead>
<tr>
<th>Frequency</th>
<th>1.0 mm Test Port</th>
<th>1.85 mm PNA Test Port</th>
<th>Waveguide Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>63 dB</td>
<td>65 dB</td>
<td></td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>94 dB</td>
<td>97 dB</td>
<td></td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>120 dB</td>
<td>123 dB</td>
<td></td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>116 dB</td>
<td>123 dB</td>
<td></td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>111 dB</td>
<td>121 dB</td>
<td></td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>100 dB</td>
<td>112 dB</td>
<td></td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>92 dB</td>
<td>107 dB</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>84 dB</td>
<td>101 dB</td>
<td></td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>85 dB</td>
<td>103 dB</td>
<td></td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>80 dB</td>
<td>100 dB</td>
<td></td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>75 dB</td>
<td>95 dB</td>
<td></td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>68 dB</td>
<td></td>
<td>82 dB</td>
</tr>
<tr>
<td>70 GHz to 75 GHz</td>
<td>74 dB</td>
<td></td>
<td>87 dB</td>
</tr>
<tr>
<td>75 GHz to 80 GHz</td>
<td>85 dB</td>
<td></td>
<td>98 dB</td>
</tr>
<tr>
<td>80 GHz to 100 GHz</td>
<td>89 dB</td>
<td></td>
<td>101 dB</td>
</tr>
<tr>
<td>100 GHz to 110 GHz</td>
<td>87 dB</td>
<td></td>
<td>98 dB</td>
</tr>
</tbody>
</table>

Table 2. Test Port Power

<table>
<thead>
<tr>
<th>Frequency</th>
<th>1.0 mm Test Port (Std Configuration\textsuperscript{a} or Opt 017\textsuperscript{b})</th>
<th>1.85 mm PNA Port</th>
<th>Waveguide Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 45 MHz</td>
<td>-8 dBm</td>
<td>-7 dBm</td>
<td></td>
</tr>
<tr>
<td>45 MHz to 500 MHz</td>
<td>-3 dBm</td>
<td>-1 dBm</td>
<td></td>
</tr>
<tr>
<td>500 MHz to 2 GHz</td>
<td>0 dBm</td>
<td>2 dBm</td>
<td></td>
</tr>
<tr>
<td>2 GHz to 10 GHz</td>
<td>-2 dBm</td>
<td>2 dBm</td>
<td></td>
</tr>
<tr>
<td>10 GHz to 24 GHz</td>
<td>-5 dBm</td>
<td>0 dBm</td>
<td></td>
</tr>
<tr>
<td>24 GHz to 30 GHz</td>
<td>-7 dBm</td>
<td>0 dBm</td>
<td></td>
</tr>
<tr>
<td>Frequency</td>
<td>1.0mm Test Port</td>
<td>1.85mm Test Port</td>
<td>Waveguide Port</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------</td>
<td>------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>30 GHz to 40 GHz</td>
<td>-10 dBm</td>
<td>-1 dBm</td>
<td></td>
</tr>
<tr>
<td>40 GHz to 45 GHz</td>
<td>-15 dBm</td>
<td>-5 dBm</td>
<td></td>
</tr>
<tr>
<td>45 GHz to 50 GHz</td>
<td>-12 dBm</td>
<td>-1 dBm</td>
<td></td>
</tr>
<tr>
<td>50 GHz to 60 GHz</td>
<td>-17 dBm</td>
<td>-4 dBm</td>
<td></td>
</tr>
<tr>
<td>60 GHz to 67 GHz</td>
<td>-22 dBm</td>
<td>-8 dBm</td>
<td></td>
</tr>
<tr>
<td>67 GHz to 70 GHz</td>
<td>-9 dBm</td>
<td></td>
<td>-2 dBm</td>
</tr>
<tr>
<td>70 GHz to 75 GHz</td>
<td>-7 dBm</td>
<td></td>
<td>0 dBm</td>
</tr>
<tr>
<td>75 GHz to 80 GHz</td>
<td>-6 dBm</td>
<td></td>
<td>+1 dBm</td>
</tr>
<tr>
<td>80 GHz to 100 GHz</td>
<td>-5 dBm</td>
<td></td>
<td>+1 dBm</td>
</tr>
<tr>
<td>100 GHz to 110 GHz</td>
<td>-8 dBm</td>
<td></td>
<td>-2 dBm</td>
</tr>
</tbody>
</table>

a Assumes a 30” cable from the PNA 1.85mm Test Port Out is used to provide the 10 MHz to 67 GHz source signal. The Standard configuration does not have a bias tee in the 1.0mm head.

b Assumes a 30” cable from the PNA Source Out bulkhead connector is used to provide the 10 MHz to 67 GHz source signal. Option 017 includes a bias tee in the 1.0mm head.

**Table 3: Noise Floor**
<table>
<thead>
<tr>
<th>Frequency</th>
<th>1.0mm Test Port</th>
<th>1.85mm Test Port</th>
<th>Waveguide Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>67 GHz to 70 GHz</td>
<td>-77 dBm</td>
<td>-84 dBm</td>
<td></td>
</tr>
<tr>
<td>70 GHz to 75 GHz</td>
<td>-81 dBm</td>
<td>-87 dBm</td>
<td></td>
</tr>
<tr>
<td>75 GHz to 80 GHz</td>
<td>-91 dBm</td>
<td>-97 dBm</td>
<td></td>
</tr>
<tr>
<td>80 GHz to 100 GHz</td>
<td>-94 dBm</td>
<td>-100 dBm</td>
<td></td>
</tr>
<tr>
<td>100 GHz to 110 GHz</td>
<td>-95 dBm</td>
<td></td>
<td>-100 dBm</td>
</tr>
</tbody>
</table>

Table 4. Test Port Damage Level

<table>
<thead>
<tr>
<th>Frequency</th>
<th>1.0mm Test Port</th>
<th>1.85mm Test Port</th>
<th>Waveguide Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 MHz to 110 GHz</td>
<td>27 dBm</td>
<td>27 dBm</td>
<td>27 dBm</td>
</tr>
</tbody>
</table>

Table 5. Option H08 & H11 Rear Panel Connectors (typical)

<table>
<thead>
<tr>
<th>IF Connectors</th>
<th>A, R1, R2, B (BNC Connectors)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF Connector Input Frequency</td>
<td>8 1/3 MHz</td>
</tr>
<tr>
<td>Nominal Input Impedance at IF Inputs</td>
<td>50</td>
</tr>
<tr>
<td>RF Damage Level to IF Connector Inputs</td>
<td>-20.0 dBm</td>
</tr>
<tr>
<td>DC Damage Level to IF Connector Inputs</td>
<td>25 volts</td>
</tr>
<tr>
<td>0.1 dB Compression Point at IF Inputs</td>
<td>-27.0 dBm</td>
</tr>
<tr>
<td>Pulse Input Connectors</td>
<td>A, R1, R2, B (BNC Connectors)</td>
</tr>
<tr>
<td>Nominal Input Impedance at Pulse Inputs</td>
<td>1 Kohm</td>
</tr>
<tr>
<td>Minimum IF Gate Width</td>
<td>20 ns for less than 1 dB deviation from theoretical performance</td>
</tr>
<tr>
<td>DC Damage Level to Pulse Connector Inputs</td>
<td>5.5 volts</td>
</tr>
<tr>
<td>Drive Voltage</td>
<td>TTL (0, +5.0) Volts</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Rear Panel LO Power Test Port Frequency</td>
<td></td>
</tr>
<tr>
<td>(see 836x H11 Specs for Test Port Frequencies up to 67 GHz)</td>
<td></td>
</tr>
<tr>
<td>67 GHz to 110 GHz³</td>
<td>-7 to 13 dBm</td>
</tr>
</tbody>
</table>

| Rear Panel RF Power - Test Port Frequencies  |
| (see 836x H11 Specs for Test Port Frequencies up to 67 GHz)  |
| 67 GHz to 76 GHz² | -4 to 10 dBm |
| 76 GHz to 96 GHz⁴ | +1 to 5 dBm |
| 96 GHz to 110 GHz⁴ | +5 to 1 dBm |

1 Pulse input connectors are operational only with Option H08 (Pulse Measurement Capability) enabled.
2 Based on deviation from signal reduction equation: Signal Reduction (dB) = 20log₁₀(Duty_cycle) = 20log₁₀(pulse_width/period). Measured at Pulse Repetition Frequency (PFR) of 1 MHz.
3 For rear panel LO port frequency, divide by 8.
4 For rear panel RF port frequency, divide by 6.

**Note:** Typical system performance for front panel jumpers is not provided for the N5250A.

**Test Set Block Diagram**

N5250A - Standard Network Analyzer
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>

**Test Set with Option 016 Block Diagram**

N5250A - Option 016 Receiver Attenuators Network Analyzer
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SOURCE OUT</td>
<td>h</td>
<td>RCVR B IN</td>
</tr>
<tr>
<td>b</td>
<td>RCVR R1 IN</td>
<td>i</td>
<td>CPLR ARM</td>
</tr>
<tr>
<td>c</td>
<td>SOURCE OUT</td>
<td>j</td>
<td>PORT 2</td>
</tr>
<tr>
<td>d</td>
<td>CPLR THRU</td>
<td>k</td>
<td>CPLR THRU</td>
</tr>
<tr>
<td>e</td>
<td>PORT 1</td>
<td>l</td>
<td>SOURCE OUT</td>
</tr>
<tr>
<td>f</td>
<td>CPLR ARM</td>
<td>m</td>
<td>RCVR R2 IN</td>
</tr>
<tr>
<td>g</td>
<td>RCVR A IN</td>
<td>n</td>
<td>SOURCE OUT</td>
</tr>
</tbody>
</table>
Glossary

12-Term Error Correction See Error Correction, 12-Term.

1-Port Device A device with a single connector or path to the device's circuitry. Examples include an oscillator and a load.

2-Port Calibration, Full See Error Correction, 12-Term.

2-Port Device A device with two connectors or other paths to the device’s circuitry. Examples include filters, SAW devices, attenuators, matching pads, and amplifiers.

3-Term Error Correction See Error Correction, 3-Term.

A

Active Channel The highlighted channel affected by front panel functions.

Active Function Readout The area of a display screen where the active function and its state are displayed. The active function is the one that was completed by the last key selection or remote programming command.

Active Marker The marker on a trace that can be repositioned either by front panel controls or by programming commands.

Active Trace A trace that is being swept (updated) with incoming signal information.

ADC Analog to Digital Converter

Address The identification (represented by a name, label, or number) for a register, location in storage, or any other data source or destination. Examples are the location of a station in a communications network, or a device on the GP-IB.

ADM Add-Drop Multiplexer

Admittance (Y) The inverse of an impedance (i.e. the ratio of current to voltage). Complex admittances take the form \( Y = G + jB(t) \).

ALC Automatic Level Control. See Automatic Gain Control.

AM Amplitude Modulation

AM Group Delay A technique for the measurement of group delay through a device which utilizes an amplitude modulated (AM) source. Note: The actual delay of the modulation envelope is measured directly with an external scalar detector. Devices that distort the amplitude of a signal cannot be measured. These include amplifiers with automatic gain control (AGC) and devices subject to saturation or power limiting.

Amplitude Modulation The process, or result of the process, of varying the amplitude of a carrier signal. The resulting modulated carrier contains information that can be recovered by demodulation. See also Modulation.

Analog The general class of devices or circuits in which the output varies as a continuous function of the input.

Annotation The labeling of specific information on the display (such as frequency or power).

ANSI American National Standards Institute: A national membership organization (open to manufacturers, organizations, users, and communications carriers) that approves standards, accredits standards development groups and certificate programs, and represents and coordinates US interests in non-treaty and non-government
standards bodies.

**Aperture** The frequency span of the network analyzer used for calculating group delay. The narrower the aperture, the finer the resolution of the group delay variations, but noise is reduced by increasing the aperture.

**Array** A set of numbers or characters that represents any given function.

**ASCII** American Standard Code for Information Interchange

**Attenuation** Denotes a reduction in signal amplitude. The difference between transmitted and received power due to loss through equipment, lines, or other transmission devices; usually expressed in decibels.

**Attenuator** An RF or microwave device used to reduce the power level of a signal by precise, incremental amounts over its entire frequency range.

**Automatic Calibration System** AutoCal: Feature offered on Rohde&Schwarz network analyzers.

**Automatic Gain Control (AGC)** A circuit used in amplifiers and other active devices to keep its RF power level constant as other parameters change, such as frequency. Synonym: Automatic Leveling Control (ALC)

**Autoscale** An analyzer feature that evaluates waveforms and adjusts controls to stable and enhance the display.

**AUX** Auxiliary; refers to rear-panel input connector.

**Averaging** A noise reduction technique that computes each data point based on consecutive sweeps and weighted by a user-specified averaging factor. Each new sweep is averaged into the trace until the total number of sweeps is equal to the averaging factor.

**B**

**B/R** The ratio of data sampled at B to the data sampled at R.

**Band Pass** A range of frequencies that are passed through a device, such as a filter. Frequencies not within the band pass are limited or attenuated. See also **Cutoff Frequency**.

**Bandwidth (BW)** The difference between the frequencies of a continuous frequency band within which performance of a device falls within specifications.

**Bandwidth Limit** The condition prevailing when the system bandwidth is exceeded and signal distortion occurs beyond specifications.

**Bandwidth Selectivity** A measure of a filter's ability to resolve signals unequal in amplitude. It is the ratio of the 60 dB bandwidth to the 3 dB bandwidth for a given resolution filter (IF). Bandwidth selectivity tells us how steep the filter skirts are. Bandwidth selectivity is sometimes called shape factor.

**Binary** A method of representing numbers in a scale of two (on or off, high-level or low-level, one or zero). A compact, fast format used to transfer information to and from the analyzer.

**BMP** Bit-Mapped

**Brightness** See **Color Brightness**.

**Broadband Device** A device that operates over a very wide frequency range and exhibits only small variations in response over that range.

**Buffer** A storage device used when transmitting information to compensate for a difference in the rate of flow of information between two devices.

**Burst Carrier** A carrier that is periodically turned off and on. A burst carrier may or may not be modulated.

**BUS** Basic Utility System

**Bus** One or more conductors used as a path to deliver transmitted information from any of several sources to any of several destinations.
BW Bandwidth

Byte Eight bits of data representing one character processed as a unit.

C

CAD Computer Aided Design

CAE Computer Aided Engineering

Calibration In HP instrumentation, the process of periodically (usually annually) verifying an instrument is performing to specifications. A calibration certificate is awarded after verification.

In network analyzers, the process of removing systematic errors from measurements. See Error Correction.

Calibration Kit Hardware and software required to perform error correction on a network analyzer for a specific measurement and/or test set.

Calibration, 2-Port See Error Correction, 12-Term.

Calibration, Blackburn Calibrations of transmission path with corrected source match involving 15 calibration terms. Synonym: 15-term error correction

Calibration, Frequency Response The simplest error correction procedure to perform, but only corrects for a few of the twelve possible systematic error terms. Frequency response corrections can be made for reflection measurements, transmission measurements, and isolation measurements.

Calibration, Interpolation A user selectable network analyzer feature that calculates (interpolates) new error correction terms from existing terms when there is a change in network analyzer parameters, such as IF bandwidth, power, or sweep time. The resulting error correction is not as accurate as completing a full 2-port calibration.

Calibration, Port Extension See Port Extension.

Calibration, Reference Plane See Reference Plane.

Calibration, Set Z Sets the system impedance, usually 50 or 75 ohms.

Calibration, SOLT A calibration using four known standards: Short-Open-Load-Through. Also known as a full two-port calibration and 12-term error correction. See also Error Correction.

Calibration, TRL and LRM A calibration used in environments where the DUT cannot be connected directly to the network analyzer ports, (MMIC, microstrip, beam-lead diodes etc.). Thru-Reflect-Line (TRL) and M (Match) standards are fabricated and used because known high-quality standards are not readily available. The requirements for characterizing these standards are less stringent, but the calibration is not as accurate as the traditional full two-port calibration using S-O-L-T standards. The terms are used interchangeably (TRL, LRL, LRM etc.) but they all refer to the same basic calibration method.

Characteristic Impedance The impedance looking into the end of an infinitely long lossless transmission line.

Color Brightness A measure of the intensity (brightness) of a color.

Command A set of instructions that are translated into instrument actions. The actions are usually made up of individual steps that together can execute an operation.

Continuous Sweep Mode The analyzer condition where traces are automatically updated each time trigger conditions are met.

Controller A device capable of specifying the talker and listeners for an information transfer. An external computer connected to an instrument to control its operation.

Corrected Measurements made after performing error correction.

Coupler See Directional Coupler.
**CPU** Central Processing Unit

**Crosstalk** The occurrence of a signal at one port of a device being affected by a signal in any other path. Isolation is the measurement of crosstalk.

**Cursor** An electronically generated pointer that moves across the display to manipulate controls.

**Cutoff Frequency** In filters, the frequency at which attenuation is 3dB below the band pass signal level, known as the 3dB points.

**CW** Continuous wave: A single frequency (rather than a swept frequency).

---

**DAC** Digital to Analog Converter

**dB** Decibel: a relative unit of measure. The ratio in dB is given by: $10 \log_{10} (P_1/P_2)$ where $P_1$ and $P_2$ are the measured powers. The dB is preferred instead of arithmetic ratios or percentages because when components are connected in series, their effect on power, expressed in dB, may be arithmetically added and subtracted. For example, if a 3dB attenuator is connected to a 10dB amplifier, the net gain of the two components is $(-3dB + 10dB = +7dB)$.

**dBm** Absolute unit of measure in decibels: $0dBm = 1 \text{ mW}$. The conventions of the dB (adding and subtracting) continue to apply.

**DBMS** Database Management System

**DC** Direct Current

**Default** A known set of conditions used in the absence of user-defined conditions.

**Delay** See [Group Delay](#).

**Demodulation** The process of recovering from a modulated carrier, information in the form of a signal having essentially the same characteristics as the original modulating signal. Recovery of the modulating signal accomplished by signal detection.

**Detection** The process of demodulating signal carriers. There are two basic ways of providing signal detection in network analyzers: Diode detectors (used in broadband applications) and heterodyning, (used in narrowband applications).

**Detector, Diode** A device used to convert a RF signal to a proportional DC level. If the signal is amplitude modulated, the diode strips the RF carrier signal from the modulation. Many sources used with scalar analyzers are amplitude modulated with a 27.778 kHz signal and then detected in the network analyzer. Phase information on the signal carrier is lost in diode detection.

**Deviation from Linear Phase** Linear phase refers to the nature of the phase shift of a signal through a device. The phase is linear if a plot of phase shift versus frequency is a straight line using linear scales. Deviation from linear phase causes signal distortion.

**Digital** Pertaining to the class of devices or circuits in which the output varies in discrete steps.

**Digital Demodulation** Describes a technique of extracting the information used to modulate a signal. Digital signal processing algorithms are used on the signal after it has been converted from an analog to a digital form (digitized).

**Dimension** To specify the size of an array. The number of array rows or columns.

**Directivity** In a 3-port directional coupler, the ratio of the power present at the auxiliary port when the signal is traveling in the forward direction to the power present at the auxiliary port when the same signal is traveling in the reverse direction.

**Directional Coupler** A 3-port device typically used for separately sampling the backward (reflected) wave in a
transmission line.

**Disk** A circular, magnetic storage medium.

**Display**

Verb: To show annotation and measurement data on the display.

**Display Detector Mode** The manner in which analog, video information is processed prior to being digitized and stored in memory.

**Display Dynamic Accuracy** The amplitude uncertainty, usually in dB, over the display dynamic range.

**Display Dynamic Range** The amplitude range, in dB, over which the display dynamic accuracy applies.

**Display Formats** Graphical formats for displaying measurement data. These include single channel, overlay (multiple traces on one graticule), split (each trace on separate graticules).

**Display Modes** The ways in which measurement data can be presented graphically. On a network analyzer, the choices are Cartesian/rectilinear (XY plot with log or linear magnitude, phase, group delay, SWR, real and imaginary, and dBV, dBmV and dBuV), polar (magnitude and angle), magnitude and phase, and Smith chart. Not all display modes are available on all network analyzers. In addition, displays can present this information in various combinations of traces. Common modes are dual, (the ability to display more than one trace, usually over the same frequency range), and alternate, (the ability to display more than one trace, each with different frequency range and type).

**Display Phase Dynamic Accuracy** The phase measurement uncertainty, usually in degrees, for measurements whose units are in degrees.

**Display Points** The total number of measurement points made in a single measurement. The points can be in units of frequency, power, or time. The number of points often dictates measurement speed, resolution, and aperture.

**Display Trace Noise, Magnitude** The amplitude uncertainty of the trace, in dB, due to random noise in the test system.

**Display Trace Noise, Phase** The phase uncertainty of the trace, in degrees, due to random noise in the test system.

**Display Type** The type of display screen built into the analyzer. Data can be displayed as a raster drawing (a computer-like dot map) or as a vector drawing (lines drawn on the display). Color and display standard can also be specified as monochrome (single color), or color (two or more colors). The format standard may also be specified, such as VGA or SVGA, for IBM-compatible personal computers.

**Distortion** Deterioration of a signal's quality due to the nonlinear characteristics of a device or system transfer function. Distortion is measured as a combination of the changes in amplitude, frequency and phase of signal at the output of a device or system as compared to the signal at the input.

**Drift** The slow change in signal frequency.

**DSP** Digital Signal Processing

**DUT** Device Under Test

**DVM** Digital Volt Meter

**Dynamic Range** In a receiver, the range of signal levels, from minimum to maximum, that can be reliably measured simultaneously. Dynamic range allows small signals to be measured in the presence of large signals. Source power and receiver compression usually limits the maximum boundary to dynamic range. Receiver residual responses and noise floor usually limit the minimum power boundary.
ECal See **Electronic Calibration**.

**Electrical Delay** A simulated variable length of lossless transmission line, added to or subtracted from a receiver input, to compensate for interconnecting cables. The firmware equivalent of mechanical or analog "line stretchers" in other network analyzers.

**Electronic Calibration (ECal)** A calibration system for electronic calibration of RF and microwave vector network analyzers. The electronic calibration system creates a twelve-term, two-port error model and then provides a confidence check of the calibration. The Ecal system consists of a repeatable, variable-impedance, solid-state calibration standard and a mainframe control unit which interfaces with the 8510, 8720 series, and the 8753 network analyzers or a USB module which interfaces with the PNA series network analyzers.

**EMC** Electro-Magnetic Compatibility

**EMI** Electro-Magnetic Interference: Unintentional interfering signals generated within or external to electronic equipment. Typical sources could be power-line transients, noise from switching-type power supplies and/or spurious radiation from oscillators. EMI is suppressed with power-line filtering, shielding, etc.

**Engage** To activate a function.

**Enter** The process of inputting information.

**EPROM** Electronically Programmable, Read-Only Memory

**Error Correction** In network analyzers, a process that removes or reduces systematic (repeatable) measurement errors by measuring known standards from a calibration kit. Synonym: measurement calibration

**Error Correction, 3-Term** Used to remove systematic measurement errors on a device with one port, such as a load.

**Error Correction, 12-Term** Correction for a two port device using six parameters:
- Directivity
- Source match
- Load match
- Reflection frequency response
- Transmission frequency response
- Isolation

To completely characterize a two-port device, these six parameters must be characterized in the forward and reverse directions, making a total of 12 terms. The user usually has the option of omitting isolation from the correction process. Synonym: Full two-port error correction

**Error Correction, 1-Port** Corrects a test set for port 1 or port 2 directivity, frequency response, and source match errors. The process requires three known standard terminations, for example, open, short, and load.

**Error Message** A message on a display that indicates an error condition. Missing or failed hardware, improper user operation, or other conditions that require additional attention can cause an error condition. Generally, the requested action or operation cannot be completed until the condition is resolved.

**ESD** Electro Static Discharge

**Ethernet** A network that adheres to the IEEE 802.3 Local Area Network standard.

**Ethernet address** A hexadecimal number which is used to identify a machine on a network. Each analyzer is assigned a unique Ethernet address at the factory and it is stored in the analyzer's ROM.

**External trigger signal** A TTL signal that is input to an analyzer and initiates a measurement sweep or similar event, making the measurements synchronous with the external triggering source.
Filter A passive device that allows some frequencies to pass and attenuates others, depending on the type and specifications. A high-pass filter passes frequencies above the cutoff frequency, a low-pass filter passes frequencies below the cutoff frequency, and a band-pass filter passes frequencies between two specific frequencies.

Firmware An assembly made up of hardware and instruction code. The hardware and instruction code is integrated and forms a functional set that cannot be altered during normal operation. The instruction code, permanently installed in the circuitry of the instrument, is classified as ROM (read only memory). The firmware determines the operating characteristics of the instrument or equipment.

Flatness The amplitude and phase response of a device under test (DUT), a signal source, a receiver, or a combination of these. See also Frequency Response.

FM Frequency Modulation

Frequency The number of periodic oscillations, vibrations, or waves per unit of time, usually expressed in cycles per second, or Hertz (Hz).

Frequency Accuracy The uncertainty with which the frequency of a signal or spectral component is indicated, either in an absolute sense or relative to another signal or spectral component. Absolute and relative frequency accuracies are specified independently.

Frequency Range The range of frequencies over which a device or instrument performance is specified.

Frequency Resolution The ability of a network analyzer to measure device characteristics at closely spaced frequencies and display them separately. Resolution of equal amplitude responses is determined by IF bandwidth. Resolution of unequal amplitude responses is determined by IF bandwidth and bandwidth selectivity.

Frequency Response The peak-to-peak variation in the displayed amplitude response over a specified center frequency range. Frequency response is typically specified in terms of dB, relative to the value midway between the extremes.

Frequency Span The magnitude of the displayed frequency component. Span is represented by the horizontal axis of the display. Generally, frequency span is given as the total span across the full display. Some analyzers represent frequency span (scan width) as a per-division value.

Frequency Stability The ability of a frequency component to remain unchanged in frequency or amplitude over short and long-term periods of time. Stability refers to an oscillator's ability to remain fixed at a particular frequency over time.

Front Panel Key Keys that are located on the front panel of an instrument. The key labels identify the function the key activities. Numeric keys and step keys are two examples of front panel keys.

Full 2-Port Calibration See Error Correction, 12-Term.

Function The action or purpose that a specific item is intended to perform or serve. The network analyzer contains functions that can be executed via front panel key selections, or through programming commands. The characteristics of these functions are determined by the firmware in the instrument. In some cases, a DLP (downloadable program) execution of a function allows you to execute the function from front panel key selections.

Fundamental Frequency In any waveform, the lowest frequency component; all other components are harmonics. A pure sinusoid has only one component, the fundamental.

Gb Gigabit

GB Gigabyte
GHz Gigahertz

GIF Graphics Interchange Format - Standard graphic format to store bitmapped graphics files.

Giga Prefix for one billion.

GP I/O General Purpose Input / Output; a connector usually on the back of an instrument that allows communication with other test equipment, external test sets, switches, and computers that enable the instrument to be triggered or to trigger external equipment. An example is a foot switch that continues or cycles a measurement, allowing the operator to use both hands on the test hardware.

GPIB General Purpose Interface Bus - IEEE 488 bus is interconnect bus and protocol, allows linking of instruments and computer.

Graticule (or Grid) Enclosed area where waveform is displayed on instrument. Tick marks, on frame or axis, are a scaling aid for making visual measurements.

Group Delay A measure of the transit time of a signal through a DUT versus frequency. Group delay can be calculated by differentiating the DUT's insertion-phase response with respect to frequency. See also AM Group Delay and Deviation from Linear Phase.

GUI Graphical User Interface

Hardcopy Paper copy of data.

Hardkey A front-panel key, which engages a single analyzer function or presents a single menu of softkeys.

Horizontal Reference See Reference Level.

Horizontal Resolution The analyzer's ability to take closely spaced horizontal data points over the full sweep.

Host Computer A computer or device on a network that provides end users with services such as computation and database access and that usually performs network control functions.

Host Name A unique name that is used to identify each host machine on a network. The host name is directly linked to, and can usually be used in place of, the IP address. The user or the system administrator usually creates the host name.

HP Hewlett-Packard Company

HPGL Hewlett-Packard Graphics Language

HP-IB Hewlett-Packard Interface Bus. A parallel interface that allows "daisy chaining" of more than one device to a port on a computer or instrument. Interface protocol is defined in IEEE 488.2; equivalent to the industry standard GPIB.

HTTP HyperText Transfer Protocol: Used to carry World Wide Web (WWW) traffic.

Hue The dimension of color referred to a scale of perceptions ranging from red through yellow, green, and blue, and back to red. A particular gradation of color, tint, shade.

I/O Input/Output

I/O Path Input/Output Path

IEEE Institute of Electrical and Electronic Engineers

IF Intermediate Frequency: the frequency at which a signal is processed after mixing.
**Impedance**  The ratio of voltage to current at a port of a circuit, expressed in ohms.

**Initialize**  The process that assigns information locations to a disk to prepare the magnetic media to accept files.

**Input**  A path intended for putting a signal into an instrument.

Most network analyzers have either 3 (labeled A, B, and R) or 4 inputs (labeled A, B, R1, and R2). Inputs are not the same as channels.

**Input Attenuator**  An attenuator between the input connector and the first mixer of a spectrum analyzer (also called an RF attenuator). The input attenuator is used to adjust the signal level incident to the first mixer, and to prevent gain compression due to high-level or broadband signals. It is also used to set the dynamic range by controlling the degree of internally-generated distortion. For some analyzers, changing the input attenuator settings changes the vertical position of the signal on the display, which then changes the reference level accordingly. In Agilent microprocessor-controlled analyzers, the IF gain is changed to compensate for changes in input attenuator settings. Because of this, the signals remain stationary on the display, and the reference level is not changed.

**Insertion Loss**  The difference between the power measured before and after the insertion of a device. The attenuation between the input and output of a device.

**Intensity**  Brightness; emitting or reflecting light; luminosity.

**Interface**  A connection that allows a common communication link between two or more instruments.

**Intermodulation Distortion**  Undesired frequency components resulting from the interaction of two or more spectral components passing through a device having nonlinear behavior, such as a mixer or an amplifier. The undesired components are related to the fundamental components by sums and differences of the fundamentals and various harmonics. The algorithm is: $f_1 \pm f_2$, $2xf_1 \pm f_2$, $2xf_2\pm f_1$, $3xf_1 \pm 2xf_2$, and so on.

**Internet**  The connection of two or more distinct networks. Often a gateway or router is used to make the connection.

**Interpolate**  To determine a value of a signal between to adjacent points by a procedure or algorithm.

**IP**  Internet Protocol

**IP Address**  Internet protocol address: a unique number that is assigned to each device which is to be connected to a TCP/IP network. Before using an analyzer on a network, your network administrator will need to assign an IP address. An IP address consists of a 32-bit value presented in decimal dot notation: 4 octets (bytes) separated by a dot.

**ISDN**  Integrated Services Digital Network: A standard digital service capability that features one or more circuit-switched communication channels capable of carrying digital voice, data, or image signals, a packet-switched channel for out-of-band signaling and control. In addition, ISDN provides a collection of standard and optional features that support information productivity for the user, providing higher-speed Internet access than analog systems.

**ISO**  International Standards Organization

**Isolation**  A specification or measure of the immunity that one signal has to being affected by another adjacent signal. The occurrence is known as crosstalk.

**Isolator**  An RF device used for providing isolation between paths and components. Made from a 3-port circulator, the third port being terminated in a 50ohm load.
Kilo Prefix for one thousand.
KB Kilobyte
Kb/s Kilobytes per second

L
LAN Local Area Network
LANS Local Area Network System
LCD Liquid Crystal Display
LED Light Emitting Diode
LIF Logical Interchange Format (used for older HP disk drives/computers)
Limit Lines Lines input by the user that overlay the analyzer's measurement data to allow automatic detection of data that is out of the acceptable range. Pass/Fail annotation, audio alarms, or electronic output can be triggered to notify the operator or on-line computer program of the over-limit condition.
Limit-Line File The user-memory file that contains the limit-line table entries.
Limit-Line Table The line segments of a limit line are stored in the limit-line table. The table can be recalled to edit the line segments, then restored in the limit-line file.
Linear Device A device in which the output is continuously proportional to the input.
LO Local Oscillator. In a superheterodyne system, the LO is mixed with the received signal to produce a sum or difference equal to the intermediate frequency (IF) of the receiver.
LO Feedthrough The response that in a superheterodyne system when the first local oscillator frequency is equal to the first IF.
Load A one port microwave device used to terminate a path in its characteristic impedance.
Load Match A measure of how close the device’s terminating load impedance is to the ideal transmission line impedance. Match is usually measured as return loss or standing wave ratio (SWR) of the load.
Local Lock Out A condition or command that prevents analyzer front-panel entries (and disables the Local key).
Local Operation To operate manually from the front panel.
Log Logarithm
Log Display The display mode in which vertical deflection is a logarithmic function of the input signal amplitude. Log display is also called logarithmic display. The display calibration is set by selecting the value of the reference level position and scale factor in dB per division.
LRM Line-Reflect-Match. See Calibration, TRL, and LRM.

M
Magnitude The amplitude of a signal measured in its characteristic impedance without regard to phase. See also Scalar.
Marker A graphical symbol along a display trace that is annotated with measurement characteristics of that specific data point.
Marker Functions Mathematical or statistical computation on the data of one or more markers to provide the operator more information. For example, the marker delta function calculates and displays the difference between two markers.
Maximum Input Level The maximum signal power that may be safely applied to the input of an analyzer. The maximum input level is typically 1 W (+30 dBm) for Agilent spectrum analyzers.

MB Megabyte

Measurement Uncertainty The quantified amount of error in a measurement situation. Calibrations are intended to reduce the amount of uncertainty. The following are sources of measurement errors that lead to uncertainty:

- Systematic errors (imperfections in calibration standards, connectors, cables, and instrumentation)
- Random errors (noise, connector repeatability)
- Drift (source and instrumentation)

Mega Prefix for one million.

Memory A storage medium, device, or recording medium into which data can be stored and held until some later time, and from which the entire original data may be retrieved.

Memory Card A small memory device shaped like a credit card that can store data or programs.

Menu The analyzer functions that appear on the display and are selected by pressing front panel keys. These selections may invoke a series of other related functions that establish groups called menus.

MHz Megahertz

milli Prefix for one-thousandth.

Modem Modulator/Demodulator

Modulation The process, or the result of the process, of varying a characteristic of a carrier signal with an information-bearing signal, causing the carrier to contain the information. See AM and FM.

Monitor Any external display.

Monochrome Having only one color (chromaticity).

ms Millisecond

mW Milliwatt: one thousandth of a watt

Multisync A type of monitor that can synchronize its horizontal sweep to various frequencies within a specified range.

N

Narrowband In network analysis, the frequency resolution of the analyzer's receiver that is sufficiently narrow to resolve the magnitude and phase characteristics of narrowband devices. The reduced receiver bandwidth usually decreases the noise floor of the receiver, providing more measurement amplitude range.

Narrowband Device A device whose transfer characteristics are intended to operate over a very narrow frequency range and are designed to provide well-defined amplitude responses in that range, such as a band pass filter.

Network Analysis The characterization of a device, circuit, or system derived by comparing a signal input going into the device to a signal or signals coming out from the device.

NIST National Institute of Standards and Technology

Nit The unit of luminance (photometric brightness) equal to one candela per square meter.

Noise Random variations of unwanted or disturbing energy in a communications system from man-made and natural sources that affects or distorts the information carried by the signal. See also Signal-to-Noise Ratio.
**Noise Figure** (F): For a two-port device, a measure of how the noise generated inside the device degrades the signal-to-noise ratio of a signal passing through the device at 290 degrees, usually expressed in dB.

**Noise Floor** The analyzer's internal displayed noise. The noise level often limits how small a signal magnitude can be measured. In network analysis, noise floor is measured with the test ports terminated in loads, full two-port error correction, 10 Hz IF bandwidth, maximum test port power, and no averaging during the test.

**Non-Insertable Devices** In measurement calibration, a device that cannot be substituted for a Zero-Length Through Path. It has the same type and sex connectors on each port, or a different type of connector on each port.

**Nonvolatile Memory** Memory data that is retained in the absence of an ac power source. This memory is typically retained with a battery. Refer also to battery-backed RAM.

**Normalize** To subtract one trace from another to eliminate calibration data errors or to obtain relative information.

---

**O**

**Offset** To move or set off a determined amount. Used in instruments for offsetting frequencies, limits, delay, loss, impedance, etc.

**Output Attenuation** The ability to attenuate the signal, the source, in order to control its power level.

---

**P**

**PC** Personal Computer

**PDF** Portable Document Format (used on the Web)

**Parser, Command** Reads program messages from the input queue of a device in the order they were received from the controller. The parser determines what actions the analyzer should take. One of the most important functions of the command parser is to determine the position of a program message in the analyzer SCPI command tree. When the command parser is reset, the next element it receives is expected to arise from the base of the analyzer command tree.

**Peak Search** A function on an analyzer that searches for the largest response and places a marker on it.

**Phase** The fractional part of a cycle through which an oscillation has advanced, measured from an arbitrary starting point; usually measured in radians or degrees. In network analysis, the phase response of the device under test is the change in phase as a function of frequency between the input stimulus and the measured response.

**Port** The physical input or output connection of an instrument or device.

**Port Extension** Redefining the reference plane to other than that established at calibration. A new reference plane is defined in seconds of delay from the test set port.

**Positive Peak** The maximum, instantaneous value of an incoming signal.

**Postscript (.ps files)** Stores bitmapped graphics files in an encapsulated format for direct use by postscript printers.

**Power, Max Input** The upper limit to input power for which the specifications apply. Some specifications may have different levels of maximum inputs. For example, compression power maximum is usually higher than the harmonic distortion maximum.

**Power, Safe Input** The input power, usually in dBm, allowed without damaging the instrument.

**Preset** A pre-defined instrument state (that also runs an analyzer self-test). The action of pushing the Preset key.

**Protocol** A set of conventions that specify how information will be formatted and transmitted on a network, and how machines on a network will communicate.
Q

**Q or Q Factor** The ratio of energy stored to energy lost in a resonant circuit. High Q indicates a sharp resonance response over frequency.

**Query** Any analyzer programming command having the distinct function of returning a response. These commands may end with a question mark (?). Queried commands return information to the computer.

R

**r + jx** Expression for complex impedance, where r represents the resistive portion and x represents the reactive portion.

**R Channel Reference Channel**

**RAM** Random Access Memory, or read-write memory: A storage area allowing access to any of its storage locations. Data can be written to or retrieved from RAM, but data storage is only temporary. When the power is removed, the information disappears. User-generated information appearing on a display is RAM data.

**ROM** Read Only Memory: A storage area that can be read only; it cannot be written to or altered by the user. In instruments, the storage area that contains the "brains" or operational programming; the firmware.

**Receiver** A circuit or system designed for the reception and/or measurement of signals in a specified frequency spectrum.

**Receiver Dynamic Range** See Dynamic Range.

**Reference Level** An instrument function that allows the user to set the amplitude value at the reference position. On network analyzers, the reference position is also selectable. On some spectrum analyzers, the reference position is fixed at the top of the display.

**Reference Plane** The electrical location at which a network analyzer assumes the system connectors and fixturing ends and the DUT begins. The reference plane is set by using calibration standards with known electrical length. The closer the reference plane is to the device under test (DUT), the better the characterization of the device because of the elimination of test system uncertainties.

**Reference Receiver** In a network analyzer, the receiver that measures signals as they come out of the source, before they are incident on the test port and DUT. Typically, these signals are used to compare with the signal at the Test Port Receiver, to determine the affect that the DUT has on the signal. In a 2-port network analyzer, these are typically named ‘R1' (port 1) and ‘R2' (port 2). See a block diagram of the receivers in your PNA.

**Reflection** The phenomenon in which a traveling wave strikes a discontinuity and returns to the original medium.

**Reflection Coefficient** The ratio of the reflected voltage to the incident voltage into a transmission line or circuit. If a transmission line is terminated in its characteristic impedance, the reflection coefficient is zero. If the line is shorted or open the coefficient is 1. See also Return Loss and SWR.

**Reflection Measurements** Measurements that characterize the input and /or output behavior of the device under test (DUT). Measured as the ratio of the reflected signal to the incident signal as a function of frequency. Parameters are called return loss, reflection coefficient, impedance, and standing wave ratio (SWR), all as a function of frequency. See also S-Parameters.

**Remote** A mode of operation where another device (or computer) controls an instrument via the HP-IB. In this mode, the instrument front panel keys are disabled. Front panel operation is called local operation.

**Remote Programming** The automatic operation of an instrument by a computer, usually through a HP-IB, LAN, or RS-232 link.

**Resolution** The ability of a receiver to resolve two signals.
Resolution Bandwidth  The ability of a spectrum analyzer to display adjacent responses discretely (Hertz, Hertz decibel down). This term is used to identify the width of the resolution bandwidth filter of a spectrum analyzer at some level below the minimum insertion loss point (maximum deflection' point on the display). Typically, it is the 3 dB resolution bandwidth that is specified, but in some cases the 6 dB resolution bandwidth is specified.

Return Loss  The amount of dB that the reflected signal is below the incident signal. If zero signal is reflected, the impedance of the device is equal to the characteristic impedance of the transmission system, and return loss is infinite. If the entire incident signal is reflected, the return loss is zero. See also S-Parameters, Reflection Coefficient, and SWR.

Reverse Measurement  The measurement of a device from output to input.

RF  Radio Frequency (from approximately 50 kHz to approximately 3 GHz). Usually referred to whenever a signal is radiated through the air.

ROM  Read Only Memory

S

S/N  Signal-to-Noise Ratio

Sampler  An electronic component that captures the signal level and phase across a known impedance at a uniform rate. In Network Analyzers, this sampling rate must be sufficiently high and precisely timed to make accurate measurements. Network analyzers typically have three or four samplers or mixers.

Sampler Bounce  The leakage or crosstalk between a network analyzer's samplers. Delay in this crosstalk caused by leakage transmission propagation, give the interference its "bounce" appearance. Sampler bounce causes an increase in the noise level of the affected channel, reducing the sensitivity of the analyzer.

Saturation  The degree of color purity, on a scale from white to pure color.

Scalar  A quantity that has magnitude but no phase. A network analyzer capable of measuring only magnitude.

Scale Factor  The display vertical axis calibration in terms of units per division.

SCPI  Standard Commands for Programmable Instruments

Screen  The physical surface of the CRT or flat panel upon which the measurement results, setup information, softkey definitions, and other instrument communication is presented.

Self-Test  A group of tests performed at power-up (or at preset) that verify proper instrument operation.

Sensitivity  The minimum input signal required to produce a specified output signal having a specified signal-to-noise ratio, or other specified criteria.

On a spectrum analyzer, the level of the smallest sinusoid that can be observed, usually under optimized conditions of minimum resolution bandwidth, 0 dB input attenuation, and minimum video bandwidth.

The normalized change in YIG component's center frequency resulting from a change in tuning coil current, specified in MHz/mA.

Serial Prefix  The five-character prefix that begins an instrument serial number; used to represent versions of firmware or hardware changes that have occurred.

Server  A device that is configured to provide a service to other devices on a network, such as shared access to a file system or printer.

Signal-to-Noise Ratio  SNR: The ratio of the amplitude of the desired signal to the amplitude of noise signals, usually expressed in dB and in terms of peak values for impulse noise and root-mean-square values for random noise.

Single Sweep Mode  The spectrum analyzer sweeps once when trigger conditions are met. Each sweep is initiated
by pressing an appropriate front panel key, or by sending a programming command.

**Small Signal Gain Compression** A situation when the input signal's measured amplitude is less than its actual level due to overloading of the network analyzer's input mixer; the analyzer is operating nonlinearly. For broadband analyzer detectors, a signal other than the one under test can put the analyzer into this gain compressed mode, thereby making even lower level signals appear at a lower level than actual. The broadband mode measures all the power incident to the analyzer, not just the signals at the frequency of interest.

**Smith Chart** A graphical mapping of the complex reflection coefficient into normalized complex impedance. Circles on the chart represent constant resistance and radiating lines orthogonal to the circles represent constant reactance. The center of the chart represents the characteristic impedance of the transmission system. Any point on the chart defines a single complex impedance. A line on the chart represents changing impedance over frequency.

**SOLT** Short-Open-Load-Through calibration. See also Calibration, SOLT.

**Source** A device that supplies signal power; a sweep oscillator or synthesized sweeper.

**Source Amplitude Accuracy** The amplitude uncertainty, in dB, of the source power readout.

**Source Amplitude Flatness** The amplitude flatness, in dB, of the source power over the frequency range specified.

**Source Frequency Resolution** The smallest unit of frequency which can be set and/or measured, in Hz.

**Source Frequency Time Base Accuracy** A measure of the analyzer's frequency stability measured in parts per million (ppm. or 1 part in 10E6). For example, a stability of ±5.0 ppm means that an analyzer will measure 1 MHz to an accuracy of ±5 x 10-6 x 10E6 Hz = ±5 Hz.

**Source Frequency Time Base Stability** A measure of the analyzer's time base accuracy over time and temperature. Typically the time base accuracy will be specified for 1 year. A typical temperature frequency stability is ±10 ppm for 250 C± 50 C.

**Source Harmonics** The level of harmonics generated by the analyzer's signal source, in dBc from the fundamental.

**Source Match** A measure of how close the signal source impedance is to the ideal transmission line impedance of the test system. Match is usually measured as return loss or standing wave ratio (SWR) of the source.

**Span** The stop frequency minus the start frequency. The span setting determines the horizontal-axis scale of the analyzer display.

**Span Accuracy** The uncertainty of the indicated frequency separation of any two signals on the display.

**S-Parameters (Scattering Parameters)** A convention used to characterize the way a device modifies signal flow using a network analyzer. A two port device has four S-parameters: forward transmission (S21), reverse transmission (S12), forward reflection (S11), and reverse reflection (S22).

**Stop/Start Frequency** Terms used in association with the stop and start points of the frequency measurement range. Together they determine the span of the measurement range.

**Storage States** The number of settings, programs, traces, and other parameters available to be saved, cataloged, and recalled at any one time.

**Storage, Disk** An internal or external digital storage disk for saving test data, instrument settings, IBASIC programs, and other measurement parameters. Storage formats include MS-DOS (R) and HPs standard LIF with binary, PCX, HP-GL, or ASCII data formats.

**Structural Return Loss** Poor return loss in cable due to a periodic fault such as a periodic dent caused by dropping the cable spool or by the cable pulling process during manufacture.

**Supplemental Characteristics** Typical but non-warranted performance parameters, denoted as "typical", "nominal" or "approximate".
Sweep The ability of the source to provide a specified signal level over a specified frequency range in a specified time period. Also see **Sweep Mode** and **Sweep Type**.

In data processing mode, a series of consecutive data point measurements, taken over a sequence of stimulus values.

**Sweep Mode** The way in which a sweep is initiated or selected, e.g., single, continuous, alternate, or chopped.

**Sweep Type** The method of sweeping the source, e.g., linear, log, or frequency step.

**Sweeper** A signal source that outputs a signal that varies continuously in frequency.

**SWR** Standing Wave Ratio, calculated as \( \frac{1 + p}{1 - p} \) where \( p \) is the reflection coefficient.

**Sync** Synchronization, or Synchronized

**Syntax** The grammar rules that specify how commands must be structured for an operating system, programming language, or applications.

**System Dynamic Range** The difference between the maximum receiver input level and the receiver's noise floor. System dynamic range applies to transmission measurements only, since reflection measurements are limited by directivity.

---

**T**

**T/R** See Transmission/Reflection.

**Termination** A load connected to a transmission line or other device.

**Test Limit** The acceptable result levels for any given measurement.

**Test Port** See Port.

**Test Port Receiver** In a network analyzer, the receiver directly behind the test ports, used to measure the signal as it is reflected off, or transmitted through, the DUT. This signal is typically compared with the signal at the Reference Receiver to determine how the DUT affects a signal. In a 2-port network analyzer, these are typically named 'A' (port 1) and 'B' (port 2). See a block diagram of the receivers in your PNA.

**Test Set** The arrangement of hardware (switches, couplers, connectors and cables) that connect a test device input and output to the network analyzer's source and receiver to make s-parameter measurements.

**Third Order Intercept** TOI: The power input to a non-linear device that would cause third order distortion at the same power level. TOI is a measurement to determine the distortion characteristics of a mixer or receiver. The higher the value, the more immune the receiver to internal distortion.

**Thru** Through line: A calibration standard. See Calibration, SOLT.

**Tint** A shade of color; hue.

**Toggle** To switch states, usually to change a function from on to off, or off to on.

**TOM** Thru-Open-Match: A Rohde&Schwarz term to describe a calibration method.

**Trace** A series of data points containing frequency and response information. The series of data points is often called an array. The number of traces is specific to the instrument.

**Tracking** The ability of the analyzer's receiver to tune to the source frequency over the measurement frequency range. Poor tracking results in amplitude and phase errors due to the receiver IF circuits attenuating and delaying the device under test output.

**Transfer Function** The ratio of the output signal to the stimulus signal, both as a function of frequency.

**Transmission** See Transmission Measurements.
Transmission Intermodulation Spurious A measure of the capability of the transmitter to inhibit the generation of intermodulation distortion products. Intermodulation spurious is sometimes called intermodulation attenuation.

Transmission Measurements The characterization of the transfer function of a device, that is, the ratio of the output signal to the incident signal. Most common measurements include gain, insertion loss, transmission coefficient, insertion phase, and group delay, all measured over frequency. See also S-Parameters.

Transmission/Reflection (T/R) Refers to the suite of measurements made by a scalar or vector network analyzer to characterize a device's behavior over frequency. See also S-Parameters.

Transparent Something that is not visible to the user. Usually a procedure that occurs without the user's initiation or knowledge.

Trigger A signal that causes the instrument to make a measurement. The user can select several options for triggering, such as manual, continuous, or external (for synchronizing measurements to an external source).

TRL Through-Reflect-Line. See Calibration, TRL and LRM.

TTL Transistor-Transistor Logic

Two-Port Error Correction See Error Correction, 12-Term.

U

Uncorrected Measurements made without performing error correction.

Uncoupled Channels Stimulus or receiver settings allowed to be set independently for each channel.

UNI User-Network Interface: The point at which users connect to the network.

Units Dimensions on the measured quantities. Units usually refer to amplitude quantities because they can be changed. In analyzers with microprocessors, available units are dBm (dB relative to 1 mW dissipated in the nominal input impedance), dBmV (dB relative to 1 mV), dBW (dB relative to 1 W), V (volts), W (watts).

V

Variable A symbol, the value of which changes either from one iteration of a program to the next, or within each iteration of a program.

Vector A quantity that has both magnitude and phase.

A network analyzer capable of measuring both magnitude and phase.

VEE Visual Engineering Environment (Agilent software product)

Velocity Factor A numerical value related the speed of energy through transmission lines with different dielectrics (.66 for polyethylene). Used in making time domain measurements.

Vertical Resolution The degree to which an instrument can differentiate amplitude between two signals.

Video An electrical signal containing timing, intensity, and often color information that, when displayed, gives a visual image.

Video Bandwidth In spectrum analyzers, the cutoff frequency (3 dB point) of an adjustable low-pass filter in the video circuit. When the video bandwidth is equal to or less than the resolution bandwidth, the video circuit cannot fully respond to the more rapid fluctuations of the output of the envelope detector. The result is a smoothing of the trace, or a reduction in the peak-to-peak excursion, of broadband signals such as noise and pulsed RF when viewed in broadband mode. The degree of averaging or smoothing is a function of the ratio of the video bandwidth to the resolution bandwidth.

Video Filter In spectrum analyzers, a post-detection, low-pass filter that determines the bandwidth of the video.
amplifier. It is used to average or smooth a trace. Refer also to Video Bandwidth.

VNA Vector Network Analyzer

W

Waveform A representation of a signal plotting amplitude versus time.

Wireless A term that refers to a broad range of technologies that provide mobile communications for home or office, and "in-building wireless" for extended mobility around the work area, campus, or business complex. It is also used to mean "cellular" for in-or out-of-building mobility services.

WWW World Wide Web

X

Y

Z

Zero-Length Through Path In a measurement calibration, when the two test cables mate together directly without using adapters or a thru-line. See also Non-Insertable Devices.
<table>
<thead>
<tr>
<th>INDEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>.</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>8</td>
</tr>
</tbody>
</table>

3135
about Network Analyzer 276
absolute output power 635
access, front panel interface 254
accessories- list of 2657
accessories, effects of 407
accuracy of calibration 461
accuracy, improving
accuracy, on low-loss 2-port devices 393
accuracy, phase measurements 387
accuracy, reflection 393
accurate measurement calibrations 461
active channel 257
active entry keys 360
adapter - Centronics
adapter removal cal - ECal 455
adapter removal cal - mechanical stds 455
adapters - using to calibrate 455
adaptor - characterize macro 206
ADC Receiver Measurements 376
add users 46
address - GPIB 2492
adjust the display 360
adjustments 2706
  10 MHz 2706
  3.8 GHz 2704
  LO power 2709
offset lo power adjustment 2711
phase-lock IF gain 2718
receiver calibration 2734
source calibration 2731
administrative tasks
administrator - user 46
administrator password 46
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>admittance chart</td>
<td>344</td>
</tr>
<tr>
<td>AGC loops, testing amps with</td>
<td>680</td>
</tr>
<tr>
<td>Agilent IO libraries</td>
<td>2523</td>
</tr>
<tr>
<td>AgileUpdate</td>
<td>2667</td>
</tr>
<tr>
<td>ALC leveling</td>
<td>311</td>
</tr>
<tr>
<td>aliasing</td>
<td>216</td>
</tr>
<tr>
<td>alternate sweep, crosstalk</td>
<td>406</td>
</tr>
<tr>
<td>AM input, external connector</td>
<td></td>
</tr>
<tr>
<td>amplifier - high power measurements</td>
<td>680</td>
</tr>
<tr>
<td>amplifier parameters</td>
<td>642</td>
</tr>
<tr>
<td>AM-to-PM conversion</td>
<td>637</td>
</tr>
<tr>
<td>analog receivers</td>
<td>376</td>
</tr>
<tr>
<td>analyzer specifications</td>
<td></td>
</tr>
<tr>
<td>annotation - display</td>
<td>360</td>
</tr>
<tr>
<td>antivirus protection</td>
<td>54</td>
</tr>
<tr>
<td>aperture, group delay</td>
<td>674</td>
</tr>
<tr>
<td>app notes</td>
<td>620</td>
</tr>
<tr>
<td>application crashes</td>
<td>2588</td>
</tr>
<tr>
<td>applying cal sets</td>
<td>443</td>
</tr>
<tr>
<td>arbitrary impedance</td>
<td>509</td>
</tr>
<tr>
<td>arbitrary ratio</td>
<td>292</td>
</tr>
<tr>
<td>arbitrary segment sweep</td>
<td>320</td>
</tr>
<tr>
<td>arbitrary Z standard</td>
<td>420</td>
</tr>
<tr>
<td>archiving cal sets</td>
<td>443</td>
</tr>
<tr>
<td>arranging windows</td>
<td>353</td>
</tr>
<tr>
<td>ASCII file types</td>
<td>598</td>
</tr>
<tr>
<td>assignments - cal class</td>
<td>509</td>
</tr>
<tr>
<td>attenuation-source</td>
<td>311</td>
</tr>
<tr>
<td>attenuators, receiver</td>
<td>311</td>
</tr>
<tr>
<td>auto port ext.</td>
<td>554</td>
</tr>
<tr>
<td>auto save</td>
<td>598</td>
</tr>
<tr>
<td>autocheck</td>
<td>2667</td>
</tr>
<tr>
<td>automation</td>
<td></td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>autostart PNA</td>
<td>251</td>
</tr>
<tr>
<td>auxiliary front-panel jumpers</td>
<td></td>
</tr>
<tr>
<td>auxiliary i/o connector</td>
<td>2537</td>
</tr>
<tr>
<td>auxiliary i/o interface control</td>
<td>2531</td>
</tr>
<tr>
<td>auxiliary trigger</td>
<td>334</td>
</tr>
<tr>
<td>averaging, sweep</td>
<td>398</td>
</tr>
<tr>
<td>how to set</td>
<td>398</td>
</tr>
<tr>
<td>avoid spurs</td>
<td>135</td>
</tr>
<tr>
<td>balanced measurements</td>
<td>648</td>
</tr>
<tr>
<td>banded millimeter modules</td>
<td>2743</td>
</tr>
<tr>
<td>banded test heads</td>
<td>2743</td>
</tr>
<tr>
<td>beeping - stop</td>
<td>49</td>
</tr>
<tr>
<td>bias input</td>
<td></td>
</tr>
<tr>
<td>binary file types</td>
<td>598</td>
</tr>
<tr>
<td>blanking - frequency band crossings</td>
<td>303</td>
</tr>
<tr>
<td>block diagram - configurable test set</td>
<td></td>
</tr>
<tr>
<td>block diagram - standard test set</td>
<td></td>
</tr>
<tr>
<td>booster amp with SMC</td>
<td>213</td>
</tr>
<tr>
<td>boot from the recovery partition</td>
<td>53</td>
</tr>
<tr>
<td>broadband LO tuning</td>
<td>200</td>
</tr>
<tr>
<td>C and COM</td>
<td>1868</td>
</tr>
<tr>
<td>cables - front-panel jumpers</td>
<td></td>
</tr>
<tr>
<td>cables - how to ground</td>
<td>634</td>
</tr>
<tr>
<td>cal kit</td>
<td></td>
</tr>
<tr>
<td>cal kit manager</td>
<td>509</td>
</tr>
<tr>
<td>cal set viewer</td>
<td>497</td>
</tr>
<tr>
<td>cal sets</td>
<td>443</td>
</tr>
<tr>
<td>calibrate touchscreen</td>
<td></td>
</tr>
<tr>
<td>calibration</td>
<td>439</td>
</tr>
<tr>
<td>accurate</td>
<td>461</td>
</tr>
<tr>
<td>cal set</td>
<td>443</td>
</tr>
<tr>
<td>Term</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Cal Wizard</td>
<td>424</td>
</tr>
<tr>
<td>class assignments</td>
<td>509</td>
</tr>
<tr>
<td>converter</td>
<td>145</td>
</tr>
<tr>
<td>data-based model</td>
<td>461</td>
</tr>
<tr>
<td>electronic (ECal)</td>
<td>469</td>
</tr>
<tr>
<td>enhanced response</td>
<td>439</td>
</tr>
<tr>
<td>instrument calibration and verification</td>
<td>2686</td>
</tr>
<tr>
<td>isolation (crosstalk)</td>
<td>406</td>
</tr>
<tr>
<td>kits</td>
<td></td>
</tr>
<tr>
<td>mixer</td>
<td>145</td>
</tr>
<tr>
<td>model</td>
<td>461</td>
</tr>
<tr>
<td>data-based polynomial</td>
<td>461</td>
</tr>
<tr>
<td>overview</td>
<td>418</td>
</tr>
<tr>
<td>polynomial model</td>
<td>461</td>
</tr>
<tr>
<td>power calibration</td>
<td>528</td>
</tr>
<tr>
<td>preferences</td>
<td>272</td>
</tr>
<tr>
<td>select a,</td>
<td>439</td>
</tr>
<tr>
<td>standards</td>
<td>420</td>
</tr>
<tr>
<td>types</td>
<td>439</td>
</tr>
<tr>
<td>using thru adapters</td>
<td>455</td>
</tr>
<tr>
<td>validity of</td>
<td>464</td>
</tr>
<tr>
<td>window</td>
<td>424</td>
</tr>
<tr>
<td>calibration kits</td>
<td>420</td>
</tr>
<tr>
<td>creating</td>
<td>509</td>
</tr>
<tr>
<td>databased</td>
<td>461</td>
</tr>
<tr>
<td>list of models</td>
<td>2673</td>
</tr>
<tr>
<td>modifying</td>
<td>509</td>
</tr>
<tr>
<td>waveguide</td>
<td>509</td>
</tr>
<tr>
<td>calibration standards - measuring</td>
<td>464</td>
</tr>
<tr>
<td>care of connectors</td>
<td>623</td>
</tr>
<tr>
<td>change computer name</td>
<td>49</td>
</tr>
<tr>
<td>Term</td>
<td>Page</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>channel settings</td>
<td>257</td>
</tr>
<tr>
<td>channels - copy</td>
<td>372</td>
</tr>
<tr>
<td>channels - traces and windows</td>
<td>257</td>
</tr>
<tr>
<td>characterize adapter macro</td>
<td>206</td>
</tr>
<tr>
<td>choose configuration</td>
<td>2586</td>
</tr>
<tr>
<td>citifiles</td>
<td>598</td>
</tr>
<tr>
<td>class assignments - modify</td>
<td>509</td>
</tr>
<tr>
<td>classes calibration - about</td>
<td>420</td>
</tr>
<tr>
<td>classes-measurement</td>
<td>300</td>
</tr>
<tr>
<td>clean connectors</td>
<td>623</td>
</tr>
<tr>
<td>client for networking</td>
<td>2587</td>
</tr>
<tr>
<td>code translator application</td>
<td></td>
</tr>
<tr>
<td>coefficient, reflection</td>
<td>292</td>
</tr>
<tr>
<td>coefficient, transmission</td>
<td>292</td>
</tr>
<tr>
<td>collections</td>
<td>1857</td>
</tr>
<tr>
<td>COM Data Types</td>
<td>1858</td>
</tr>
<tr>
<td>COM Events</td>
<td>1862</td>
</tr>
<tr>
<td>COM Examples</td>
<td></td>
</tr>
<tr>
<td>COM Fundamentals</td>
<td>1850</td>
</tr>
<tr>
<td>COM Object Model</td>
<td>699</td>
</tr>
<tr>
<td>COM versus SCPI</td>
<td>2580</td>
</tr>
<tr>
<td>combiner</td>
<td>357</td>
</tr>
<tr>
<td>COM-DCOM configuration</td>
<td>1845</td>
</tr>
<tr>
<td>commands for programming</td>
<td></td>
</tr>
<tr>
<td>common mode</td>
<td>648</td>
</tr>
<tr>
<td>compensating for test fixture</td>
<td>542</td>
</tr>
<tr>
<td>compression, gain</td>
<td>56</td>
</tr>
<tr>
<td>computer name</td>
<td>49</td>
</tr>
<tr>
<td>conductive table mat</td>
<td>634</td>
</tr>
<tr>
<td>confidence check</td>
<td>464</td>
</tr>
<tr>
<td>configurable test set</td>
<td></td>
</tr>
<tr>
<td>block diagram</td>
<td></td>
</tr>
<tr>
<td>dynamic range</td>
<td>381</td>
</tr>
</tbody>
</table>

3140
options 2673
configuration - millimeter module 2743
configuration - PNA standard 2673
configuration IF path 2747
configuration RF path 357
configure an external LO source 175
configure for COM-DCOM 1845
configure for GPIB-SICL 2501
connect to disc drive 612
connect to pc 271
connectivity 271
connector care 623
connectors - test port 2673
conversion-loss measurement calibration 145
converter
Copy Channels 372
corrected system performance - specifications

correction level 451
correction ON/off 451
corrupted operating system 53
counter averaging 398
coupled markers 564
coupled trace settings - time domain 216
coupler rolloff, compensate for 407
CPU speed 49
crash - PNA 2588
creating windows 257
crossover cable 2583
crosstalk specification
crosstalk, receiver 406
cti files 598
curves - uncertainty
customize your analyzer screen 360
<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>data trace</td>
<td>360</td>
</tr>
<tr>
<td>status bar</td>
<td>360</td>
</tr>
<tr>
<td>tables</td>
<td>360</td>
</tr>
<tr>
<td>title bars</td>
<td>360</td>
</tr>
<tr>
<td>toolbars</td>
<td>360</td>
</tr>
<tr>
<td>CW time sweep</td>
<td>320</td>
</tr>
<tr>
<td>cycle time</td>
<td></td>
</tr>
<tr>
<td>data transfer</td>
<td></td>
</tr>
<tr>
<td>specifications</td>
<td></td>
</tr>
<tr>
<td>typical</td>
<td></td>
</tr>
<tr>
<td>versus IF bandwidth</td>
<td></td>
</tr>
<tr>
<td>versus points</td>
<td></td>
</tr>
<tr>
<td>data display</td>
<td>344</td>
</tr>
<tr>
<td>data points</td>
<td>385</td>
</tr>
<tr>
<td>data processing - 8510 mode</td>
<td>574</td>
</tr>
<tr>
<td>data processing chain</td>
<td></td>
</tr>
<tr>
<td>data saving</td>
<td>598</td>
</tr>
<tr>
<td>data trace display</td>
<td>360</td>
</tr>
<tr>
<td>data transfer speed</td>
<td>413</td>
</tr>
<tr>
<td>data transfer time - specs</td>
<td></td>
</tr>
<tr>
<td>data-based model for calibration</td>
<td>461</td>
</tr>
<tr>
<td>date and time</td>
<td>49</td>
</tr>
<tr>
<td>default conditions</td>
<td>284</td>
</tr>
<tr>
<td>define data saves</td>
<td>598</td>
</tr>
<tr>
<td>definitions - specs</td>
<td></td>
</tr>
<tr>
<td>defragmenter - disk</td>
<td>52</td>
</tr>
<tr>
<td>delay, electrical</td>
<td>387</td>
</tr>
<tr>
<td>delay, group</td>
<td>674</td>
</tr>
<tr>
<td>accuracy considerations</td>
<td>674</td>
</tr>
<tr>
<td>aperture, what is</td>
<td>674</td>
</tr>
<tr>
<td>what is</td>
<td>674</td>
</tr>
<tr>
<td>why measure</td>
<td>674</td>
</tr>
</tbody>
</table>
E5091 Testset 2751
easy versus secure configuration 2586
ECal 469
  confidence check 464
  models 2657
  procedure 469
  thru methods 455
  user-characterization 477
edge triggering 334
editor - equation 580
effects of accessories 407
electrical delay, using 659
electrical length, add 391
electrically-long device measurements 391
electronic calibration (ECal) 418
electrostatic discharge protection 634
embed and De-Embed a fixture 542
embedded LO meas 200
EMI specification
emissions- display
enhanced response cal  439
ENR file
entry keys  232
entry toolbar
environment specifications
equation editor  580
equations - uncertainty curves
erase memory  266
error correction  451
  no correction  451
  potential degradation  451
  validation calibration accuracy  464
error messages - about  2653
error messages - list of  2592
error terms - 4 port  497
error terms-monitoring  497
errors, measurement  497
  drift  497
  monitoring  497
  sources
errors, random  497
  connector repeatability  497
  instrument noise  497
  switch repeatability  497
errors, systematic  497
  directivity  497
  isolation  497
  load match  497
  reflection tracking  497
  source match  497
  transmission tracking  497
ESD protection  634
ethernet

event log 2588

text of cal set usage 443

texts-programming

texts

excursion 564

expanded math 509

extensions, port 461

external AM input

text CD-RW drive, save file to 598

text detector connector

text monitor - VGA

text PC drive sharing 612

text source control 175

dwell time 175

text source synchronization 662

text testset control 2755

text testset io connector 2542

text trigger 334

F

F1, F2, F3, F4 keys 232

factor - velocity 387

fault detection resolution 216

favorites

FCA 133

file recall 598

file save 598

file types 598

files - manage without a mouse 598

finder for commands

firmware and help upgrade 2667

firmware rev number 276

fixed marker type 564

fixed output converter measurements 166
fixture simulator 542
flatness and gain, small signal 666
floppy disk drive 232
save file to 598
flush thru cal 469
FOM 106
formats - data 344
group delay 344
imaginary 344
linear mag 344
log mag 344
phase 344
polar 344
real 344
rectangular 344
smith chart 344
SWR 344
forward and reverse FCA scalar meas. 135
frequency band crossings 303
frequency blanking 266
frequency converter application 133
calibration overview 145
configure a mixer 166
known issues 132
measurements 135
using the 135
frequency drift 396
frequency offset 106
frequency range 303
PNA models 2673
preset 284
frequency reference, 10 MHz 396
frequency resolution 303
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>frequency, CW</td>
<td>303</td>
</tr>
<tr>
<td>front panel jumpers</td>
<td></td>
</tr>
<tr>
<td>dynamic range</td>
<td>381</td>
</tr>
<tr>
<td>option 015</td>
<td></td>
</tr>
<tr>
<td>option014</td>
<td></td>
</tr>
<tr>
<td>specifications</td>
<td></td>
</tr>
<tr>
<td>front panel keypad not working</td>
<td>2588</td>
</tr>
<tr>
<td>front panel specification</td>
<td></td>
</tr>
<tr>
<td>front panel tour</td>
<td>232</td>
</tr>
<tr>
<td>full computer name</td>
<td>49</td>
</tr>
<tr>
<td>functions, marker</td>
<td>564</td>
</tr>
<tr>
<td>gage connectors</td>
<td>623</td>
</tr>
<tr>
<td>gain</td>
<td>292</td>
</tr>
<tr>
<td>gain and flatness, small signal</td>
<td>666</td>
</tr>
<tr>
<td>gain compression app</td>
<td>56</td>
</tr>
<tr>
<td>gain definition</td>
<td>642</td>
</tr>
<tr>
<td>gain drift versus time, definition</td>
<td>642</td>
</tr>
<tr>
<td>gain flatness, definition</td>
<td>642</td>
</tr>
<tr>
<td>gating</td>
<td>407</td>
</tr>
<tr>
<td>reflection response</td>
<td>407</td>
</tr>
<tr>
<td>transmission response</td>
<td>407</td>
</tr>
<tr>
<td>general specifications</td>
<td></td>
</tr>
<tr>
<td>getting data with scpi</td>
<td>2505</td>
</tr>
<tr>
<td>getting started</td>
<td></td>
</tr>
<tr>
<td>global - trigger</td>
<td>330</td>
</tr>
<tr>
<td>global delta match cal</td>
<td>560</td>
</tr>
<tr>
<td>global pass-fail indicator</td>
<td>593</td>
</tr>
<tr>
<td>GPIB - configure</td>
<td>2501</td>
</tr>
<tr>
<td>GPIB Address</td>
<td>2492</td>
</tr>
<tr>
<td>GPIB Commands</td>
<td></td>
</tr>
<tr>
<td>GPIB connector</td>
<td></td>
</tr>
<tr>
<td>GPIB Examples</td>
<td></td>
</tr>
<tr>
<td><strong>GPIB Fundamentals</strong></td>
<td>2492</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------</td>
</tr>
<tr>
<td><strong>GPIB Interface</strong></td>
<td>2664</td>
</tr>
<tr>
<td><strong>GPIB Local - Remote Label</strong></td>
<td>2501</td>
</tr>
<tr>
<td><strong>GPIB Pass-Through</strong></td>
<td>2485</td>
</tr>
<tr>
<td>grid, screen</td>
<td></td>
</tr>
<tr>
<td>grounding test cables</td>
<td>634</td>
</tr>
<tr>
<td>group delay</td>
<td>674</td>
</tr>
<tr>
<td>specifications</td>
<td></td>
</tr>
<tr>
<td>group delay format</td>
<td>344</td>
</tr>
<tr>
<td>GUID</td>
<td>443</td>
</tr>
<tr>
<td>guided calibration</td>
<td>424</td>
</tr>
</tbody>
</table>

**H**

| **H08 option** | 91 |
| **H08 specifications** | |
| **H11 option** | 2739 |
| **H11 specifications** | |
| Verification    | 2715 |
| handle to an object | 1854 |
| handler io connector | 2548 |
| handshake - trigger | 334 |
| hard disk - version | 276 |
| hard disk problems - preventing | |
| hard disk recovery | 53 |
| hard disk, save file to | 598 |
| hard drive - removable | |
| hard drive defragmenter | 52 |
| harmonics specification | |

**HDD Problems**

<p>| <strong>heel strap - ESD</strong> | 634 |
| <strong>help - using</strong>     | 276 |
| <strong>hibernate</strong>        | 251 |
| <strong>high power measurements</strong> | 680 |
| <strong>hold mode</strong>        | 311 |</p>
<table>
<thead>
<tr>
<th>HPIB Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF access (H11)</td>
</tr>
<tr>
<td>IF bandwidth</td>
</tr>
<tr>
<td>IF bandwidth - for segment sweep</td>
</tr>
<tr>
<td>IF bandwidth - how to set</td>
</tr>
<tr>
<td>IF gain - phase lock- adjustment</td>
</tr>
<tr>
<td>IF gain configuration</td>
</tr>
<tr>
<td>IF Path Config (PNA-X)</td>
</tr>
<tr>
<td>IF switch configuration</td>
</tr>
<tr>
<td>imbalance parameters</td>
</tr>
<tr>
<td>impedance</td>
</tr>
<tr>
<td>setting system</td>
</tr>
<tr>
<td>Incident power - FCA scalar</td>
</tr>
<tr>
<td>input receiver - specifications</td>
</tr>
<tr>
<td>input, 10 MHz reference</td>
</tr>
<tr>
<td>inspect connectors</td>
</tr>
<tr>
<td>Installation and Quick Start Guide</td>
</tr>
<tr>
<td>installed options</td>
</tr>
<tr>
<td>instrument calibration</td>
</tr>
<tr>
<td>instrument specifications</td>
</tr>
<tr>
<td>instrument state</td>
</tr>
<tr>
<td>interface control</td>
</tr>
<tr>
<td>interface, front panel</td>
</tr>
<tr>
<td>interpolation ON/off</td>
</tr>
<tr>
<td>interpolation, effects on error correction</td>
</tr>
<tr>
<td>inverse Smith chart</td>
</tr>
<tr>
<td>IP address</td>
</tr>
<tr>
<td>isolation error</td>
</tr>
<tr>
<td>isolation, 2-port calibration</td>
</tr>
<tr>
<td>isolation, reverse</td>
</tr>
<tr>
<td>K</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>keypad not working</td>
</tr>
<tr>
<td>knob, front panel</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>L</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LAN Connections</td>
<td>54</td>
</tr>
<tr>
<td>LAN connector</td>
<td></td>
</tr>
<tr>
<td>LCD display test</td>
<td>2708</td>
</tr>
<tr>
<td>leakage, crosstalk</td>
<td>406</td>
</tr>
<tr>
<td>leveling-ALC</td>
<td>311</td>
</tr>
<tr>
<td>limit lines</td>
<td>593</td>
</tr>
<tr>
<td>limit lines, default</td>
<td>284</td>
</tr>
<tr>
<td>limit table</td>
<td>360</td>
</tr>
<tr>
<td>linear frequency sweep</td>
<td>320</td>
</tr>
<tr>
<td>linear phase shift, compensate for</td>
<td>387</td>
</tr>
<tr>
<td>linear phase; deviation from</td>
<td>659</td>
</tr>
<tr>
<td>LO - embedded in mixer</td>
<td>200</td>
</tr>
<tr>
<td>LO power adjustment</td>
<td>2709</td>
</tr>
<tr>
<td>LO source - configure for mixers</td>
<td>175</td>
</tr>
<tr>
<td>load match error</td>
<td>497</td>
</tr>
<tr>
<td>load types</td>
<td>509</td>
</tr>
<tr>
<td>local</td>
<td>2501</td>
</tr>
<tr>
<td>log errors</td>
<td>2653</td>
</tr>
<tr>
<td>log on</td>
<td>46</td>
</tr>
<tr>
<td>log sweep</td>
<td>320</td>
</tr>
<tr>
<td>logical receiver notation</td>
<td>292</td>
</tr>
<tr>
<td>long devices electrically</td>
<td>391</td>
</tr>
<tr>
<td>loops - front-panel jumpers</td>
<td></td>
</tr>
<tr>
<td>loss</td>
<td>292</td>
</tr>
<tr>
<td>lost phase lock</td>
<td>2588</td>
</tr>
<tr>
<td>low-loss 2-port devices</td>
<td>393</td>
</tr>
<tr>
<td>LRL line auto characterization</td>
<td>509</td>
</tr>
<tr>
<td>LVL</td>
<td>311</td>
</tr>
<tr>
<td>------------</td>
<td>-----</td>
</tr>
<tr>
<td>LXI</td>
<td>2501</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>M</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>macro - characterize adaptor</td>
<td>206</td>
</tr>
<tr>
<td>macro/local key</td>
<td>232</td>
</tr>
<tr>
<td>macros</td>
<td>414</td>
</tr>
<tr>
<td>magnitude offset</td>
<td>344</td>
</tr>
<tr>
<td>manage files without a mouse</td>
<td>598</td>
</tr>
<tr>
<td>map - data access</td>
<td></td>
</tr>
<tr>
<td>mapping a drive</td>
<td>612</td>
</tr>
<tr>
<td>marker</td>
<td>564</td>
</tr>
<tr>
<td>default</td>
<td>284</td>
</tr>
<tr>
<td>delta</td>
<td>564</td>
</tr>
<tr>
<td>discrete</td>
<td>564</td>
</tr>
<tr>
<td>display of</td>
<td>360</td>
</tr>
<tr>
<td>distance settings - time domain</td>
<td>216</td>
</tr>
<tr>
<td>drag</td>
<td></td>
</tr>
<tr>
<td>formats</td>
<td>564</td>
</tr>
<tr>
<td>move</td>
<td>564</td>
</tr>
<tr>
<td>readout</td>
<td>360</td>
</tr>
<tr>
<td>reference</td>
<td>564</td>
</tr>
<tr>
<td>table</td>
<td>564</td>
</tr>
<tr>
<td>marker functions</td>
<td>564</td>
</tr>
<tr>
<td>marker search</td>
<td>564</td>
</tr>
<tr>
<td>bandwidth</td>
<td>564</td>
</tr>
<tr>
<td>domain</td>
<td>564</td>
</tr>
<tr>
<td>execute</td>
<td>564</td>
</tr>
<tr>
<td>peak</td>
<td>564</td>
</tr>
<tr>
<td>target</td>
<td>564</td>
</tr>
<tr>
<td>types</td>
<td>564</td>
</tr>
<tr>
<td>marker types</td>
<td>564</td>
</tr>
<tr>
<td>fixed</td>
<td>564</td>
</tr>
<tr>
<td>normal</td>
<td>564</td>
</tr>
<tr>
<td>Term</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>markers coupled</td>
<td>564</td>
</tr>
<tr>
<td>masking</td>
<td>216</td>
</tr>
<tr>
<td>material handler interface control</td>
<td>2531</td>
</tr>
<tr>
<td>material handler io connector</td>
<td>2548</td>
</tr>
<tr>
<td>math operations, data-memory</td>
<td>574</td>
</tr>
<tr>
<td>statistics</td>
<td>574</td>
</tr>
<tr>
<td>mating plane surfaces</td>
<td>623</td>
</tr>
<tr>
<td>measure key</td>
<td>232</td>
</tr>
<tr>
<td>measurement annotation</td>
<td>360</td>
</tr>
<tr>
<td>measurement calibration</td>
<td>418</td>
</tr>
<tr>
<td>accurate</td>
<td>461</td>
</tr>
<tr>
<td>improve crosstalk</td>
<td>406</td>
</tr>
<tr>
<td>select type</td>
<td>439</td>
</tr>
<tr>
<td>validity of</td>
<td>464</td>
</tr>
<tr>
<td>measurement calibration method</td>
<td>439</td>
</tr>
<tr>
<td>measurement classes</td>
<td>300</td>
</tr>
<tr>
<td>measurement display</td>
<td>360</td>
</tr>
<tr>
<td>measurement errors</td>
<td>497</td>
</tr>
<tr>
<td>measurement parameters</td>
<td>292</td>
</tr>
<tr>
<td>arbitrary ratio</td>
<td>292</td>
</tr>
<tr>
<td>balanced</td>
<td>292</td>
</tr>
<tr>
<td>default</td>
<td>284</td>
</tr>
<tr>
<td>S-parameters</td>
<td>292</td>
</tr>
<tr>
<td>unratioed power</td>
<td>292</td>
</tr>
<tr>
<td>measurement reference plane</td>
<td>461</td>
</tr>
<tr>
<td>measurement sequence</td>
<td>265</td>
</tr>
<tr>
<td>measurement setups, preconfigured</td>
<td>353</td>
</tr>
<tr>
<td>measurement stability, increase</td>
<td>396</td>
</tr>
<tr>
<td>measurement throughput - specifications</td>
<td>410</td>
</tr>
<tr>
<td>measurement throughput techniques</td>
<td>398</td>
</tr>
<tr>
<td>measurement, averages</td>
<td>687</td>
</tr>
<tr>
<td>measurements, phase</td>
<td>398</td>
</tr>
<tr>
<td>mechanical stds calibration</td>
<td>424</td>
</tr>
<tr>
<td>Term</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>memory - random access</td>
<td>49</td>
</tr>
<tr>
<td>memory trace</td>
<td>574</td>
</tr>
<tr>
<td>menu/dialog key</td>
<td>232</td>
</tr>
<tr>
<td>messages - error</td>
<td>2653</td>
</tr>
<tr>
<td>Microsoft Networks</td>
<td>2587</td>
</tr>
<tr>
<td>Millimeter Module Configuration</td>
<td>2743</td>
</tr>
<tr>
<td>minimize application</td>
<td>360</td>
</tr>
<tr>
<td>mixed mode S-parameters</td>
<td>648</td>
</tr>
<tr>
<td>mixer calibration</td>
<td>145</td>
</tr>
<tr>
<td>mixer characterization</td>
<td>145</td>
</tr>
<tr>
<td>mixer testing</td>
<td></td>
</tr>
<tr>
<td>mixer with embedded LO</td>
<td>200</td>
</tr>
<tr>
<td>model for data-based calibration</td>
<td>461</td>
</tr>
<tr>
<td>model for polynomial calibration</td>
<td>461</td>
</tr>
<tr>
<td>modify calibration kits</td>
<td>509</td>
</tr>
<tr>
<td>monitors, external</td>
<td></td>
</tr>
<tr>
<td>mouse - manage files without</td>
<td>598</td>
</tr>
<tr>
<td>move marker</td>
<td>564</td>
</tr>
<tr>
<td>multiple measurements for throughput</td>
<td>410</td>
</tr>
<tr>
<td>multiple standards</td>
<td>424</td>
</tr>
<tr>
<td>multiport testset - E5091A</td>
<td>2751</td>
</tr>
<tr>
<td>multiport testset control</td>
<td>2755</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>N44xx testsets</td>
<td>2755</td>
</tr>
<tr>
<td>N5260A</td>
<td>2743</td>
</tr>
<tr>
<td>name of computer</td>
<td>49</td>
</tr>
<tr>
<td>NET</td>
<td>1871</td>
</tr>
<tr>
<td>network analyzer specifications</td>
<td></td>
</tr>
<tr>
<td>network analyzer users</td>
<td>46</td>
</tr>
<tr>
<td>network client</td>
<td>2587</td>
</tr>
<tr>
<td>network drive mapping</td>
<td>612</td>
</tr>
<tr>
<td>new features</td>
<td>41</td>
</tr>
<tr>
<td>new windows</td>
<td>257</td>
</tr>
</tbody>
</table>
Noise Cal
noise correlation matrix 77
Noise Figure App 77
noise floor 406
noise reduction 398
nominal incident power 135
non-insertable device calibrations 455
normal marker type 564
normalize 574
Novell NetWare 2587
number of points 385

object hierarchy 1854
obsolete COM commands
offset - magnitude 344
offset - phase 387
offset lo power adjustment 2711
offset load 509
offset-receiver atten 311
offset-source atten 311
on and off 251
on-wafer 4-port cal
open-loop leveling 311
operating system errors 2653
operating system recovery 53
operator's check 2712
option 010, time domain 216
option 014 or 015 - configurable test set
option 082 SMC 2673
option 083 2673
option 550 2755
option 551 2755
option enable 2682
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>options available</td>
<td>2673</td>
</tr>
<tr>
<td>options installed</td>
<td>2673</td>
</tr>
<tr>
<td>output power - FCA scalar</td>
<td>135</td>
</tr>
<tr>
<td>output power, absolute</td>
<td>635</td>
</tr>
<tr>
<td>output, 10 MHz reference</td>
<td>2706</td>
</tr>
<tr>
<td>overlapping sweep segments</td>
<td>320</td>
</tr>
<tr>
<td>overlay windows</td>
<td>353</td>
</tr>
<tr>
<td>override - stimulus CW</td>
<td>106</td>
</tr>
<tr>
<td>parallel interface port connector</td>
<td></td>
</tr>
<tr>
<td>parameters, amplifier</td>
<td>642</td>
</tr>
<tr>
<td>AM-to-PM conversion</td>
<td>642</td>
</tr>
<tr>
<td>complex impedance</td>
<td>642</td>
</tr>
<tr>
<td>deviation from linear phase</td>
<td>642</td>
</tr>
<tr>
<td>gain</td>
<td>642</td>
</tr>
<tr>
<td>gain compression</td>
<td>642</td>
</tr>
<tr>
<td>gain drift versus time</td>
<td>642</td>
</tr>
<tr>
<td>gain flatness</td>
<td>642</td>
</tr>
<tr>
<td>group delay</td>
<td>642</td>
</tr>
<tr>
<td>return loss</td>
<td>642</td>
</tr>
<tr>
<td>reverse isolation</td>
<td>642</td>
</tr>
<tr>
<td>parameters, measurement</td>
<td>292</td>
</tr>
<tr>
<td>arbitrary ratio</td>
<td>292</td>
</tr>
<tr>
<td>S-parameters</td>
<td>292</td>
</tr>
<tr>
<td>unratioed power</td>
<td>292</td>
</tr>
<tr>
<td>pass-fail indicator</td>
<td>593</td>
</tr>
<tr>
<td>pass-through - GPIB</td>
<td>2485</td>
</tr>
<tr>
<td>password</td>
<td>46</td>
</tr>
<tr>
<td>path configuration IF</td>
<td>2747</td>
</tr>
<tr>
<td>path configuration RF</td>
<td>357</td>
</tr>
<tr>
<td>pc - connect to</td>
<td>271</td>
</tr>
<tr>
<td>peak, marker</td>
<td>564</td>
</tr>
<tr>
<td>permission to launch and access PNA</td>
<td>1845</td>
</tr>
</tbody>
</table>
phase lock lost 2588
phase measurement accuracy 387
electrical delay 387
phase offset 387
port extensions 554
spacing between frequency points 387
phase measurements 687
types of 687
what are 687
why measure 687
phase offset 387
phase shift 387
phase, linear 659
phase-lock if gain adjustment 2718
phase-shift component 674
PIA 1871
pixel test 2708
PNA autostart 251
PNA Release 2 41
PNA specifications
PNA start 284
PNA-X 238
point spacing - x axis 320
point-averaging 398
point-in-pulse 91
points, number of 385
polar format 344
polynomial model for calibration 461
port extensions 554
power 311
absolute output 635
attenuation 311
coupling between ports 311
during retrace and sweep 311
independent segments 320
level 311
optimum 311
range 311
receiver attenuation 311
slope 311
sweep 320
unratioed 292
power - high 680
power calibration 528
power I/O connector 2563
power On and Off 251
power unleveled 311
power, line specifications
power, probe 232
power-up conditions 284
precise LO tuning 200
preferences - cal 493
preferences - error 2653
preferences - PNA 272
preset conditions 284
calibration 284
data display 284
frequency 284
global display 284
limit lines 284
marker 284
measurement parameter 284
power 284
response 284
segment 284
sweep 284
<table>
<thead>
<tr>
<th>Time Domain</th>
<th>284</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger</td>
<td>284</td>
</tr>
<tr>
<td>Preset User Defined</td>
<td>284</td>
</tr>
</tbody>
</table>

Preventing HDD problems

<table>
<thead>
<tr>
<th>Primary GPIB Address</th>
<th>2492</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Interop Assembly</td>
<td>1871</td>
</tr>
<tr>
<td>Print Help Documentation</td>
<td>276</td>
</tr>
<tr>
<td>Print Options</td>
<td>614</td>
</tr>
<tr>
<td>Print to File</td>
<td>614</td>
</tr>
<tr>
<td>Printers</td>
<td>614</td>
</tr>
<tr>
<td>Probe Power</td>
<td>232</td>
</tr>
<tr>
<td>Probe Power - PNA-X</td>
<td>238</td>
</tr>
<tr>
<td>Probe Specifications</td>
<td></td>
</tr>
<tr>
<td>Processing Data Map</td>
<td></td>
</tr>
<tr>
<td>Processor - CPU Speed</td>
<td>2673</td>
</tr>
<tr>
<td>Product Support</td>
<td>2699</td>
</tr>
<tr>
<td>Programming Guide</td>
<td>698</td>
</tr>
<tr>
<td>Protection - Antivirus</td>
<td>54</td>
</tr>
<tr>
<td>Protection - ESD</td>
<td>634</td>
</tr>
<tr>
<td>PSG Synchronize</td>
<td>662</td>
</tr>
<tr>
<td>Pulse I/O Connector</td>
<td>2561</td>
</tr>
<tr>
<td>Pulsed Application</td>
<td>91</td>
</tr>
</tbody>
</table>

**Q**

<table>
<thead>
<tr>
<th>Queues</th>
<th>2505</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quick Start Guide</td>
<td>276</td>
</tr>
<tr>
<td>Quiet Beeping</td>
<td>49</td>
</tr>
</tbody>
</table>

**R**

<table>
<thead>
<tr>
<th>RAM</th>
<th>49</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Measurement Errors</td>
<td>497</td>
</tr>
<tr>
<td>Range, Dynamic</td>
<td>381</td>
</tr>
<tr>
<td>Range, Frequency</td>
<td>2673</td>
</tr>
<tr>
<td>Range, Power</td>
<td>311</td>
</tr>
<tr>
<td>Ratio, Arbitrary</td>
<td>292</td>
</tr>
</tbody>
</table>
ratio, standing wave 693
reading data
readout marker 360
rear panel interface control 2531
rear panel specifications
rear panel tour 246
recall and sweep speeds - specifications
recall measurements quickly 410
receiver attenuation 311
receiver calibration 2734
receiver crosstalk 406
receiver display 2737
receiver input - specifications
receiver power cal 528
receivers - number of reference 2673
receivers - measurement 292
recovering the operating system 53
reduce trace noise 398
  IF bandwidth 398
  sweep averages 398
  trace smoothing 398
reference channel jumper
reference impedance for a fixture 542
reference level 344
reference level & stability - specs
reference marker 564
reference mixer 145
reference plane 461
reference position 344
reference receivers 292
reference, 10 MHz
reflection accuracy 393
reflection classes, calibration standards 509
<table>
<thead>
<tr>
<th>Reflection Measurements</th>
<th>693</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reflection Tracking Error</td>
<td>497</td>
</tr>
<tr>
<td>Relative Velocity</td>
<td>387</td>
</tr>
<tr>
<td>Remote-GPIB</td>
<td>2501</td>
</tr>
<tr>
<td>Replacement COM Commands</td>
<td></td>
</tr>
<tr>
<td>Response Calibration</td>
<td>439</td>
</tr>
<tr>
<td>Restart - Trigger</td>
<td>330</td>
</tr>
<tr>
<td>Restore ECAL Data</td>
<td>477</td>
</tr>
<tr>
<td>Restore the Operating System</td>
<td>53</td>
</tr>
<tr>
<td>Retrace - Power Level During</td>
<td>311</td>
</tr>
<tr>
<td>Return Loss</td>
<td>642</td>
</tr>
<tr>
<td>Reverse Isolation</td>
<td>690</td>
</tr>
<tr>
<td>Reverse Sweep</td>
<td>320</td>
</tr>
<tr>
<td>Revision Number - PNA</td>
<td>276</td>
</tr>
<tr>
<td>Right-Click Functions</td>
<td></td>
</tr>
<tr>
<td>Rolloff, Coupler</td>
<td>407</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S1p - S2p - S3p Files</th>
<th>598</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save and Recall a File</td>
<td>598</td>
</tr>
<tr>
<td>Save As</td>
<td>598</td>
</tr>
<tr>
<td>Save Calibration</td>
<td>424</td>
</tr>
<tr>
<td>Saving Data</td>
<td>598</td>
</tr>
<tr>
<td>Scalar Mixer Calibration</td>
<td>145</td>
</tr>
<tr>
<td>Scale</td>
<td>344</td>
</tr>
<tr>
<td>Scan Disk</td>
<td>52</td>
</tr>
<tr>
<td>Scope, Trigger</td>
<td>330</td>
</tr>
<tr>
<td>SCPI Commands</td>
<td>1873</td>
</tr>
<tr>
<td>SCPI Errors</td>
<td>2688</td>
</tr>
<tr>
<td>SCPI Examples</td>
<td></td>
</tr>
<tr>
<td>SCPI Fundamentals</td>
<td>2492</td>
</tr>
<tr>
<td>SCPI Syntax</td>
<td>2497</td>
</tr>
<tr>
<td>Screen Arrange</td>
<td>353</td>
</tr>
<tr>
<td>Screen Test</td>
<td>2708</td>
</tr>
</tbody>
</table>
screen, display
search, marker
  bandwidth
  domain
  execute
  peak
  target
  types
securing the PNA
security frequency blanking
security settings for the DCOM
segment point spacing
segment sweep
segment table
segments, limit
sequence, basic measurement
serial interface port (COM1) connector
serial number - instrument
service request
service routines
setup macros
sharing a drive
SICL
signal purity specification
Simcal
simulator - fixture
single and double click option.
single-ended devices
sliding load
slope, power
small signal gain and flatness
SMC
smith chart
smoothing - trace 398
sockets 2501
softkeys
software version 276
SOLT calibration 439
source - external synch 662
source attenuation 311
source calibration - service 2731
source match error 497
source power cal 528
source unleveled 311
source, trigger 330
  external 330
  internal 330
  manual 330
S-parameters 292
speaker 49
specifications
  corrected system performance
cycle time versus IFBW
cycle time versus number of points
data transfer time
definitions
dynamic accuracy
dynamic range
environment
front panel
front panel jumpers
general
group delay
measurement throughput
option H08
option H11
rear panel
receiver dynamic range
reference level & stability
system bandwidths
test port input (receiver)
test port output (source)
trace noise
typical cycle time
uncertainty curve equations
uncorrected system performance
speed, data transfer 413
speed, sweep 408
split windows 353
spreadsheet - saving data to 598
spurious specification
spurs - avoid 135
SRQ 2520
stability, measurement 396
device connections 396
frequency drift 396
temperature drift 396
under-sampled data 396
stack windows 353
standards, calibration 509
class assignment 420
modify 509
standard definition 420
start PNA application 284
state - instrument 598
state, trigger 330
continuous 330
group 330
Hold 330
point 330
single 330
static electricity 634
statistics 574
status bar 360
  second 2755
status registers 2520
stepped sweep 320
stimulus sweep 320
stimulus, default 284
support 2699
swap equal adapter cal 455
sweep averages 398
sweep in reverse 320
sweep indicator 320
sweep setup 320
sweep speed 408
sweep time 320
sweep types 320
SWR 292
synchronize an external source 662
synchronizing commands 2509
system bandwidths - specifications
  system impedance 379
system performance, corrected
  system performance, uncorrected
system restoration 53
system verification 2720
systematic measurement errors 497
  connector repeatability 497
  instrument noise 497
  switch repeatability 497
table mat - ESD

tables 360
  limit 360
  marker 360
  segment 360

technical specifications

technical support 2699
telnet 2501
temperature drift 396
terminate reflection measurement 393
test port specifications
  input
  output
test set (configurable) block diagram
test set block diagram
test set interface control 2531
test set IO connector 2542
test set reference switch 106
testset control - N44xx 2755
testset control - other 2755
testset control -E5091A 2751
theme - windows 54
threshold 564
throughput, increased 410
thru - unknown 455
thru response calibration 439
time and date 49
time domain 216
  aliasing 216
  default 284
  distance marker settings 216
  gating 216
  impulse and step modes 216
mask 216
overview 216
range 216
resolution 216
toolbar 360
trace coupling 216
windowing 216
timeout 2509
title - trace 257
title - window 360
title bars 360
TMSA 648
toolbars 360
active entry 360
all off 360
markers 360
measurement 360
stimulus 360
sweep control 360
time domain 360
tour
trace status 360
topology - balanced 648
torque wrench 623
touchscreen
touchstone format 598
tour 232
front panel 232
rear panel 246
trace 257
math 574
max 257
memory 574

3167
noise - reduce 398
noise specification
statistics 574
status 360
title 257
traces - channels and windows 257
transfer speed, data 413
transferring data
transmission 292
transmission classes, calibration standards 509
transmission tracking error 497
trigger 330
  auxiliary 334
  default 284
  external 334
IN / OUT connectors
model 343
scope 330
source 330
state 330
TRL calibration 489
troubleshoot the analyzer 2588
True Mode Stimulus App 648
tuning for embedded LO 200
type library 1845
types, marker 564
  fixed 564
  normal 564
uncertainty curves
uncertainty model
uncorrected system performance - specs
unknown thru cal 455
<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>unleveled</td>
<td>311</td>
</tr>
<tr>
<td>unratioed power</td>
<td>292</td>
</tr>
<tr>
<td>unwrapped phase display</td>
<td>344</td>
</tr>
<tr>
<td>upgrade options</td>
<td>2673</td>
</tr>
<tr>
<td>upgrade PNA firmware</td>
<td>2667</td>
</tr>
<tr>
<td>uploading error terms</td>
<td>2514</td>
</tr>
<tr>
<td>USB device - the PNA</td>
<td>2519</td>
</tr>
<tr>
<td>USB hub</td>
<td></td>
</tr>
<tr>
<td>USB port specifications</td>
<td></td>
</tr>
<tr>
<td>USB/GPIB Interface</td>
<td>2664</td>
</tr>
<tr>
<td>user interface</td>
<td>254</td>
</tr>
<tr>
<td>User Key</td>
<td></td>
</tr>
<tr>
<td>user name</td>
<td>46</td>
</tr>
<tr>
<td>user preset</td>
<td>284</td>
</tr>
<tr>
<td>user range</td>
<td>574</td>
</tr>
<tr>
<td>user span</td>
<td>564</td>
</tr>
<tr>
<td>user-characterization</td>
<td>477</td>
</tr>
</tbody>
</table>

**V**

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>vector mixer calibration</td>
<td>145</td>
</tr>
<tr>
<td>characterizing the cal mixer</td>
<td>145</td>
</tr>
<tr>
<td>VEE RunTime - Agilent</td>
<td>2526</td>
</tr>
<tr>
<td>velocity factor</td>
<td>387</td>
</tr>
<tr>
<td>verification - system</td>
<td>2720</td>
</tr>
<tr>
<td>verifying calibration</td>
<td>464</td>
</tr>
<tr>
<td>version - firmware</td>
<td>276</td>
</tr>
<tr>
<td>VGA output connector</td>
<td></td>
</tr>
<tr>
<td>Video Graphics Adapter</td>
<td></td>
</tr>
<tr>
<td>view/modify standards</td>
<td>509</td>
</tr>
<tr>
<td>VISA</td>
<td>2523</td>
</tr>
<tr>
<td>VMC</td>
<td>145</td>
</tr>
<tr>
<td>VMC with embedded LO</td>
<td>200</td>
</tr>
<tr>
<td>VNC</td>
<td>275</td>
</tr>
<tr>
<td>Term</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------</td>
</tr>
<tr>
<td>warm-up spec</td>
<td>2673</td>
</tr>
<tr>
<td>warranty period</td>
<td>509</td>
</tr>
<tr>
<td>waveguide</td>
<td>2514</td>
</tr>
<tr>
<td>window - calibration</td>
<td>424</td>
</tr>
<tr>
<td>window - customize</td>
<td></td>
</tr>
<tr>
<td>windowing time domain</td>
<td>216</td>
</tr>
<tr>
<td>windows - creating and managing</td>
<td>257</td>
</tr>
<tr>
<td>windows theme</td>
<td>54</td>
</tr>
<tr>
<td>windows traces and channels</td>
<td>257</td>
</tr>
<tr>
<td>Windows XP / 2000</td>
<td>54</td>
</tr>
<tr>
<td>windows, arranging feature</td>
<td>353</td>
</tr>
<tr>
<td>wizard, calibration</td>
<td>424</td>
</tr>
<tr>
<td>workgroup</td>
<td>1845</td>
</tr>
</tbody>
</table>

**X**

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-axis display for FCA</td>
<td>135</td>
</tr>
<tr>
<td>X-axis point spacing -segment sweep</td>
<td>320</td>
</tr>
<tr>
<td>XP - Windows</td>
<td>54</td>
</tr>
</tbody>
</table>

**Y**

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y-axis Scale</td>
<td>344</td>
</tr>
</tbody>
</table>

**Z**

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z conversion - balanced</td>
<td>542</td>
</tr>
<tr>
<td>Z0 System Impedance</td>
<td>379</td>
</tr>
<tr>
<td>Z5326A H08 test set commands</td>
<td>2531</td>
</tr>
<tr>
<td>zoom-trace</td>
<td></td>
</tr>
</tbody>
</table>