Agilent 4396B Network/Spectrum/Impedance Analyzer

GPIB Programming Guide

SERIAL NUMBERS

This manual applies directly to instruments with serial number prefix JP1KE.
For additional important information about serial
numbers, read “Serial Number” in Appendix A.

Agilent Technologies

Agilent Part No. 04396-90063
Printed in Japan May 2003

Sixth Edition

Notice

Agilent Technologies Japan, Ltd.
Component Test PGU-Kobe
1-3-2, Murotani, Nishi-ku, Kobe-shi,
Hyogo, 651-2241 Japan

Manual Printing History

The manual's printing date and part number indicate its current edition. The printing date changes when a new edition is printed. (Minor corrections and updates that are incorporated at reprint do not cause the date to change.) The manual part number changes when extensive technical changes are incorporated.

March 1997	First Edition (part number: 04396-90023)
July 1997	Second Edition (part number: 04396-90033)
March 1998	Third Edition (part number: 04396-90043)
March 2000	Fourth Edition (part number: 04396-90043)
November 2002	Fifth Edition (part number: 04396-90053)
May 2003	Sixth Edition (part number: 04396-90063)

Typeface Conventions

Bold	Boldface type is used when a term is defined. For example: **icons** are symbols.
Italics	Italic type is used for emphasis and for titles of manuals and other publications. Italic type is also used for keyboard entries when a name or a variable must be typed in place of the words in italics. For example: `copy` *filename* means to type the word `copy`, to type a space, and then to type the name of a file such as `file1`.
`Computer`	Computer font is used for on-screen prompts and messages.
(HARDKEYS)	Labeled keys on the instrument front panel are enclosed in ( ).
SOFTKEYS	Softkeys located to the right of the CRT are enclosed in ▒.

Graphic Symbols

General definitions of other graphic symbols used in manuals.

COMPUTER denotes information for a programmer using an external computer as the system controller.

iBASIC denotes information for a programmer using an analyzer with Instrument BASIC as the system controller.

How to Use This Manual

This manual provides an introduction to writing BASIC programs for the 4396B Network/Spectrum Analyzer (analyzer). To reduce the time required for you to learn how to write programs for the analyzer, the examples shown in this guide are supplied on sample disks. You can perform each example sequentially or you can select the examples that apply to your immediate needs and learn those techniques. Use the table of contents and the index to quickly locate these examples.

Also, depending upon your experience in writing BASIC programs using GPIB commands, you may want to do one of the following:

1. If you are an experienced programmer and have programmed GPIB systems before, you can scan the examples in this guide to find out how the analyzer can be used in your system. If you have never programmed an instrument similar to the analyzer, you can start at the beginning and do the examples that apply to your application.
2. If you are an experienced programmer, but do not have any knowledge of GPIB commands, review some examples to decide where you need help. See the *GPIB Command Reference* for additional information on GPIB commands.
3. If you are not an experienced programmer and you do not have any knowledge of GPIB commands, see the *GPIB Command Reference* for a list of the documentation that you will need to review before using this guide.

The analyzer can also use Instrument BASIC. Documentation for Instrument BASIC and the other manuals available for the analyzer is listed in the Documentation Map on the next page.

Documentation Map

The following manuals are available for the analyzer.

User's Guide (Agilent Part Number 04396-900x1 [1])

The User's Guide walks you through system setup and initial power-on, shows how to make basic measurements, explains commonly used features, and typical application measurement examples. After you receive your analyzer, begin with this manual.

Task Reference (Agilent Part Number 04396-900x0 [1])

Task Reference helps you to learn how to use the analyzer. This manual provides simple step-by-step instructions without concepts.

Function Reference (Agilent Part Number 04396-900x2 [1])

The Function Reference describes all function accessed from the front panel keys and softkeys. It also provides information on options and accessories available, specifications, system performance, and some topics about the analyzer's features.

Programming Guide (Agilent Part Number 04396-900x3 [1])

The Programming Guide shows how to write and use BASIC program to control the analyzer and describes how Instrument BASIC works with the analyzer..

GPIB Command Reference (Agilent Part Number 04396-900x4 [1])

The GPIB Command Reference provides a summary of all available GPIB commands. It also provides information on the status reporting structure and the trigger system (these features conform to the SCPI standard).

Option 010 Operating Handbook (Agilent Part Number 04396-900x6 [1])

The option 010 Operation Handbook describes the unique impedance measurement functions of the 4396B with option 010.

Instrument BASIC Manual Set (Agilent Part Number 04155-90151(E2083-90000))

The Instrument BASIC User's Handbook introduces you to the Instrument BASIC programming language, provide some helpful hints on getting the most use from it, and provide a general programming reference. It is divided into three books, *Instrument BASIC Programming Techniques*, *Instrument BASIC Interface Techniques*, and *Instrument BASIC Language Reference*.

Performance Test Manual (Agilent Part Number 04396-901x0 [1])

The Performance Test Manual explains how to verify conformance to published specifications.

Service Manual (Agilent Part Number 04396-901x1 [1])

The Service Manual explains how to adjust, troubleshoot, and repair the instrument. This manual is option 0BW only.

1 The number indicated by "x" in the part number of each manual, is allocated for numbers increased by one each time a revision is made. The latest edition comes with the product.

Contents

1. Learning GPIB Remote Control Basics

Required Equipment 1-1
To Prepare for GPIB Control 1-2
GPIB Commands Introduction 1-4
To Execute an GPIB Command 1-4
To Program a Basic Measurement 1-5
  Set I/O Path 1-6
  Set Up the Measurement Parameters 1-6
  Perform Calibration 1-7
  Connect DUT 1-7
  Trigger a Measurement 1-7
  Post-Processing 1-8
  Transfer Data 1-8
To Execute an GPIB Command with a Parameter 1-8
To Execute a Query 1-8

2. Triggering the Analyzer from Remote

To Measure Continuously 2-2
  Set Trigger Source 2-2
  Start Continuous Measurement Sweep 2-2
To Trigger a Measurement From the Controller 2-3
  Set Trigger Source 2-3
  Trigger a Measurement 2-3
  Set Trigger Source 2-4
  Trigger a Measurement 2-4

3. Synchronizing the Analyzer from Remote

To Wait For the Preceding Operation to Complete 3-2
  Let Controller Wait For Operation to Complete (OPC) 3-2
To Wait for Sweep End 3-3
  Enable Sweep-End Bit 3-3
  Enable SRQ Interrupt 3-4
  Wait Until Measurement Is Done 3-4
  Generate SRQ 3-4
To Report Command Error Occurrence 3-5
  Enable Error Bit 3-5
  Report Command Error 3-6
  Output Error 3-6
  Return to Execute GPIB command 3-7

4. Reading Measurement Data
To Read Data Using the Marker Search Function 4-2
Search Maximum Value 4-2
Read Data 4-3
To Get Measurement Trace Using ASCII Format 4-4
Set the Receive Array 4-4
Set Data Transfer Format 4-5
Read Data 4-5
To Get Measurement Trace Using Binary Format 4-6
Set the Receive Array 4-7
Set Data Transfer Format 4-7
Read Data 4-8
Set the Receive Array 4-9
Set Data Transfer Format 4-10
Read Data 4-10

5. Writing Data Arrays to the Analyzer
To Modify Calibration Data 5-1
Read Calibration Data 5-2
Modify Calibration Data 5-3
Restore Modified Calibration Data 5-3
Redrawing Measurement Trace with Modified Calibration Data 5-5
To Modify Error-Corrected Data 5-6
Read Error-Corrected Data 5-7
Restore Modified Error-Corrected Data 5-7
To Modify Trace Data 5-8
Read Trace Data 5-9
Restore Modified Trace Data 5-9

6. Printing or Plotting the Analyzer's Display
To Print Analyzer Display 6-1
Printer Preparation 6-1
Execute Print 6-1
To Observe Printing 6-1

7. Controlling Instrument BASIC from Remote
To Control GPIB from Instrument BASIC 7-1
To Execute an Instrument BASIC Command from the External Controller 7-3
To Run an Instrument BASIC Program From the External Controller 7-4
Open the Instrument BASIC Editor 7-4
Send the Instrument BASIC Program 7-4
Close the Instrument BASIC Editor 7-4
Run the Instrument BASIC Program 7-5
To Transfer the Program to Instrument BASIC 7-5
Open the Instrument BASIC Editor 7-5
Transfer the Program 7-6
Close the Instrument BASIC Editor 7-6
To Load an Array in an Instrument BASIC Program to the External Controller . . 7-6
Transfer the Program Array of Instrument BASIC 7-6

8. Programming Limit Test from Remote
To Perform limit Test 8-1
Edit Limit Line 8-2
Set Limit Line 8-3
Read the Limit Test Results 8-4

9. Using the List Sweep Function
To Set List Sweep 9-1
Edit List Table 9-2
Set List Table 9-3

10. Using the Analyzer's I/O Port
To Synchronize External Handler with Analyzer 10-2
Send Signal to the External Handler 10-2
Read Signal from the External Handler 10-3

11. Using Application Programs
Total Harmonic Distortion 11-1
Adjacent Channel Power Calculation 11-4
Occupied Power Bandwidth Calculation 11-6
File Transfer Function 11-9
File Transfer from 4396B to External Controller 11-10
File Transfer from External Controller to 4396B 11-12
Displaying List of Files in Current Directory 11-15

12. If You Have a Problem
If There Is No Response From an Instrument on the GPIB Bus 12-1
If an Error Message is Displayed 12-1
If the Disk Cannot Be Read 12-2
If the GPIB Command Does Not Work 12-3

A. Manual Changes
Introduction A-1
Manual Changes A-1
Serial Number A-2
Change 1 . . . . . A-3

Index

Figures

1-1. System Configuration for GPIB Remote Control 1-2
1-2. Program Flow 1-5
1-3. Sample Program : Basic Measurement (1/2) 1-5
2-1. Trigger System 2-1
2-2. Sample Program : To Trigger Measurements Continuously 2-2
2-3. Sample Program : To Trigger a Measurement from Controller (1) 2-3
2-4. Sample Program : To Trigger a Measurement from Controller (2) 2-4
3-1. SRQ Generation 3-1
3-2. Sample Program: To Wait for the Preceding Operation to Complete 3-2
3-3. Sample Program : To Wait for Sweep End 3-3
3-4. Sweep-End Bit Enabling 3-3
3-5. Sample Program : To Report Command Error Occurrence 3-5
3-6. Command-Error Bit Enabling 3-6
4-1. Sample Program : To Read Data Using Marker Search Function 4-2
4-2. Sample Program : To Get Measurement Trace Using ASCII Format 4-4
4-3. Sample Program : To Get Measurement Trace Using IEEE 64-bit Floating Point Format (For External Controller) (1/2) 4-6
4-4. FORM3 Data Transfer Format 4-8
4-5. Sample Program : To Get Measurement Trace Using IEEE 64-bit Floating Point Format (For Instrument BASIC) 4-9
5-1. Sample Program : To Modify Calibration Data (1/2) 5-1
5-2. Data Arrays, Data Processing, and GPIB Command 5-5
5-3. Sample Program : To Modify Error-Corrected Data 5-6
5-4. Sample Program : To Modify Trace Data 5-8
6-1. Sample Program : To Observe Printing 6-2
7-1. Pass Control 7-1
7-2. Sample Program: To Receive Control (On Instrument BASIC) 7-2
7-3. Sample Program : To Run the Instrument BASIC Program From the External Controller (On External Controller) 7-4
7-4. Sample Program : To Transfer the Program to Instrument BASIC (on External Controller) 7-5
7-5. Sample Program : To Load Instrument BASIC Program Array (on External Controller) 7-6
8-1. Sample Program : Limit Test (1/2) 8-1
9-1. Sample Program : List Sweep (1/2) 9-1
10-1. I/O Port 10-1
10-2. Sample Program : Synchronization of an External Handler with the Analyzer . 10-2
10-3. 8-Bit Data of OUT0-7 10-2
10-4. Sending Signal to an the External Handler 10-2
10-5. Reading Signal from the External Handler 10-3
11-1. Harmonic Distortion in a Signal 11-1
11-2. Sample Program : Total Harmonic Distortion (THD) (1/2) 11-2
11-3. Adjacent Channel Power 11-4
11-4. Sample Program : Adjacent Channel Power Calculation (1/2) 11-4
11-5. 99 % Occupied Power Bandwidth 11-6
11-6. Sample Program : Occupied Power Bandwidth Calculation (1/2) 11-7

11-7. Sample Program: File Transfer from 4396B to External Controller (1/2) 11-10
11-8. Fixed length block format 11-12
11-9. Sample Program: File Transfer from External Controller to 4396B (1/2) 11-13
11-10. Sample Program: Displaying List of Files in Current Directory of 4396B 11-15
A-1. Serial Number Plate A-2

Tables

A-1. Manual Changes by Serial Number A-1
A-2. Manual Changes by Firmware Version A-1

1

Learning GPIB Remote Control Basics

This chapter provides information on how to configure the GPIB remote-control system and the basic use of the GPIB commands. In the examples used in this manual, most of the commands are the simple GPIB commands. For each of these commands, there is also a corresponding command that conforms to the Standard Commands for Programmable Instruments (SCPI) standard. For additional information of about all commands, see the *GPIB Command Reference* manual.

What is GPIB?

The General Purpose Interface Bus (GPIB) is used for remote control of the 4396B Network/Spectrum/Impedance Analyzer (analyzer). GPIB is a standard for interfacing instruments to computers and peripherals. This standard supports worldwide standards IEEE 488.1, IEC-625, and IEEE 488.2. The GPIB interface allows the analyzer to be controlled by an external computer. The computer sends commands or instructions to and receives data from the instrument through the GPIB.

Required Equipment

To perform the examples in this manual, you need the following equipment:

1. The analyzer and the accessories required to test a specific device under test (DUT).
2. For the GPIB system controller,

iBASIC

If the analyzer has the Instrument BASIC installed, it can be used as the system controller.

Or,

COMPUTER

An HP 9000 Series 200 or 300 computer or an HP Vectra PC with a measurement coprocessor or card (82300 or 82324). The computer must have enough memory to hold BASIC, needed binaries, and at least 64 kilobytes of program space.

BASIC 3.0 or higher operating system and the following binary extensions:

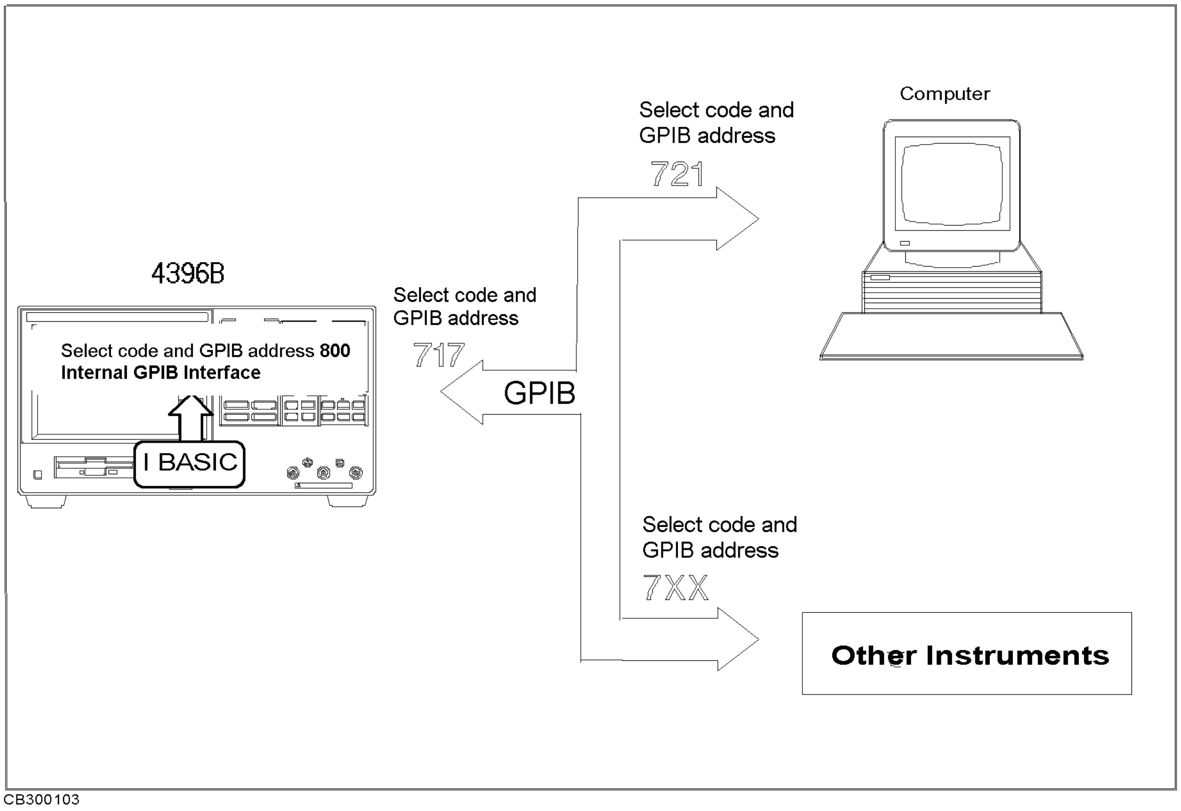
HPIB, GRAPH, IO, KBD, and ERR

A disk drive is required to load BASIC, if no internal disk drive is available. (Depending on the disk drive, a binary such as CS80 may be required.)

3. Peripherals (printer, plotter, and so on) and any GPIB instruments that are required for your application.
4. 10833A/B/C/D GPIB cables to interconnect the computer, the analyzer, and any peripherals.

To Prepare for GPIB Control

1. Connect the analyzer and controller, plus any other instruments and peripherals with GPIB cables.



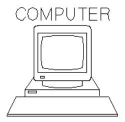
Computer
Select code and GPIB address
721
4396B
Select code and GPIB address
Select code and GPIB address **800**
Internal GPIB Interface
717
GPIB
I BASIC
Select code and GPIB address
7XX
Other Instruments
CB300103

Figure 1-1. System Configuration for GPIB Remote Control

* To set printer or plotter see Chapter 6.

2. Turn on the analyzer.
3. Prepare the system controller.

If you are using only Instrument BASIC and no external controller, prepare the analyzer for your use. For details, see *Using Instrument BASIC with the 4396B*.



If you are using a computer as an external controller,

a. Set the analyzer to addressable only mode.

Press (Local) `ADDRESSABLE ONLY` .

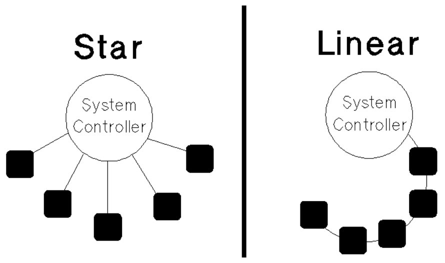
b. Set GPIB address of the analyzer to 17.

Press (Local) `SET ADDRESS` `ADDRESS: 4396` (1) (7) (x1).

c. Turn on the controller. Then load the BASIC operating system and the binary extensions.

How large a system can you configure?

- A maximum of 15 devices can be connected on one bus system.
- The length of cable between one device and another must be less than or equal to four meters. The total length of cable in one bus system must be less than or equal to two meters times the number of devices connected on the bus (the GPIB controller counts as one device). The total length of cable must not exceed 20 meters.
- Star, linear, and combinational cable configurations are allowed. There must be no loop.



- It is recommended that no more than four piggyback connectors be stacked together on one device. Otherwise, the resulting structure could exert enough force on the connector mounting to damage it.

GPIB Commands Introduction

All the analyzer's front-panel keys have a corresponding GPIB command. By executing an GPIB command, you can operate the analyzer as if you were pressing the corresponding key.

For example,

Pressing (Preset) is the same as executing the GPIB command, `PRES`.

Note Each of the analyzer's functions has two corresponding GPIB commands: One is unique to analyzer, and the other corresponds to the SCPI (Standard Commands for Programmable Instruments) standard. In this guide, only the commands that are unique to the analyzer are described.

For example, you can use either `NA` (which is unique to the analyzer) or `INST:TYPE NA` (which conforms to SCPI) to select the analyzer type.

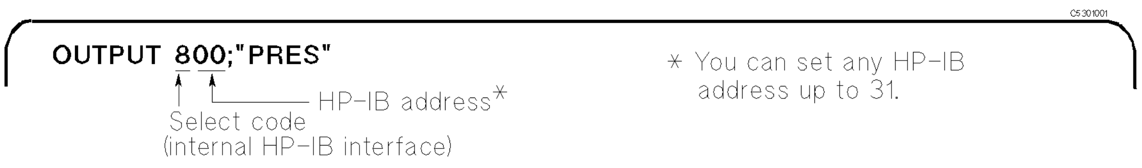
For details on SCPI, see *GPIB Command Reference*.

To Execute an GPIB Command

Combine the BASIC `OUTPUT` statement with the GPIB select code, the device address, and finally the analyzer command. For example, to execute `PRES` command, type:

iBASIC

Using Instrument BASIC



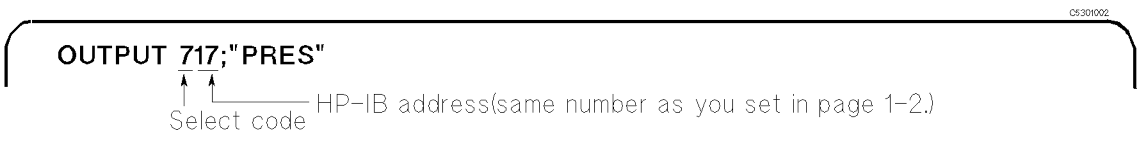
```
OUTPUT 800;"PRES"
```

C5301001
HP-IB address*
Select code
(internal HP-IB interface)
* You can set any HP-IB address up to 31.

And press (Return). The analyzer goes to the preset state.

COMPUTER

Using an External Controller



```
OUTPUT 717;"PRES"
```

And press (Return). The analyzer is set to GPIB remote mode. Then the analyzer goes to the preset state.

What is GPIB remote mode?

COMPUTER

Executing an `OUTPUT` statement that is addressed to the analyzer, sets it to the GPIB remote mode. In the remote mode, all the analyzer's front-panel keys are locked out, except (Local). Pressing (Local) puts the analyzer back in local mode. In local mode, all front-panel keys are enabled.

To Program a Basic Measurement

This section describes how to organize the commands into a measurement sequence. Figure 1-2 shows a typical program flow for a measurement.

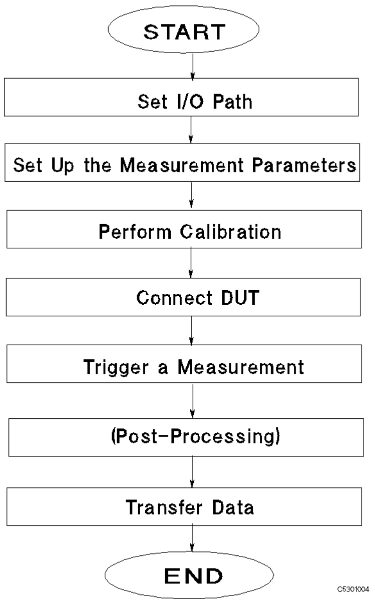


Figure 1-2. Program Flow

The following program performs the measurement flow controlling the analyzer using GPIB.

iBASIC

This guide shows program lists of sample programs for an external controller. To use the sample programs in this guide with Instrument BASIC, change the select code from 7 to 8 and change the GPIB address from 17 to 00 (that is, use 800 instead of 717).

```
10    !
20    ! Figure 1-3. Basic Measurement
30    !
40    ASSIGN @Hp4396 TO 717 ! When iBASIC is used, change "717" to "800".
50    !
60    OUTPUT @Hp4396;"PRES" ! Preset 4396B
70    OUTPUT @Hp4396;"CHAN1;NA;MEAS S21;FMT LOGM"
80    INPUT "Enter center frequency (Hz).",F_cent
90    INPUT "Enter frequency span (Hz).",F_span
100   OUTPUT @Hp4396;"CENT ";F_cent
110   OUTPUT @Hp4396;"SPAN ";F_span
120   !
130   ! Frequency Response Calibration
140   OUTPUT @Hp4396;"CALK N50"       ! Select 50 ohm type-N Cal. kit
150   OUTPUT @Hp4396;"CALI RESP"      ! Select Response cal.
160   OUTPUT @Hp4396;"CLES"           ! Clear all status
170   INPUT "Connect THRU, then press [Return].",Dum$
180   OUTPUT @Hp4396;"*SRE 4;ESNB 1" ! Set enable STB and ESB
190   ON INTR 7 GOTO Cal_end      ! \ When iBASIC is used, change "7" to "8".
200   ENABLE INTR 7;2             ! /
210     OUTPUT @Hp4396;"STANC"        ! Measure THRU
```

Figure 1-3. Sample Program : Basic Measurement (1/2)

```
220 Calibrating:    GOTO Calibrating
230 Cal_end:    !
240   OUTPUT @Hp4396;"RESPDONE"      ! Calculating cal coefficients
250   OUTPUT @Hp4396;"*OPC?"      ! \ Waiting calculation end
260   ENTER @Hp4396;Dum           ! /
270   DISP "Response cal completed."
280   !
290   ! Measurement
300   INPUT "Connect DUT, then press [Return].",Dum$
310   OUTPUT @Hp4396;"CLES"   ! Clear all status registers
320   OUTPUT @Hp4396;"*SRE 4;ESNB 1"
330   ON INTR 7 GOTO Sweep_end    ! \ When iBASIC is used,
340   ENABLE INTR 7;2             ! / change "7" to "8"
350     OUTPUT @Hp4396;"SING"   ! Sweep mode is SINGLE
360 Measuring:    GOTO Measuring
370 Sweep_end:   !
380   OUTPUT @Hp4396;"MKR ON"       ! Marker 1 ON
390   OUTPUT @Hp4396;"SEAM MAX"     ! Search MAX
400   OUTPUT @Hp4396;"OUTPMKR?"     ! Output marker value
410   ENTER @Hp4396;Val1,Val2,Swp
420   PRINT "Max val:",Val1;"dB"
430   PRINT "Swp.Prmtr:",Swp;"Hz"
440   END
```

Figure 1-3. Sample Program : Basic Measurement (2/2)

Set I/O Path

```
40    ASSIGN @Hp4396 TO 717 ! When iBASIC is used, change "717" to "800".
```

This operation allows you to use @Hp4396, instead of 717(or 800), as the GPIB address in the program.

Set Up the Measurement Parameters

```
60    OUTPUT @Hp4396;"PRES" ! Preset 4396B
70    OUTPUT @Hp4396;"CHAN1;NA;MEAS S21;FMT LOGM"
80    INPUT "Enter center frequency (Hz).",F_cent
90    INPUT "Enter frequency span (Hz).",F_span
100   OUTPUT @Hp4396;"CENT ";F_cent
110   OUTPUT @Hp4396;"SPAN ";F_span
```

You can execute GPIB commands in the same sequence as key operation. Lines 60 and 70 perform the same operation as pressing (Preset) (Chan 1) (Meas) ANALYZER TYPE NETWORK ANALYZER S PARAMETERS Trans:FDW S21[B/R] (Format) LOG MAG.

In general, the procedure for setting up measurements on the analyzer via GPIB follows the same sequence as performing the procedure manually. There is no required order, as long as the desired frequency range, number of points, and power level are set before performing the calibration.

In line 70, several GPIB commands, separated by semicolon, are executed in a line. This is the same as:

```
70    OUTPUT @Hp4396;"CHAN1"
71    OUTPUT @Hp4396;"NA"
72    OUTPUT @Hp4396;"MEAS S21"
73    OUTPUT @Hp4396;"FMT LOGM"
```

In lines 80 to 110 (setting frequency), parameters are required with the GPIB command. To set parameters, see "To Execute an GPIB Command with a Parameter" later in this chapter.

Perform Calibration

```
130   ! Frequency Response Calibration
140   OUTPUT @Hp4396;"CALK N50"       ! Select 50 ohm type-N Cal. kit
150   OUTPUT @Hp4396;"CALI RESP"      ! Select Response cal.
160   OUTPUT @Hp4396;"CLES"           ! Clear all status
170   INPUT "Connect THRU, then press [Return].",Dum$
180   OUTPUT @Hp4396;"*SRE 4;ESNB 1" ! Set enable STB and ESB
190   ON INTR 7 GOTO Cal_end     ! \ When iBASIC is used, change "7" to "8".
200   ENABLE INTR 7;2            ! /
210     OUTPUT @Hp4396;"STANC"         ! Measure THRU
220 Calibrating:    GOTO Calibrating
230 Cal_end:    !
240   OUTPUT @Hp4396;"RESPDONE"      ! Calculating cal coefficients
250   OUTPUT @Hp4396;"*OPC?"      ! \ Waiting calculation end
260   ENTER @Hp4396;Dum           ! /
270   DISP "Response cal completed."
```

In lines 140 to 240, the GPIB program follows the key strokes required to calibrate from the front panel. This program performs a response calibration.

Line 170 requests the operator to connect a THRU calibration standard.

Lines 180 through 220 use the status bytes to detect the completion of the THRU calibration. See "To Wait for Sweep End" in Chapter 3.

Lines 240 through 270 use the *OPC? command to detect the completion of the calculation of the calibration coefficients. See "To Wait For the Preceding Operation to Complete" in Chapter 3.

Connect DUT

```
300   INPUT "Connect DUT, then press [Return].",Dum$
```

Line 300 requests the operator to connect a DUT to the analyzer.

All instrument settings and calibration are done. You can now measure the DUT.

Trigger a Measurement

```
310   OUTPUT @Hp4396;"CLES"   ! Clear all status registers
320   OUTPUT @Hp4396;"*SRE 4;ESNB 1"
330   ON INTR 7 GOTO Sweep_end    ! \ When iBASIC is used,
340   ENABLE INTR 7;2             ! / change "7" to "8"
350     OUTPUT @Hp4396;"SING"   ! Sweep mode is SINGLE
360 Measuring:    GOTO Measuring
370 Sweep_end:   !
```

Lines 310 to 370 enable SRQ interruption for sweep end detection. For details, see Chapter 3.

In line 350, the analyzer executed a single trigger. For more advanced trigger control, see Chapter 2.

Post-Processing

```
380   OUTPUT @Hp4396;"MKR ON"       ! Marker 1 ON
390   OUTPUT @Hp4396;"SEAM MAX"      ! Search MAX
```

Line 380 activates the marker and line 390 moves the marker to the maximum value on the trace. For details on using the marker, see Chapter 4.

Transfer Data

```
400   OUTPUT @Hp4396;"OUTPMKR?"  ! Output marker value
410   ENTER @Hp4396;Val1,Val2,Swp
```

Line 400 the measured data is transferred to the controller. For details about data transfer, see Chapter 4.

To Execute an GPIB Command with a Parameter

Some GPIB commands require a numeric parameter. For example:

```
OUTPUT @Hp4396;"CENT 25000000"   ! Set center frequency to 25 MHz.
```

(The space between the command and the numeric parameter is mandatory.)

You can program it to be entered each time the program is run. For example:

```
100 INPUT "Enter center frequency(Hz).";F_cent
110 OUTPUT @Hp4396;"CENT ";F_cent
```

Executing this,

```
Enter center frequency (Hz).
25000000
```

The analyzer's center frequency is set to 25 MHz.

To Execute a Query

Any GPIB command that is used with a numeric parameter can also be used as query command. For example, the `CENT` *numeric_parameter* command used in the previous example, can be combined with a `?`, and used as a query command as follows,

```
10 OUTPUT @Hp4396;"CENT?"
20 ENTER @Hp4396;A
30 PRINT A
```

The `CENT?` command returns the current center frequency, which is put into `A`. Executing this program results in the following:

```
25000000
```

By interrogating the analyzer to determine the values of the start and stop frequencies, or the center frequency and frequency span, the computer can keep track of the actual frequencies.

2

Triggering the Analyzer from Remote

This chapter describes how to control the trigger system of the analyzer.

To trigger a measurement from a controller, the following steps are commonly used:

1. Set the trigger source to:

   Bus, or Internal (free run)

   (In External, Video, Manual or Gate trigger, you cannot trigger from the controller, so these sources are not mentioned in this guide.)

2. Set the number of measurements and the analyzer is initiated. You can set the number of measurements as:

   (Hold)—Single—Number of Group—Continuous

3. Generate the trigger event and the analyzer starts a measurement.

The analyzer trigger system has three states: Idle, Waiting for Trigger, and Measurement.

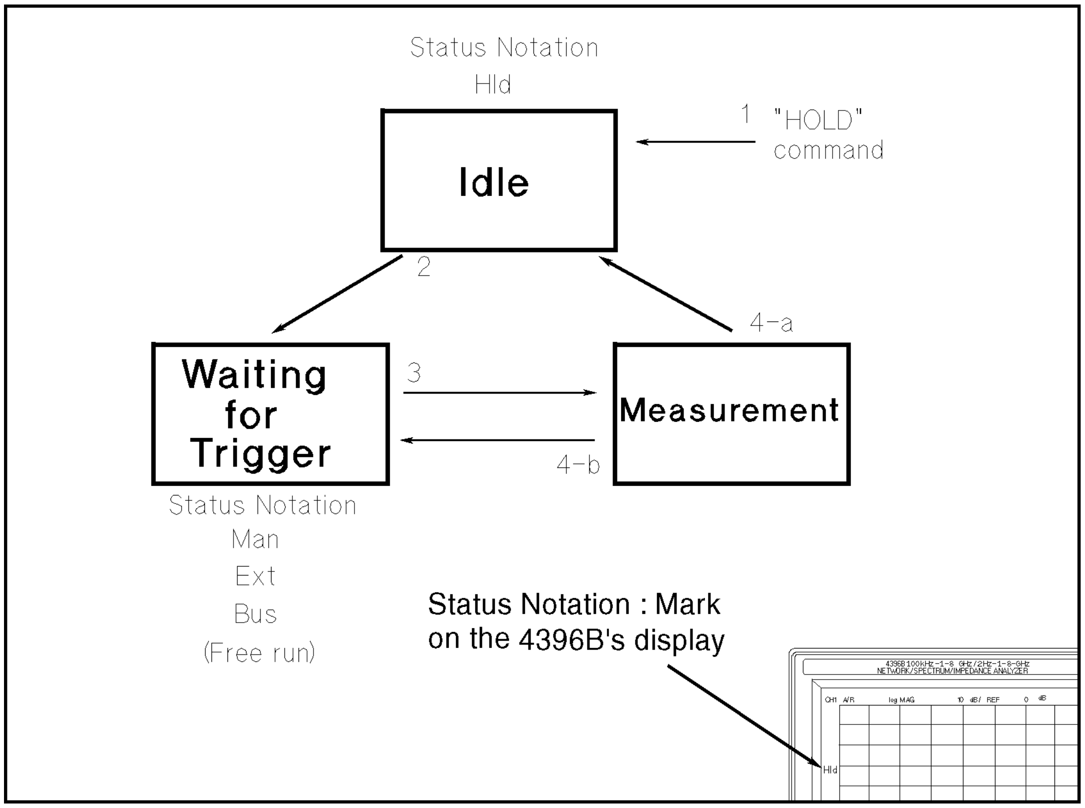


Figure 2-1. Trigger System

In Figure 2-1,

1. After a HOLD GPIB command execution, the analyzer returns to the "Idle" state.
2. By setting the number of measurements, the analyzer changes from the "Idle" state to the "Waiting for Trigger" state.
3. At the "Waiting for Trigger" state, a trigger input (corresponding to the trigger source) starts a measurement.

Bus	GPIB command *TRG or BASIC command TRIGGER triggers measurements.
Internal (free run)	There is no need for a trigger input. The analyzer starts the measurements immediately.

4. After the measurement is complete, the next state depends on the number of measurements.

Single	goes to the "Idle" state(4-a).
Number of Groups	Goes to the "Waiting for Trigger" state until the number of groups not measured yet equals zero(4-b). After all measurements are completed, goes to "Idle" state(4-a).
Continuous	goes to the "Waiting for Trigger" state(4-b).

To Measure Continuously

```
10 !
20 ! Figure 2-2. To Trigger Measurement Continuously
30 !
40 ASSIGN @Hp4396 TO 717  ! When iBASIC is used, change "717" to "800"
50 OUTPUT @Hp4396;"TRGS INT"
60 OUTPUT @Hp4396;"CONT"
70 END
```

Figure 2-2. Sample Program : To Trigger Measurements Continuously

Set Trigger Source

```
50 OUTPUT @Hp4396;"TRGS INT"
```

Set the trigger source to internal.

Start Continuous Measurement Sweep

```
60 OUTPUT @Hp4396;"CONT"
```

The analyzer changes to the "Waiting for Trigger" state. In this program, the internal trigger source is selected and the analyzer immediately starts continuous measurements.

What can you do to abort a measurement?

Send the command:

```
OUTPUT @Hp4396;"HOLD"
```

The measurement sweep is aborted.

What are other trigger commands?

Instead of `CONT`, you can use,

`OUTPUT @Hp4396;"SING"`	*for single measurement*
`OUTPUT @Hp4396;"NUMG` *parameter*`"`	*for number of group measurements*

When you transfer measurement data to the controller, you must use either the `SING` or the `NUMG` *parameter* command to synchronize the controller and the analyzer. To use these commands, see the "To Trigger a Measurement From the Controller" example.

To Trigger a Measurement From the Controller

Two methods of triggering a measurement from the controller are shown in Figure 2-3 and Figure 2-4.

```
10 !
20 ! Figure 2-3. To Trigger Measurement From Controller(1)
30 !
40 ASSIGN @Hp4396 TO 717 ! When iBASIC is used, change "717" to "800"
50 OUTPUT @Hp4396;"TRGS INT"
60 OUTPUT @Hp4396;"SING"
70 END
```

Figure 2-3. Sample Program : To Trigger a Measurement from Controller (1)

Set Trigger Source

```
50 OUTPUT @Hp4396;"TRGS INT"
```

Set the trigger source to internal.

Trigger a Measurement

```
60 OUTPUT @Hp4396;"SING"
```

The analyzer changes to the "Waiting for Trigger" state. In this program, the internal source is selected and the analyzer immediately starts a measurement. After the measurement, the analyzer goes to the "Idle" state.

How can you perform averaging?

When you set the averaging on, you must also set the number of measurements to the same value as the averaging factor. For example, if the averaging factor is 10, replace line 60 as follows:

```
60 OUTPUT @Hp4396;"NUMG 10"
```

How can you wait for a measurement to be completed?

When you want to return the measurement data to the controller, you must wait for the measurement to be completed. For details, see Chapter 3.

```
10 !
20 ! Figure 2-4. To Trigger Measurement From the Controller(2)
30 !
40 ASSIGN @Hp4396 TO 717 ! When iBASIC is used, change "717" to "800"
50 OUTPUT @Hp4396;"TRGS BUS"
60 OUTPUT @Hp4396;"CONT"
70 OUTPUT @Hp4396;"*TRG"
80 END
```

Figure 2-4. Sample Program : To Trigger a Measurement from Controller (2)

Set Trigger Source

```
50 OUTPUT @Hp4396;"TRGS BUS"
```

Set the trigger source to bus.

Trigger a Measurement

```
70 OUTPUT @Hp4396;"*TRG"
```

Triggers the analyzer. When the trigger source is set to bus, you can use the group execution trigger as follows:

```
70 TRIGGER 7
```

What is Group Execution Trigger (GET)?

COMPUTER

The BASIC command TRIGGER can be used instead of the *TRG command. The BASIC command is used to trigger all triggerable instruments on a BUS at the same time. Therefore, to trigger all triggerable instruments on select code 7(GPIB bus) execute the command:

```
TRIGGER 7
```

Synchronizing the Analyzer from Remote

GPIB analyzer control programs that can be used for guiding you through an analyzer calibration, and for reading and manipulating measurement data.

The control program must wait until the calculation data is processed before continuing with the next instruction. Also, the control program must wait until the measurement is completed before it reads the measurement data.

This chapter describes two techniques used by the control programs to synchronize the controller and the analyzer.

- Uses the `*OPC?` command.

  This command halts the execution of the program until the analyzer completes the preceding commands in the program.

- Reports the analyzer's status and generates SRQ.

  The analyzer has a status reporting mechanism that gives information about specific functions and events inside the analyzer. The status byte is an 8-bit register with each bit summarizing the state of one aspect of the analyzer. For example, the error queue summary bit is set if there are any errors in the queue. For the status byte register bit assignment, see the *GPIB Command Reference* manual.

What is an SRQ?

An SRQ (Service Request) is an interrupt generated by the analyzer. The analyzer can be setup to sent an SRQ when it needs the attention of the controller. The controller can ignore the SRQ or it can be setup to interrupt the program using the ON INTR commands. The Status Byte can be used to define the specific event that generates an SRQ (for example, the end of sweep complete).

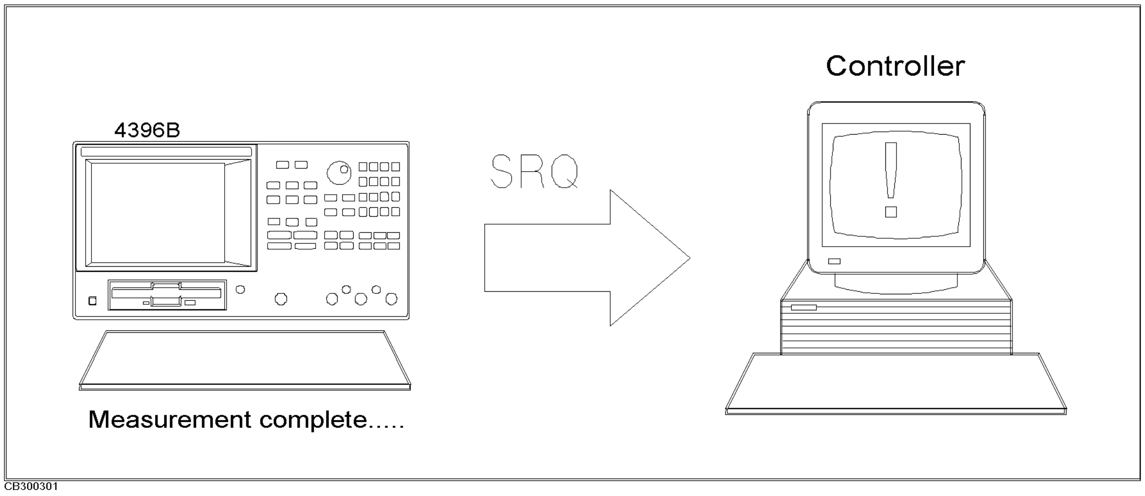


Figure 3-1. SRQ Generation

To Wait For the Preceding Operation to Complete

```
10 !
20 ! Figure 3-2. To Wait for the Preceding Operation Complete
30 !
40   ASSIGN @Hp4396 TO 717     ! When iBASIC is used, change "717" to "800"
50   !
60   !  OUTPUT statement to send GPIB command
70   !
80   OUTPUT @Hp4396;"*OPC?"
90   ENTER @Hp4396;A
100  !
110  !  Next operation
120  !
130  END
```

Figure 3-2. Sample Program: To Wait for the Preceding Operation to Complete

Let Controller Wait For Operation to Complete (OPC)

```
80   OUTPUT @Hp4396;"*OPC?"
90   ENTER @Hp4396;A
```

In line 80, the `*OPC?` command waits for the preceding operations to complete and then returns a `1`.

In line 90, the controller pauses the program until the analyzer returns a `1`.

For example, in the sample program in Figure 1-3 (Chapter 1), the `*OPC?` command is used as follows:

```
 :
240   OUTPUT @Hp4396;"RESPDONE"      ! Calculating cal coefficients
250   OUTPUT @Hp4396;"*OPC?"      ! \ Waiting calculation end
260   ENTER @Hp4396;Dum           ! /
270   DISP "Response cal completed."
 :
```

You cannot use `*OPC?` for the functions listed under SRQ (at the beginning of the chapter). Use the status byte for these functions.

To Wait for Sweep End

```
10  !
20  ! Figure 3-3. To Wait for Sweep End
30  !
40   ASSIGN @Hp4396 TO 717  ! When iBASIC is used, change "717" to "800"
50   OUTPUT @Hp4396;"TRGS INT"
60   OUTPUT @Hp4396;"CLES"
70   OUTPUT @Hp4396;"*SRE 4;ESNB 1"
80   ON INTR 7 GOTO Sweep_end  ! \ When iBASIC is used, change "7" to "8"
90   ENABLE INTR 7;2           ! /
100    OUTPUT @Hp4396;"SING"
110 Measuring:GOTO Measuring
120 Sweep_end:  !
130  DISP "MEASUREMENT COMPLETE"
140  END
```

Figure 3-3. Sample Program : To Wait for Sweep End

Enable Sweep-End Bit

```
60   OUTPUT @Hp4396;"CLES"
70   OUTPUT @Hp4396;"*SRE 4;ESNB 1"
```

Line 60 clears all bits of the Status Registers and the Enable Registers.

In line 70, the command `*SRE 4` sets the Service Request Enable Register to 00000100 (this enables bit 2 of the Status Byte Register). The command `ESNB 1` sets the Event Status Enable Register B to 0000000000000001 (this enables bit 0 of the Event Status Register B. See Figure 3-4).

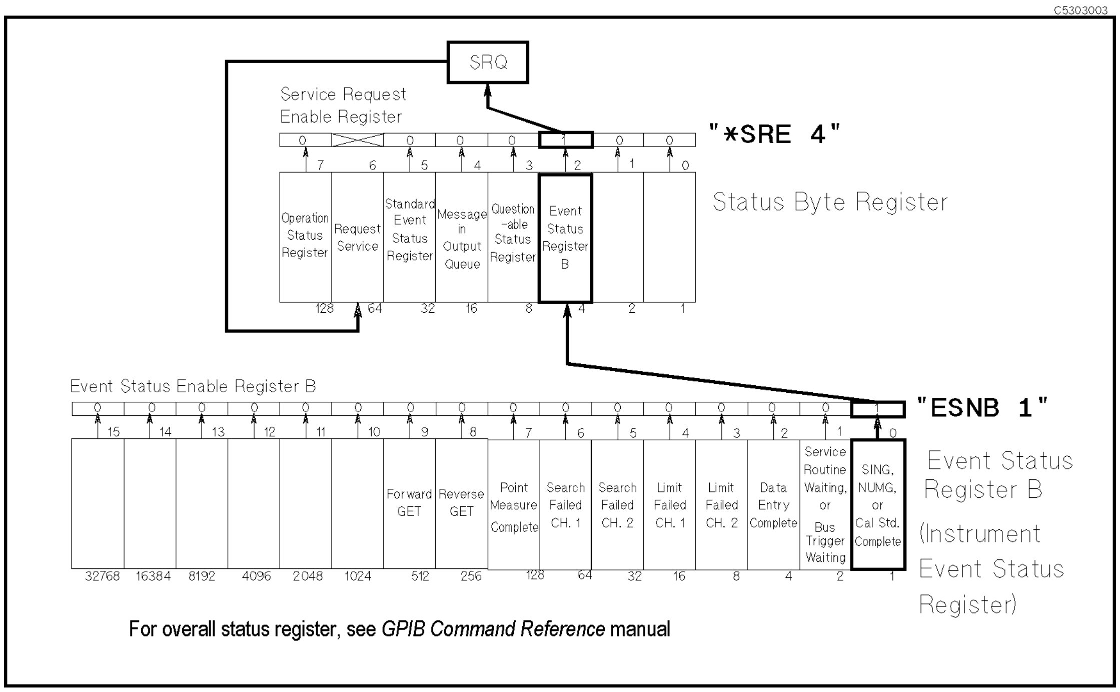

C5303003
SRQ
Service Request Enable Register
"*SRE 4"
Status Byte Register
Operation Status Register
Request Service
Standard Event Status Register
Message in Output Queue
Question-able Status Register
Event Status Register B
128 64 32 16 8 4 2 1
Event Status Enable Register B
"ESNB 1"
Forward GET
Reverse GET
Point Measure Complete
Search Failed CH. 1
Search Failed CH. 2
Limit Failed CH. 1
Limit Failed CH. 2
Data Entry Complete
Service Routine Waiting, or Bus Trigger Waiting
SING, NUMG, or Cal Std. Complete
32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
Event Status Register B (Instrument Event Status Register)
For overall status register, see GPIB Command Reference manual

Figure 3-4. Sweep-End Bit Enabling

Enable Registers select which events in the analyzer can cause a service request (SRQ). By setting bit 0 of the Event Status Enable Register B to 1, the occurrence of the corresponding event (sweep-end) sets bit 0 of the Event Status Register B. When this bit is set (and is enabled), it is used to set a summary bit in the Status Byte Register (bit 2). Also, because bit 2 of Service Request Enable Register is set, setting the corresponding bit (Event Status Register B summary bit) generates an SRQ. The SRQ sets bit 6 of the Status Byte Register.

Enable SRQ Interrupt

```
80   ON INTR 7 GOTO Sweep_end  ! \ When iBASIC is used, change "7" to "8"
90   ENABLE INTR 7;2           ! /
:
120 Sweep_end:  !
```

Line 80 defines a branch. When the SRQ interrupt is generated from the GPIB interface (whose select code is 7), the controller goes to `Sweep_end` (Line 120).

Line 90 enables an interrupt from interface 7 (GPIB) when bit 1 (SRQ bit) of the interrupt register (of the controller) is set by a value of 2. See the *GPIB Command Reference* for additional information.

Wait Until Measurement Is Done

```
100    OUTPUT @Hp4396;"SING"
110 Measuring:     GOTO Measuring
```

In line 100, the `SING` command triggers a measurement and the analyzer starts a sweep. For details on how to trigger a measurement, see Figure 2-3.

The controller loops back in line 110 until an SRQ interrupt occurs.

Generate SRQ

On a single sweep end, bit 0 of the ESB is set (which sets bit 2 of the Status Byte Register) and an SRQ is generated.

```
110 Measuring:     GOTO Measuring    Loop until SRQ interrupt
120 Sweep_end:  !                    At SRQ interrupt, jump to here
```

Once an SRQ is generated, the SRQ interrupt is disabled.

To Report Command Error Occurrence

```
10   !
20   ! Figure 3-5. To Report Command Error Occurrence
30   !
40   ASSIGN @Hp4396 TO 717  ! When iBASIC is used, change "717" to "800"
50   !
60   DIM Err$[30]
70   OUTPUT @Hp4396;"CLES"
80   OUTPUT @Hp4396;"*SRE 32 ;*ESE 32"
90   ON INTR 7 GOSUB Err_report   ! \ When iBASIC is used,
100  ENABLE INTR 7;2              ! / change "7" to "8"
110     !
120     !  OUTPUT statement to send GPIB command
130     !
140     !
150     !
160   GOTO Prog_end
170 Err_report:    !
180     OUTPUT @Hp4396;"OUTPERRO?"
190     ENTER @Hp4396;Err,Err$
200     PRINT "COMMAND ERROR DETECTED"
210     PRINT Err,Err$
220     !
230     A=SPOLL(@Hp4396)
240     OUTPUT @Hp4396;"*ESR?"
250     ENTER @Hp4396;Estat
260     ENABLE INTR 7    ! When iBASIC is used, change "7" to "8"
270   RETURN
280 Prog_end:   !
290   END
```

Figure 3-5. Sample Program : To Report Command Error Occurrence

For details on SRQ interrupt, see the “To Wait for Sweep End” example.

Enable Error Bit

```
70   OUTPUT @Hp4396;"CLES"
80   OUTPUT @Hp4396;"*SRE 32 ;*ESE 32"
```

Line 70 clears all bits of the Status Registers and Enable Registers.

In line 80, the command *SRE 32 sets the Service Request Enable Register to 00100000 (this enables bit 5 of the Status Byte Register). The command *ESE 32 sets the Standard Event Status Enable Register to 00100000 (this enables bit 5 of the Standard Event Status Register (see Figure 3-6).

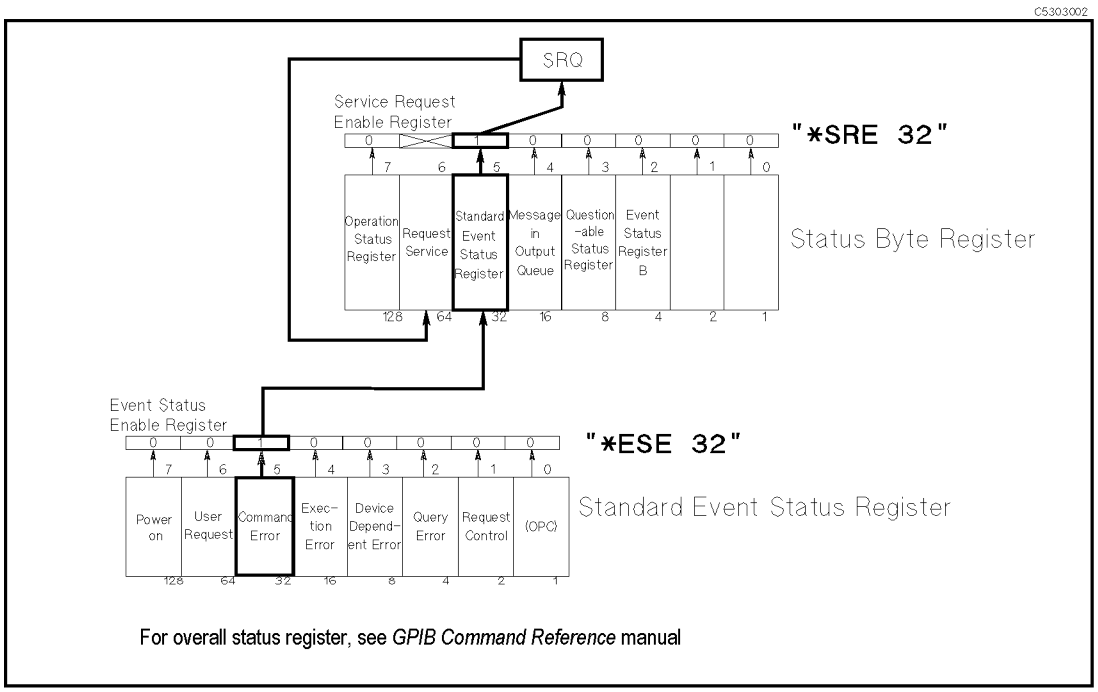

C5303002
SRQ
Service Request Enable Register
"*SRE 32"
Operation Status Register
Request Service
Standard Event Status Register
Message in Output Queue
Question-able Status Register
Event Status Register B
Status Byte Register
Event Status Enable Register
"*ESE 32"
Power on
User Request
Command Error
Exec-tion Error
Device Depend-ent Error
Query Error
Request Control
(OPC)
Standard Event Status Register
For overall status register, see GPIB Command Reference manual

Figure 3-6. Command-Error Bit Enabling

Report Command Error

```
 90   ON INTR 7 GOSUB Err_report   ! \ When iBASIC is used,
 100  ENABLE INTR 7;2              ! / change "7" to "8"
 110     !
 120     !  OUTPUT statement to send GPIB command
 130     !
 140     !
 150     !
 160    GOTO Prog_end
 170 Err_report:    !
```

If an GPIB command (executed between lines 100 and 160) causes an GPIB command error, the analyzer generates an SRQ and the controller branches to `Err_report`. For example, the `OUTPUT` statement:

`120 OUTPUT @Hp4396;"CENT "` *Setting center frequency, but no parameter*

causes an SRQ interrupt and branch to `Err_report`.

Output Error

```
 180     OUTPUT @Hp4396;"OUTPERRO?"
 190     ENTER @Hp4396;Err,Err$
 200     PRINT "COMMAND ERROR DETECTED"
 210     PRINT Err,Err$
```

These commands retrieve the error number and description.

In the error shown in the line 120 example, the controller displays the following:

```
COMMAND ERROR DETECTED
-109   "Missing parameter"
```

Return to Execute GPIB command

```
230      A=SPOLL(@Hp4396)
240      OUTPUT @Hp4396;"*ESR?"
250      ENTER @Hp4396;Estat
260      ENABLE INTR 7    ! When iBASIC is used, change "7" to "8"
270    RETURN
```

Lines 230 to 270 clear SRQ before returning to the main routine.

Line 230 reads the analyzer's status byte. The `A=SPOLL(@Hp4396)` statement reads the Status Byte Register of the address `@Hp4396`(analyzer), and enters the value into `A`. The command error causes the SRQ and with bit 5 and bit 6 of the Status Byte Register set, the value of `A` is 96. Reading the Status Byte Register by using the `SPOLL` command clears SRQ (status byte bit 6).

In line 240 and line 250, the command `*ESR?` reads the contents of the Standard Event Status Register. With Bit 5 of Standard Event Status Register set, the value of `Estat` is 32. Reading the Standard Event Status Register by using the `*ESR?` command clears the register.

A branch to `Err_report` disables the interrupt. Therefore, the return from `Err_report` must reenable the interrupt.

Reading Measurement Data

This chapter describes how to read measurement data over the GPIB.

Measurement data can be read out of the analyzer in the following ways:

1. Data can be read off the trace selectively using the markers.

   The present value of the marker (real-imaginary data and sweep parameter) is retrieved. For additional information on the marker functions, see the *Function Reference*.

2. The entire trace (or data for a specified number of points) can be read out in the following ways:

   - Data arrays — In regard to the data processing flow, the following data arrays are available.

     RAW DATA ARRAYS
     CALIBRATION COEFFICIENT ARRAYS
     DATA ARRAYS
     MEMORY ARRAYS
     DATA TRACE ARRAYS
     MEMORY TRACE ARRAYS

     For details about the data processing flow of the analyzer, see Chapter 12 of *Function Reference* manual.

   - Data format — The analyzer provides four data transfer formats.

FORM2	IEEE 32 bit floating point format
FORM3	IEEE 64 bit floating point format
FORM4	ASCII format
FORM5	MS-DOS® personal computer format

     Depending on the format, the data transfer speed and the number of digits are changed. Generally, binary data transfer (FORM2, FORM3, or FORM5) is faster than ASCII (FORM4).

     For details on data transfer format, see the *GPIB Command Reference* manual.

To Read Data Using the Marker Search Function

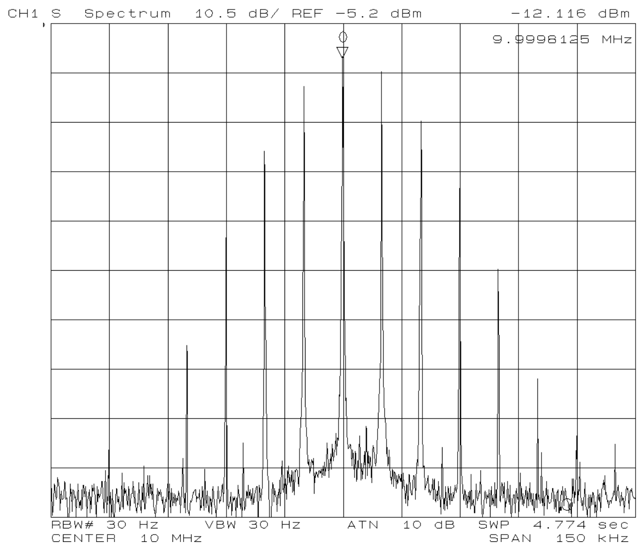
```
10    !
20    ! Figure 4-1. To Read Data Using Marker Search Function
30    !
40    ASSIGN @Hp4396 TO 717   ! When iBASIC is used, change "717" to "800"
50    INPUT "ENTER CENTER FREQUENCY (Hz)",F_cent   !  Setting 4396B
60    INPUT "ENTER FREQUENCY SPAN (Hz)",F_span     !
70    OUTPUT @Hp4396;"CENT ";F_cent                !
80    OUTPUT @Hp4396;"SPAN ";F_span                !
90    OUTPUT @Hp4396;"*OPC?"
100   ENTER @Hp4396;Dum
110  !
120   OUTPUT @Hp4396;"CLES"
130   OUTPUT @Hp4396;"*SRE 4;ESNB 1"
140   ON INTR 7 GOTO Sweep_end   ! \ When iBASIC is used, change "7" to "8"
150   ENABLE INTR 7;2            ! /
160     OUTPUT @Hp4396;"SING"        ! Trigger a Measurement
170 Measuring:    GOTO Measuring     ! Measuring
180 Sweep_end:   !
190     OUTPUT @Hp4396;"MKR ON"
200     OUTPUT @Hp4396;"SEAM MAX"
210     OUTPUT @Hp4396;"OUTPMKR?"
220     ENTER @Hp4396;Val1,Val2,Swp
230     PRINT "Max Val:",Val1;"dB"
240     PRINT "Swp.Prmtr:",Swp,"Hz"
250   END
```

Figure 4-1. Sample Program : To Read Data Using Marker Search Function

Search Maximum Value

```
190     OUTPUT @Hp4396;"MKR ON"
200     OUTPUT @Hp4396;"SEAM MAX"
```

Line 190 activates the marker and line 200 moves the marker to the maximum value on the trace.



Marker on Trace

What are the other marker commands?

You can activate sub-markers and the Δmarker using the following commands:

`SMKR{1-7} ON`, `DMKR {ON|FIX|TRAC}`

You can move the marker using the following commands:[1]

• specified sweep parameter	`MKRPRM` *parameter*
• specified measurement point	`MKRP` *parameter*

You can move sub-markers using the following commands:[1]

• specified sweep parameter	`SMKRPRM{1-7}` *parameter*
• specified measurement point	`SMKRP{1-7}` *parameter*

You can move the Δmarker using the following commands:[1]

• specified sweep parameter	`DMKRPRM` *parameter*
• specified primary part of marker value	`DMKRVAL` *parameter*
• specified secondary part of marker value	`DMKRAUV` *parameter*

1 Before executing these commands, you must turn on the markers to be moved.

Read Data

```
210      OUTPUT @Hp4396;"OUTPMKR?"
220      ENTER @Hp4396;Val1,Val2,Swp
```

The `OUTPMKR?` command returns the marker value in the following order: primary part of data, secondary part of data, and sweep parameter.

What are other marker value commands?

You can get the marker value using the following commands:

• get primary part of marker value	`MKRVAL?`
• get secondary part of marker value	`MKRAUV?`
• get sweep parameter	`MKRPRM?`
• get data point number	`MKRP?`

You can get the sub-marker value using the following commands:

• get primary part of sub-marker value	`SMKRVAL{1-7}?`
• get secondary part of sub-marker value	`SMKRAUV{1-7}?`
• get sweep parameter	`SMKRPRM{1-7}?`
• get data point number	`SMKRP{1-7}?`

You can get the Δmarker value using the following commands:

• get primary part of Δmarker value	`DMKRVAL?`
• get secondary part of Δmarker value	`DMKRAUV?`
• get sweep parameter	`DMKRPRM?`

To Get Measurement Trace Using ASCII Format

```
10    !
20    ! Figure 4-2. To Get Measurement Trace Using ASCII Format
30    !
40    ASSIGN @Hp4396 TO 717  ! When iBASIC is used, change "717" to "800"
50    !
60    INPUT "ENTER CENTER FREQUENCY (Hz)",F_cent
70    INPUT "ENTER FREQUENCY SPAN (Hz)",F_span
80    OUTPUT @Hp4396;"CENT";F_cent
90    OUTPUT @Hp4396;"SPAN";F_span
100   !
110   OUTPUT @Hp4396;"CLES"
120   OUTPUT @Hp4396;"*SRE 4;ESNB 1"
130   ON INTR 7 GOTO Sweep_end  ! \ When iBASIC is used, change "7" to "8"
140   ENABLE INTR 7;2           ! /
150     OUTPUT @Hp4396;"SING"       ! Trigger a Measurement
160 Measuring:    GOTO Measuring    ! Measuring
170 Sweep_end:   !
180     DIM Dat(1:801,1:2),Swp(1:801)  ! For spectrum measurement, change
190     OUTPUT @Hp4396;"FORM4"         ! "Dat(1:801,1:2)" to "Dat(1:801)"
200     OUTPUT @Hp4396;"OUTPDTRC?"
210     ENTER @Hp4396 USING "%,K";Dat(*)
220     OUTPUT @Hp4396;"OUTPSWPRM?"
230     ENTER @Hp4396 USING "%,K";Swp(*)
240     !
250     OUTPUT @Hp4396;"POIN?"
260     ENTER @Hp4396;Nop
270     FOR I=1 TO Nop
280       PRINT Swp(I);"Hz",Dat(I,1);"dB" ! For spectrum measurement, change
290     NEXT I                            ! "Dat(I,1)" to "Dat(I)"
300     END
```

Figure 4-2. Sample Program : To Get Measurement Trace Using ASCII Format

Set the Receive Array

```
180     DIM Dat(1:801,1:2),Swp(1:801)
```

Line 180 sets the array size to the analyzer's maximum number of measurement points (801).

In this example, it is assumed that the analyzer is in the network analyzer mode of operation, in which each point has complex data. If you use the analyzer in the spectrum analyzer mode, each measurement point has only real data, so you must set the data array `Dat` as follows:

```
180     DIM Dat(1:801),Swp(1:801)
```

```
280     PRINT Swp(I);"Hz",Dat(I);"dB"
```

If the number of measurement points changes, then so does the number of data. You must control the number of entered measurement data (see lines 210 and 230).

Set Data Transfer Format

```
190      OUTPUT @Hp4396;"FORM4"
```

Line 190 tells the analyzer to use the ASCII transfer format.

Read Data

```
200      OUTPUT @Hp4396;"OUTPDTRC?"
210      ENTER @Hp4396 USING "%,K";Dat(*)
220      OUTPUT @Hp4396;"OUTPSWPRM?"
230      ENTER @Hp4396 USING "%,K";Swp(*)
```

`OUTPDTRC?` retrieves DATA TRACE ARRAYS, and `OUTPSWPRM?` retrieves sweep parameters.

In line 210 and 230, you must choose `%,K` to allow for an insufficient number of data points to fill the array (which is 801 as declared in line 180).

What are other data arrays?

You can retrieve the following data arrays, exchanging GPIB command `OUTPDTRC?` in line 200. For details on each command, see the *GPIB Command Reference* manual.

• RAW DATA ARRAYS	`OUTPRAW{1-4}?`
• DATA ARRAYS	`OUTPDATA?`
• MEMORY ARRAYS	`OUTPMEMO?`
• MEMORY TRACE ARRAYS	`OUTPMTRC?`
• CALIBRATION COEFFICIENT ARRAYS	`OUTPCALC{1-12}?`
• SWEEP PARAMETER ARRAYS	`OUTPSWPRM?`

To Get Measurement Trace Using Binary Format

Two programs are shown to get the data arrays using binary format in Figure 4-3 and Figure 4-5.

Figure 4-3	**Figure 4-5**
COMPUTER For the external controller	iBASIC For the Instrument BASIC
• Using REDIM command (line 220) to allow for the change in the number of measurement points.	• Instrument BASIC allows for a different number of data points then the number specified in the data array declaration.

Before running the program in Figure 4-3, you must modify the dimension of the data arrays to match to the analyzer type (network or spectrum). (See the "Set the Receive Array" example.)

```
10  !
20  ! Figure 4-3. To Get Measurement Trace Using
30  !             IEEE 64-bit Floating point Format (For External Controller)
40  !
50    ASSIGN @Hp4396 TO 717
60    !
70    INPUT "ENTER CENTER FREQUENCY (Hz)",F_cent
80    INPUT "ENTER FREQUENCY SPAN (Hz)",F_span
90    OUTPUT @Hp4396;"CENT";F_cent
100   OUTPUT @Hp4396;"SPAN";F_span
110   !
120   OUTPUT @Hp4396;"CLES"
130   OUTPUT @Hp4396;"*SRE 4;ESNB 1"
140   ON INTR 7 GOTO Sweep_end   !
150   ENABLE INTR 7;2            !
160     OUTPUT @Hp4396;"SING"
170 Measuring:GOTO Measuring
180 Sweep_end:   !
190   DIM Dat(1:801,1:2),Swp(1:801)    ! For spectrum measurement, change
200   OUTPUT @Hp4396;"POIN?"           ! "Dat(1:801,1:2)" to "Dat(1:801)"
210   ENTER @Hp4396;Nop
220   REDIM Dat(1:Nop,1:2),Swp(1:Nop)
230   OUTPUT @Hp4396;"FORM3"
240   ASSIGN @Dt TO 717;FORMAT OFF
250   OUTPUT @Hp4396;"OUTPDTRC?"
260   ENTER @Dt USING "%,8A";A$
270   ENTER @Dt;Dat(*)
280   ENTER @Dt USING "%,1A";B$
290   OUTPUT @Hp4396;"OUTPSWPRM?"
300   ENTER @Dt USING "%,8A";A$
310   ENTER @Dt;Swp(*)
320   ENTER @Dt USING "%,1A";B$
```

Figure 4-3.
Sample Program : To Get Measurement Trace Using IEEE 64-bit Floating Point Format (For External Controller) (1/2)

```
330   ASSIGN @Dt TO *
340   !
350   FOR I=1 TO Nop
360     PRINT Swp(I);"Hz",Dat(I,1);"dB"    ! For spectrum measurement, change
370   NEXT I                               ! "Dat(I,1)" to "Dat(I)"
380   END
```

Figure 4-3. Sample Program : To Get Measurement Trace Using IEEE 64-bit Floating Point Format (For External Controller) (2/2)

This program is similar to the ASCII transfer program. However, you must set the data transfer format OFF when using the binary data transfer format.

Set the Receive Array

```
190   DIM Dat(1:801,1:2),Swp(1:801)    ! For spectrum measurement, change
200   OUTPUT @Hp4396;"POIN?"           ! "Dat(1:801,1:2)" to "Dat(1:801)"
210   ENTER @Hp4396;Nop
220   REDIM Dat(1:Nop,1:2),Swp(1:Nop)
```

Line 190 sets the array size to the analyzer's maximum number of measurement points (801).

In this example, it is assumed that the analyzer is in the network analyzer mode of operation, in which each point has complex data. If you use the analyzer in the spectrum analyzer mode, each measurement point has only real data, so you must set the data array `Dat` as follows:

```
190     DIM Dat(1:801),Swp(1:801)

220     REDIM Dat(1:Nop),Swp(1:Nop)

360     PRINT Swp(I);"Hz",Dat(I);"dB"
```

Lines 200 and 210 interrogate the analyzer to determine the number of measurement points. Line 220 resizes the receive array to match the data.

Set Data Transfer Format

```
200   OUTPUT @Hp4396;"FORM3"
210   ASSIGN @Dt TO 717;FORMAT OFF ! When iBASIC is used, change "717" to "800"
```

To use FORM3 the computer must be instructed to stop formatting the incoming data with the `ENTER` statement. This is done by defining an I/O path with ASCII formatting OFF. The I/O path points to the analyzer. This path can be used to read or write data to the analyzer, as long as that data is in binary rather than ASCII format.

What are other binary data formats?

You can use the following data transfer formats, by changing the GPIB command `FORM3` in line 200.

• IEEE 32 bit floating point format	`FORM2`
• MS-DOS® personal computer format	`FORM5`

Read Data

```
250     OUTPUT @Hp4396;"OUTPDTRC?"
260     ENTER @Dt USING "%,8A";A$
270     ENTER @Dt;Dat(*)
280     ENTER @Dt USING "%,1A";B$
290     OUTPUT @Hp4396;"OUTPSWPRM?"
300     ENTER @Dt USING "%,8A";A$
310     ENTER @Dt;Swp(*)
320     ENTER @Dt USING "%,1A";B$
```

FORM3 has an eight-byte header to deal with. The first two bytes are the ASCII characters #6. This indicates that a fixed length block transfer follows and that the next 6 bytes form an integer specifying the number of bytes in the block to follow. The header must be read in so that data order is maintained (lines 260 and 300).

At the data end, the terminator "LF^EOI" is sent(lines 280 and 320).

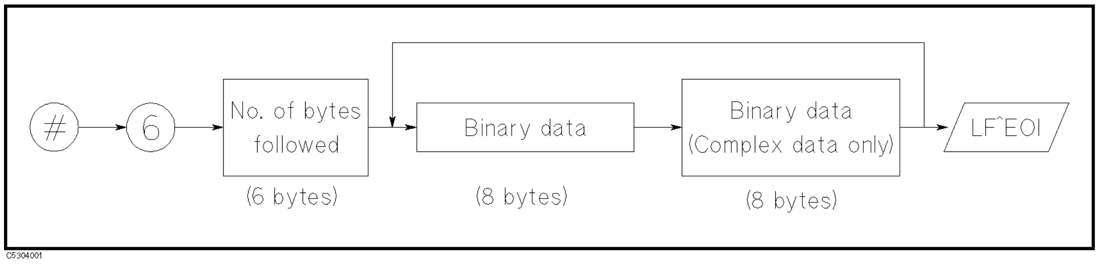


Figure 4-4. FORM3 Data Transfer Format

Before running the program in Figure 4-5, you must modify the dimension of the data arrays to match the analyzer type (network or spectrum). (See the “Set the Receive Array” below.)

```
10  !
20  ! Figure 4-5. To Get Measurement Trace Using
30  !             IEEE 64-bit Floating point Format (For Instrument BASIC)
40  !
50    ASSIGN @Hp4396 TO 800
60    !
70    INPUT "ENTER CENTER FREQUENCY (Hz)",F_cent
80    INPUT "ENTER FREQUENCY SPAN (Hz)",F_span
90    OUTPUT @Hp4396;"CENT";F_cent
100   OUTPUT @Hp4396;"SPAN";F_span
110   !
120   OUTPUT @Hp4396;"CLES"
130   OUTPUT @Hp4396;"*SRE 4;ESNB 1"
140   ON INTR 8 GOTO Sweep_end
150   ENABLE INTR 8;2
160     OUTPUT @Hp4396;"SING"
170 Measuring:GOTO Measuring
180 Sweep_end:   !
190   DIM Dat(1:802,1:2),Swp(1:802)   ! For spectrum measurement, change
200   OUTPUT @Hp4396;"FORM3"          ! "Dat(1:802,1:2)" to "Dat(1:802)"
210   ASSIGN @Dt TO 800;FORMAT OFF
220   OUTPUT @Hp4396;"OUTPDTRC?"
230   ENTER @Dt USING "%,8A";A$
240   ENTER @Dt;Dat(*)
250   OUTPUT @Hp4396;"OUTPSWPRM?"
260   ENTER @Dt USING "%,8A";A$
270   ENTER @Dt;Swp(*)
280   ASSIGN @Dt TO *
290   !
300   OUTPUT @Hp4396;"POIN?"
310   ENTER @Hp4396;Nop
320   FOR I=1 TO Nop
330     PRINT Swp(I);"Hz",Dat(I,1);"dB"   ! For spectrum measurement, change
340   NEXT I                              ! "Dat(I,1)" to "Dat(I)"
350   END
```

Figure 4-5.
Sample Program : To Get Measurement Trace Using IEEE 64-bit Floating Point Format (For Instrument BASIC)

Set the Receive Array

```
190   DIM Dat(1:802,1:2),Swp(1:802)
```

Line 190 sets the array size to the maximum number of data. The analyzer’s maximum number of measurement points is 801. At end of data, the terminator “LF^EOI” is sent (see Figure 4-4).

In this example, it is assumed that the analyzer is in the network analyzer mode of operation, in which each point has complex data. If you use the analyzer in the spectrum analyzer mode, each measurement point has only real data, so you must set the data array `Dat` as follows:

```
190     DIM Dat(1:802),Swp(1:802)
330     PRINT Swp(I);"Hz",Dat(I);"dB"
```

If the number of measurement points change, then so does the number of data. Instrument BASIC allows for insufficient data in lines 220 to 270 (see "Read Data").

Set Data Transfer Format

```
200   OUTPUT @Hp4396;"FORM3"
210   ASSIGN @Dt TO 800;FORMAT OFF
```

To use FORM3 the computer must be instructed to stop formatting the incoming data with the ENTER statement. This is the same operation that is done in the program in Figure 4-3. For more information, see "Set Data Transfer Format" in Figure 4-3.

Read Data

```
220     OUTPUT @Hp4396;"OUTPTDTRC?"
230     ENTER @Dt USING "%,8A";A$
240     ENTER @Dt;Dat(*)
250     OUTPUT @Hp4396;"OUTPSWPRM?"
260     ENTER @Dt USING "%,8A";A$
270     ENTER @Dt;Swp(*)
```

When all the data is transferred and no data is left for the receive arrays, `Dat(*)` and `Swp(*)`, the program goes to the next operation.

5

Writing Data Arrays to the Analyzer

Chapter 4 explained how to read data arrays from the analyzer. This chapter shows how to write data arrays to the analyzer. You can read, modify, and store the data arrays to the analyzer over the GPIB. This allows you to modify the trace on the analyzer's display. For details on the data arrays in the analyzer, see the *GPIB Command Reference* manual.

To Modify Calibration Data

Note

If you change measurement conditions (for example, START frequency, STOP frequency, NOP, and IFBW) of this instrument after exporting a calibration coefficient array using the OUTPCALC{1-12}? command and inporting it again using the INPUCALC{1-12} command, contents of the imported calibration coefficient array are destroyed and become invalid. If you want to perform in this way, use the SAVDSTA command to save data as STATE and recall it using the RECD command.

```
10     !
20     ! Figure 5-1. To Modify Calibration Data
30     !
40     ASSIGN @Hp4396 TO 717  ! When iBASIC is used, change "717" to "800"
50     !
60     OUTPUT @Hp4396;"PRES"
70     OUTPUT @Hp4396;"NA"
80     INPUT "Enter center frequency(Hz).",F_cent
90     INPUT "Enter frequency span(Hz).",F_span
100    OUTPUT @Hp4396;"CENT ";F_cent
110    OUTPUT @Hp4396;"SPAN ";F_span
120    OUTPUT @Hp4396;"HOLD"
130    !
140    ! Calibration
150    OUTPUT @Hp4396;"CLES"
160    OUTPUT @Hp4396;"*SRE 4;ESNB 1"    ! Set enable STB and ESB
170    INPUT "Connect THRU and press [RETURN] to do CAL.",Dum$
180    OUTPUT @Hp4396;"CALI RESP"
190    ON INTR 7 GOTO Cal_end    ! \ When iBASIC is used,
200    ENABLE INTR 7;2           ! / change "7" to "8"
210      OUTPUT @Hp4396;"STANC"          ! Measure THRU
220 Calibrating:GOTO Calibrating
230 Cal_end:     !
240    OUTPUT @Hp4396;"RESPDONE"         ! Calculating cal coefficient
250    OUTPUT @Hp4396;"*OPC?"         ! \ Wait until calculating ends
260    ENTER @Hp4396;Dum              ! /
```

Figure 5-1. Sample Program : To Modify Calibration Data (1/2)

```
 270    DISP "Calibration Complete"
 280    !
 290    !  Read Calibration Data
 300    DIM Dat(1:801,1:2)   ! When iBASIC is used, change "801" to "802"
 310    OUTPUT @Hp4396;"POIN?"  ! \
 320    ENTER @Hp4396;Nop       ! | When iBASIC is used,delete these lines
 330    REDIM Dat(1:Nop,1:2)    ! /
 340    ASSIGN @Dt TO 717;FORMAT OFF     ! When iBASIC is used,
 350    OUTPUT @Hp4396;"FORM3"           ! change "717" to "800"
 360    OUTPUT @Hp4396;"OUTPCALC1?"
 370    ENTER @Dt USING "%,8A";Head$
 380    ENTER @Dt;Dat(*)
 390    ENTER @Dt USING "%,1A";Dum$  ! When iBASIC is used, delete this line
 400 !
 410 !   Modify Calibration Data
 420 !
 430    !   Restore Calibration  Data
 440    OUTPUT @Hp4396;"INPUCALC1 ";
 450    OUTPUT @Dt USING "#,8A";Head$
 460    OUTPUT @Dt;Dat(*),END
 470    ASSIGN @Dt TO *
 480    OUTPUT @Hp4396;"SAVC" ! Redraw Trace
 490    END
```

Figure 5-1. Sample Program : To Modify Calibration Data (2/2)

This program measures calibration standards, reads the obtained calibration data, and restores the data in the analyzer. For details on how to obtain the calibration data by measuring the standard, see Chapter 1. For details on how to transfer the data arrays, see Chapter 4.

Read Calibration Data

```
 290    !  Read Calibration Data
 300    DIM Dat(1:801,1:2)   ! When iBASIC is used, change "801" to "802"
 310    OUTPUT @Hp4396;"POIN?"  ! \
 320    ENTER @Hp4396;Nop       ! | When iBASIC is used,delete these lines
 330    REDIM Dat(1:Nop,1:2)    ! /
 340    ASSIGN @Dt TO 717;FORMAT OFF     ! When iBASIC is used,
 350    OUTPUT @Hp4396;"FORM3"           ! change "717" to "800"
 360    OUTPUT @Hp4396;"OUTPCALC1?"
 370    ENTER @Dt USING "%,8A";Head$
 380    ENTER @Dt;Dat(*)
 390    ENTER @Dt USING "%,1A";Dum$  ! When iBASIC is used, delete this line
```

The controller can read out the error coefficients using the GPIB commands `OUTPCALC{1-12}`. Each point is a real/imaginary pair, and the number of points in the array is the same as the number of points in the sweep. For details on data transfer, see Chapter 4.

Each calibration type uses only as many arrays as needed, starting with array 1, and each array stores a specific error coefficient. Therefore, it is necessary to know the type of calibration about to be read out: attempting to read an array not being used in the current calibration causes the "REQUESTED DATA NOT CURRENTLY AVAILABLE" warning to be displayed. For assignment of data arrays, see the *GPIB Command Reference* manual.

Modify Calibration Data

```
400 !
410 !   Modify Calibration Data
420 !
```

In this portion of program, you modify the CALIBRATION COEFFICIENT ARRAY, which is contained in `Dat(1:801,1:2)`.

Restore Modified Calibration Data

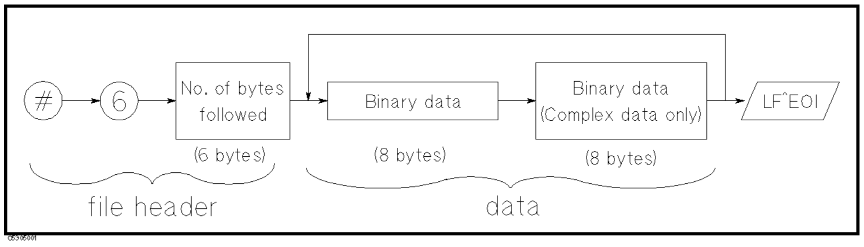
```
430     !   Restore Calibration  Data
440     OUTPUT @Hp4396;"INPUCALC1 ";
450     OUTPUT @Dt USING "#,8A";Head$
460     OUTPUT @Dt;Dat(*),END
```

Line 440 opens the CALIBRATION COEFFICIENT ARRAY 1 in the analyzer. This array is used to restore the data.

Lines 450 and 460 send the file header (`Head$`), calibration data (`Dat(*)`) and the terminator (`END`). The file header is an input in line 370.

This example sets the trigger to HOLD at line 120. The analyzer does not redraw the trace with the new CALIBRATION COEFFICIENT ARRAYS when the trigger is set to HOLD. You can redraw the trace by issuing the GPIB command `SAVC`. For details, see the "Redrawing Measurement Trace with Modified Calibration Data" description.

What is a file header? When using the binary data transfer format, the transferred data must be accompanied by the file header that represents the data length. In this example, the data transfer format is FORM3 and the transferred data is configured as follows:



FORM3 Data Transfer Format

If you are not reading the header, you can create it using the number of data points. Change the program lines 440 to 460 as follows:

```
440   OUTPUT @Hp4396;"POIN?"
441   ENTER @Hp4396;Nop
442   V$=VAL$(Nop*2*8)
443   Numv=LEN(V$)
444   Head$="000000"
445   FOR I=1 TO Numv
446     Head$[7-I,7-I]=V$[Numv-I+1,Numv-I+1]
447   NEXT I
448   !
449   OUTPUT @Hp4396;"INPUCALC1 ";
450   OUTPUT @Dt USING "#,8A";"#6"&Head$
460   OUTPUT @Dt;Dat(*),END
```

Lines 440 to 442 calculate the number of bytes transferred (8 byte for real part, 8 byte for imaginary part), and represents it in the string format.

Line 443 counts the number of characters in the string that contains the number of bytes transferred.

Line 444 enters 0 as the initial value in all header arrays.

Lines 445 to 447 place the number of bytes transferred to the header array digit by digit from the sixth array to the first array of the header.

For example, if the number of points is 201, the value of `Head$` is `003216`.

Redrawing Measurement Trace with Modified Calibration Data

```
480    OUTPUT @Hp4396;"SAVC" ! Redraw Trace
```

When all the calibration coefficients are in the analyzer, send the GPIB command `SAVC` to have the analyzer reshape a trace with the coefficients.

How can you modify the trace? (Summary) To modify the trace on the display, you rewrite the data arrays in the analyzer. Figure 5-2 shows the relation of the data arrays, data processing, and GPIB commands.

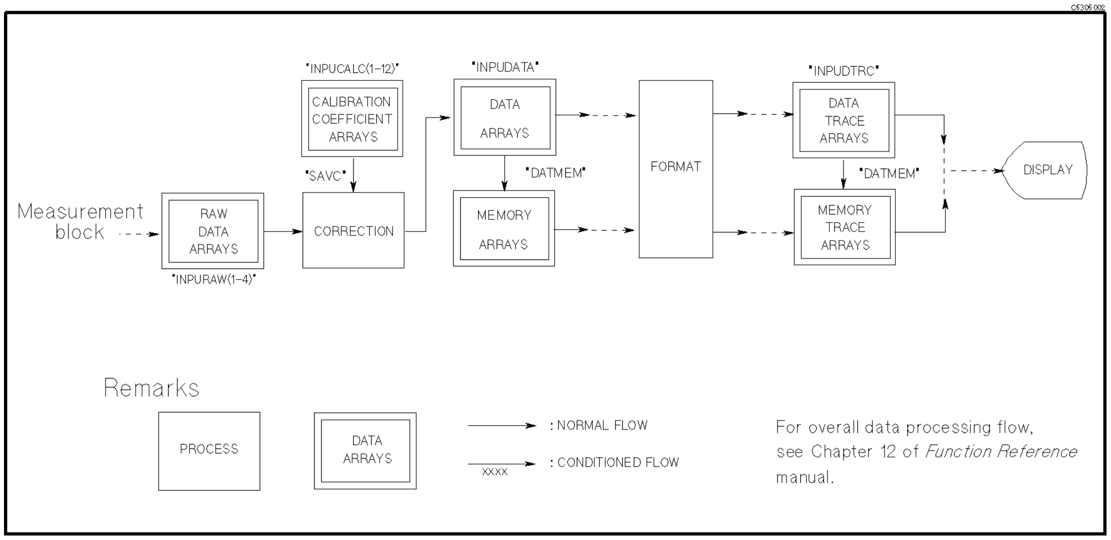


Figure 5-2. Data Arrays, Data Processing, and GPIB Command

- Reset command

`*RST` or `PRES` command clears all arrays.

- Data array writing command

`INPURAW{1-4}`, `INPUDATA` and `INPUDTRC` commands write the corresponding arrays. These commands immediately reshape the data trace on the analyzer's display.

`INPUCALC{1-12}` commands write the CALIBRATION COEFFICIENT ARRAYS.

- DATA to MEMORY command

`DATMEM` command restores the contents in DATA ARRAYS into MEMORY ARRAYS, and the contents in DATA TRACE ARRAYS into MEMORY TRACE ARRAYS.

- Data processing command

`SAVC` command executes the data processing CORRECTION with the current RAW ARRAYS and CALIBRATION COEFFICIENT ARRAYS.

The following examples show how to modify the DATA ARRAYS and DATA TRACE ARRAYS.

To Modify Error-Corrected Data

```
10    !
20    ! Figure 5-3. To Modify Error-Corrected Data
30    !
40    ASSIGN @Hp4396 TO 717  ! When iBASIC is used, change "717" to "800"
50    !
60    OUTPUT @Hp4396;"PRES"
70    OUTPUT @Hp4396;"NA"
80    INPUT "Enter center frequency(Hz).",F_cent
90    INPUT "Enter frequency span(Hz).",F_span
100   OUTPUT @Hp4396;"CENT ";F_cent
110   OUTPUT @Hp4396;"SPAN ";F_span
120    !
130   INPUT "Connect DUT and press [RETURN].",Dum$
140   OUTPUT @Hp4396;"CLES"
150   OUTPUT @Hp4396;"*SRE 4;ESNB 1"    ! Set enable STB and ESB
160   ON INTR 7 GOTO Sweep_end  ! \ When iBASIC is used,
170   ENABLE INTR 7;2           ! / change "7" to "8"
180     OUTPUT @Hp4396;"SING"
190 Measuring:     GOTO Measuring
200 Sweep_end:     !
210   DISP "Measurement Complete"
220   !
230   !  Read Error-Corrected Data
240   DIM Dat(1:801,1:2)   ! When iBASIC is used, change "801" to "802"
250   OUTPUT @Hp4396;"POIN?"  ! \
260   ENTER @Hp4396;Nop       ! | When iBASIC is used, delete these lines
270   REDIM Dat(1:Nop,1:2)    ! /
280   ASSIGN @Dt TO 717;FORMAT OFF   ! When iBASIC is used,
290   OUTPUT @Hp4396;"FORM3"         ! change "717" to "800"
300   OUTPUT @Hp4396;"OUTPDATA?"
310   ENTER @Dt USING "%,8A";Head$
320   ENTER @Dt;Dat(*)
330   ENTER @Dt USING "%,1A";Dum$    ! When iBASIC is used, delete this line
340 !
350 !   Modify Error-Corrected Data
360 !
370   !   Restore Error-Corrected Data
380   OUTPUT @Hp4396;"INPUDATA ";
390   OUTPUT @Dt USING "#,8A";Head$
400   OUTPUT @Dt;Dat(*),END
410   ASSIGN @Dt TO *
420   END
```

Figure 5-3. Sample Program : To Modify Error-Corrected Data

This program measures the DUT, reads the obtained data, and restores the data in the analyzer. For details on how to read the data array, see Chapter 4. For information on how to modify the trace on the display, see the “To Modify Calibration Data” example.

Read Error-Corrected Data

```
230   !  Read Error-Corrected Data
240   DIM Dat(1:801,1:2)   ! When iBASIC is used, change "801" to "802"
250   OUTPUT @Hp4396;"POIN?"  ! \
260   ENTER @Hp4396;Nop       ! | When iBASIC is used, delete these lines
270   REDIM Dat(1:Nop,1:2)    ! /
280   ASSIGN @Dt TO 717;FORMAT OFF   ! When iBASIC is used,
290   OUTPUT @Hp4396;"FORM3"         ! change "717" to "800"
300   OUTPUT @Hp4396;"OUTPDATA?"
310   ENTER @Dt USING "%,8A";Head$
320   ENTER @Dt;Dat(*)
330   ENTER @Dt USING "%,1A";Dum$    ! When iBASIC is used, delete this line
```

`OUTPDATA?` command (line 300) retrieves DATA ARRAYS in the analyzer. For details on data transfer, see Chapter 4.

Restore Modified Error-Corrected Data

```
370   !   Restore Error-Corrected Data
380   OUTPUT @Hp4396;"INPUDATA ";
390   OUTPUT @Dt USING "#,8A";Head$
400   OUTPUT @Dt;Dat(*),END
```

Line 380 opens the DATA ARRAYS in the analyzer to restore the data.

Lines 390 to 400 transfer data in FORM3 (a similar procedure is used in the “To Modify Calibration Data” example).

To Modify Trace Data

```
 10    !
 20    ! Figure 5-4. To Modify Trace Data
 30    !
 40    ASSIGN @Hp4396 TO 717  ! When iBASIC is used, change "717" to "800"
 50    !
 60    OUTPUT @Hp4396;"PRES"
 70    OUTPUT @Hp4396;"NA"
 80    INPUT "Enter center frequency(Hz).",F_cent
 90    INPUT "Enter frequency span(Hz).",F_span
 100   OUTPUT @Hp4396;"CENT ";F_cent
 110   OUTPUT @Hp4396;"SPAN ";F_span
 120    !
 130   INPUT "Connect DUT and press [RETURN].",Dum$
 140   OUTPUT @Hp4396;"CLES"
 150   OUTPUT @Hp4396;"*SRE 4;ESNB 1"    ! Set enable STB and ESB
 160   ON INTR 7 GOTO Sweep_end    ! \ When iBASIC is used,
 170   ENABLE INTR 7;2             ! / change "7" to "8"
 180     OUTPUT @Hp4396;"SING"
 190 Measuring:     GOTO Measuring
 200 Sweep_end:     !
 210   DISP "Measurement Complete"
 220   !
 230   !  Read Trace Data
 240   DIM Dat(1:801,1:2)  ! When iBASIC is used, change "801" to "802"
 250   OUTPUT @Hp4396;"POIN?"  ! \
 260   ENTER @Hp4396;Nop       ! | When iBASIC is used, delete these lines
 270   REDIM Dat(1:Nop,1:2)    ! /
 280   ASSIGN @Dt TO 717;FORMAT OFF   ! When iBASIC is used,
 290   OUTPUT @Hp4396;"FORM3"         ! change "717" to "800"
 300   OUTPUT @Hp4396;"OUTPDTRC?"
 310   ENTER @Dt USING "%,8A";Head$
 320   ENTER @Dt;Dat(*)
 330   ENTER @Dt USING "%,1A";Dum$  ! When iBASIC is used, delete this line
 340 !
 350 !   Modify Trace Data
 360 !
 370   !   Restore Trace Data
 380   OUTPUT @Hp4396;"INPUDTRC ";
 390   OUTPUT @Dt USING "#,8A";Head$
 400   OUTPUT @Dt;Dat(*),END
 410   ASSIGN @Dt TO *
 420   END
```

Figure 5-4. Sample Program : To Modify Trace Data

This program measures the DUT, reads the obtained data, and restores the data into the analyzer. For details on how to read the data array, see Chapter 4.

For details on how to modify the trace on the display, see the “To Modify Calibration Data” example.

Read Trace Data

```
230   !  Read Trace Data
240   DIM Dat(1:801,1:2)  ! When iBASIC is used, change "801" to "802"
250   OUTPUT @Hp4396;"POIN?"  ! \
260   ENTER @Hp4396;Nop      ! | When iBASIC is used, delete these lines
270   REDIM Dat(1:Nop,1:2)    ! /
280   ASSIGN @Dt TO 717;FORMAT OFF   ! When iBASIC is used,
290   OUTPUT @Hp4396;"FORM3"         ! change "717" to "800"
300   OUTPUT @Hp4396;"OUTPDTRC?"
310   ENTER @Dt USING "%,8A";Head$
320   ENTER @Dt;Dat(*)
330   ENTER @Dt USING "%,1A";Dum$  ! When iBASIC is used, delete this line
```

The OUTPDTRC? command (line 300) retrieves trace data in the analyzer. For details on data transfer, see Chapter 4.

Restore Modified Trace Data

```
370   !   Restore Trace Data
380   OUTPUT @Hp4396;"INPUDTRC ";
390   OUTPUT @Dt USING "#,8A";Head$
400   OUTPUT @Dt;Dat(*),END
```

Line 380 opens the DATA TRACE ARRAYS in the analyzer to restore the data.

Lines 390 and 400 transfer data in FORM3 (a similar procedure is used in the "To Modify Calibration Data" example).

6

Printing or Plotting the Analyzer’s Display

This chapter describes how to print the information on the analyzer display using GPIB commands.

To Print Analyzer Display

Printer Preparation

1. Connect a printer using a parallel cable.
2. Turn the printer on.

Execute Print

To print the screen, execute the folowing command.

```
OUTPUT 800;"PRINALL"
```

Set the GPIB address when you execute from an external controller.

To Observe Printing

The Basic program shown below gives an example to detect printing end by using a SRQ interrupt.

```
10   !
20   ! Figure 6-1. To Observe Printing
30   !
40   ASSIGN @Hp4396 TO 800
50   !
60   OUTPUT @Hp4396;"CLES"
70   OUTPUT @Hp4396;"OSNT 512" !Catch High to Low Transition
80   OUTPUT @Hp4396;"OSPT 0"   !Disable Low to High Transitions
90   OUTPUT @Hp4396;"OSE 512"  !Enable OS Event Reg.
100   OUTPUT @Hp4396;"*SRE 128" !Enable OSR bit
120   ON INTR 8 GOTO La1
130   ENABLE INTR 8;2
140   OUTPUT @Hp4396;"PRINALL"
150   La1:!
160   GOTO La1
170   DISP "PRINT COMPLETE"
180   !
190   END
```

Figure 6-1. Sample Program : To Observe Printing

7

Controlling Instrument BASIC from Remote

This chapter is for programmers who use both Instrument BASIC and an external controller at the same time. This chapter shows how to pass control and how to use the PROGram subsystem commands. The PROGram subsystem is part of the SCPI commands hierarchy. See the *GPIB Command Reference* manual for more information on how to use SCPI commands.

If your GPIB has only one controller (either an external computer or the Instrument BASIC), skip this chapter.

To Control GPIB from Instrument BASIC

The analyzer must be the active controller of the GPIB (select code 7) when controlling the device on the GPIB. Initially, the system controller is active. The active controller can pass control to the analyzer (or other controllers, if there are others on the GPIB).

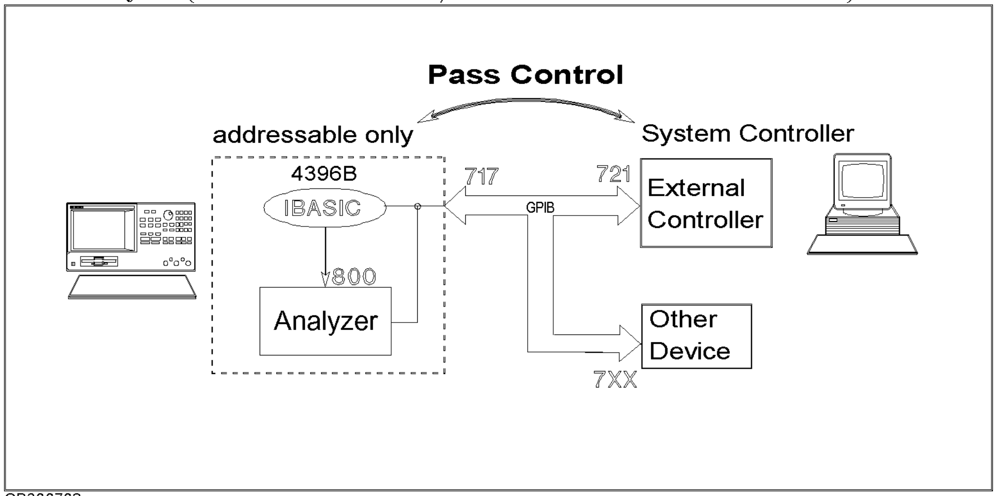


Figure 7-1. Pass Control

What is an active controller?

- The active controller determines which controller can manage the GPIB (that is, have control). The GPIB can have only one active controller at a time. If two or more controllers are on the same bus, control is passed from one controller to another.
- Resetting the GPIB (this operation can only be done by the system controller), the returns control to the system controller.

What is system controller?

- The controller that acts as the master controller.
- A controller can be set as either a system controller or a non-system controller. For details on how to set a controller, see the controller's manual. The analyzer, as a controller, can be used in either `SYSTEM CONTROLLER` (system controller) or `ADDRESSABLE ONLY` (non-system controller) mode. However, there can be only one system controller on the bus.

In this guide, it is assumed that the external controller is the system controller and the analyzer is used in `ADDRESSABLE ONLY` mode.

```
10  !
20  ! Figure 7-2. To Receive Control (on Instrument BASIC)
30  !
40  PRINTER IS 701
50  ON ERROR GOTO Not_active
60  Not_active: !
70  PRINT "HELLO WORLD!"
80  !
90  OFF ERROR
100 END
```

Figure 7-2. Sample Program: To Receive Control (On Instrument BASIC)

In line 60, to print out to the printer at address 701, the analyzer requires active control. Therefore, until active control is passed to the analyzer, the program loops back to line 30. When control is passed to the analyzer, it executes line 70.

To pass active control to Instrument BASIC:

```
PASS CONTROL 717 (Return)
```

Pass Control (On External Controller)

While the analyzer has control, it is free to address devices to talk and listen as needed. As the active controller, the analyzer can send messages to and read replies back from printers and plotters.

Note

The ability to assert the GPIB interface clear line (IFC) and remote enable line (REN) are reserved for the system controller. Even when Instrument BASIC has active control, it is denied these functions.

`ABORT 7`	*assert the interface clear line (IFC)*
`REMOTE 7`	*assert the remote enable line (REN)*

To return active control to the system controller:

```
PASS CONTROL 721 (Return)
```

Return Control (On Instrument BASIC)

Or, you can return control to the external controller by resetting the GPIB as follows:

```
ABORT 7 Return
```

Return Control (On External Controller)

This returns active control to the system controller.

Note The PROGram subsystem commands in the following programs can be used on the external controller. See the *GPIB Command Reference* manual for additional information on the use of SCPI commands.

To Execute an Instrument BASIC Command from the External Controller

Combine the `PROG:EXEC` command with Instrument BASIC command to be executed. For example, to execute `EDIT` command,

```
OUTPUT 717;"PROG:EXEC ""EDIT"""
```

Or you can change the double quoted term, `""EDIT""` to a single quote `'EDIT'`, as follows.

```
OUTPUT 717;"PROG:EXEC 'EDIT'"
```

Be careful when you execute an Instrument BASIC command that requires a parameter. For example, to execute the Instrument BASIC command "`GET "FILENAME"`", the quotation is as follows.

```
OUTPUT 717;"PROG:EXEC ""GET """"FILENAME"""""""
```

To Run an Instrument BASIC Program From the External Controller

```
10 ! Figure 7-3. To Run iBASIC Program From Ext.Controller
20 !             (On External Controller)
30 !
40   ABORT 7
50   ASSIGN @Hp4396 TO 717
60   OUTPUT @Hp4396;"PROG:DEL:ALL"
70   OUTPUT @Hp4396;"PROG:DEF #0"
80   OUTPUT @Hp4396;"10 MSI "":INTERNAL"""
90   OUTPUT @Hp4396;"20 GET ""FIG3_3"""
100  OUTPUT @Hp4396;"30 END"
110  OUTPUT @Hp4396;" " END
120  OUTPUT @Hp4396;"PROG:EXEC ""RUN"""
130  END
```

**Figure 7-3.
Sample Program : To Run the Instrument BASIC Program From the External Controller (On External Controller)**

You must put the disk that contains `FIG3_3` into the built-in disk drive of the analyzer before running the program. (The sample program disk for Instrument BASIC contains `FIG3_3`.)

Open the Instrument BASIC Editor

```
60    OUTPUT @Hp4396;"PROG:DEL:ALL"
70    OUTPUT @Hp4396;"PROG:DEF #0"
```

Scratch any program currently existing in the analyzer's Instrument BASIC editor and open the editor.

Send the Instrument BASIC Program

```
80    OUTPUT @Hp4396;"10 MSI "":INTERNAL"""
90    OUTPUT @Hp4396;"20 GET ""FIG3_3"""
100   OUTPUT @Hp4396;"30 END"
```

In the Instrument BASIC editor, the following program is now present:

```
10 MSI ":INTERNAL"
20 GET "FIG3_3"
30 END
```

Close the Instrument BASIC Editor

```
110   OUTPUT @Hp4396;" " END
```

Sending the `END` command to the analyzer closes the editor.

Run the Instrument BASIC Program

```
120   OUTPUT @Hp4396;"PROG:EXEC ""RUN"""
```

Line 120 runs the following program in the Instrument BASIC editor:

```
10 MSI ":INTERNAL"
20 GET "FIG3_3"
30 END
```

Line 20 retrieves the FIG3_3 program and at the same time runs the program.

To Transfer the Program to Instrument BASIC

```
10 !
20 !  Figure 7-4. To Transfer the Program to iBASIC (on External Controller)
30 !
40    ABORT 7
50    ASSIGN @Hp4396 TO 717
60    INPUT "FILENAME?",File_name$
70    OUTPUT @Hp4396;"PROG:DEL:ALL"
80    OUTPUT @Hp4396;"PROG:DEF #0"
90    ASSIGN @File TO File_name$
100   ON ERROR GOTO Done
110     DIM Line$[1024]
120     LOOP
130       Line$=""
140       ENTER @File;Line$
150       OUTPUT @Hp4396;Line$
160     END LOOP
170   Done:  !
180       OFF ERROR
190       OUTPUT @Hp4396;" " END
200       END
```

Figure 7-4.
Sample Program : To Transfer the Program to Instrument BASIC (on External Controller)

This Program transfers the program file in the mass storage of the external controller.

Before you run this program, confirm that the file to be transferred is on the mass storage device.

Open the Instrument BASIC Editor

```
70    OUTPUT @Hp4396;"PROG:DEL:ALL"
80    OUTPUT @Hp4396;"PROG:DEF #0"
```

Scratch any program that currently exists in the analyzer's Instrument BASIC editor and open the editor.

Transfer the Program

```
90    ASSIGN @File TO File_name$
100   ON ERROR GOTO Done
110     DIM Line$[1024]
120     LOOP
130       Line$=""
140       ENTER @File;Line$
150       OUTPUT @Hp4396;Line$
160     END LOOP
```

Transfer the program by line to the analyzer. When all program lines are transferred, the computer exits the loop and goes to line 170.

Close the Instrument BASIC Editor

```
190       OUTPUT @Hp4396;" " END
```

Sending the analyzer the END command closes the editor.

To Load an Array in an Instrument BASIC Program to the External Controller

```
10  !
20  ! Figure 7-5. To Load iBASIC Program Array (on External Controller)
30  !
40    ABORT 7
50    ASSIGN @Hp4396 TO 717
60    DIM Passed(1:801,1:2)
70    OUTPUT @Hp4396;"PROG:NUMB? ""Dat"";"
80    ENTER @Hp4396;Passed(*)
90    END
```

Figure 7-5.
Sample Program : To Load Instrument BASIC Program Array (on External Controller)

This program retrieves the array generated in the sample program listed in Figure 4-2 when that program is executed in Instrument BASIC. This information is transferred to the external controller.

First, run the Instrument BASIC program FIG4_2 to store the data into the array. Then run the program in Figure 7-5 to transfer the data.

Transfer the Program Array of Instrument BASIC

```
70    OUTPUT @Hp4396;"PROG:NUMB? ""Dat"";"
80    ENTER @Hp4396;Passed(*)
```

The PROG:NUMB? "Dat" query returns the program array Dat(1:801,1:2) of Figure 4-2. The array is entered into Passed(1:801,1:2).

8

Programming Limit Test from Remote

This chapter describes how to perform limit tests, set the limit lines, test the DUT, and transfer the results of the test.

To Perform limit Test

```
10   !
20   ! Figure 8-1. Limit Test
30   !
40    ASSIGN @Hp4396 TO 717  ! When iBASIC is used, change "717" to "800"
50   !
60    CLEAR SCREEN
70    PRINT USING "10A,15A,15A,15A";"Segment","Swp.Prmtr(Hz)","Upper","Lower"
80   !
90    DIM Table(1:18,1:3)
100   INPUT "Enter number of segments (<=18)",Numb
110   FOR I=1 TO Numb
120     GOSUB Loadlimit
130   NEXT I
140   !
150   LOOP
160      INPUT "Do you want to edit? (Y/N)",An$
170   EXIT IF An$="N" OR An$="n"
180      INPUT "Enter segment number(<=18)",I
190   IF Numb<I THEN Numb=I
200      GOSUB Loadlimit
210   END LOOP
220   !
230   OUTPUT @Hp4396;"EDITLIML"
240   OUTPUT @Hp4396;"LIMCLEL"
250   FOR K=1 TO Numb
260     OUTPUT @Hp4396;"LIMSADD"
270     OUTPUT @Hp4396;";LIMPRM ";Table(K,1)
280     OUTPUT @Hp4396;";LIMU ";Table(K,2)
290     OUTPUT @Hp4396;"LIML ";Table(K,3)
300     OUTPUT @Hp4396;"LIMSDON"
310   NEXT K
320   OUTPUT @Hp4396;"LIMEDONE"
330   OUTPUT @Hp4396;"LIMILINE ON"
340 !
350   INPUT "Connect DUT, and press Enter.",Dum$
```

Figure 8-1. Sample Program : Limit Test (1/2)

```
360   OUTPUT @Hp4396;"CLES"
370   OUTPUT @Hp4396;"*SRE 4;ESNB 1"
380   ON INTR 7 GOTO Sweep_end  ! \ When iBASIC is used, change "7" to "8"
390   ENABLE INTR 7;2           ! /
400     OUTPUT @Hp4396;"SING"
410 Measuring:GOTO Measuring
420 Sweep_end:   !
430   OUTPUT @Hp4396;"LIMITEST ON"
440   DIM Dt(1:801,1:4)
450   OUTPUT @Hp4396;"OUTPLIMF?"         ! \ Output test results.
460   ENTER @Hp4396 USING "%,K";Dt(*)    ! /
470   OUTPUT @Hp4396;"OUTPFAIP?"
480   ENTER @Hp4396;Failp
490   IF Failp=0 THEN Passed
500   PRINT "           FAIL POINTS             "
510   FOR I=1 TO Failp
520     PRINT
530     PRINT "Swp. prmtr : ";Dt(I,1)
540     PRINT "   Results       Upper         Lower    "
550     PRINT TAB(5);Dt(I,2);TAB(17);Dt(I,3);TAB(32);Dt(I,4)
560   NEXT I
570 Passed: !
580   DISP "Program End"
590   STOP
600   !
610 Loadlimit: !
620   INPUT "ENTER SWEEP PARAMETER (Hz)",Table(I,1)
630   INPUT "ENTER UPPER LIMIT VALUE",Table(I,2)
640   INPUT "ENTER LOWER LIMIT VALUE",Table(I,3)
650   PRINT I;TAB(11);Table(I,1);TAB(27);Table(I,2);TAB(42);Table(I,3)
660   RETURN
670 END
```

Figure 8-1. Sample Program : Limit Test (2/2)

Edit Limit Line

```
60   CLEAR SCREEN
70   PRINT USING "10A,15A,15A,15A";"Segment","Swp.Prmtr(Hz)","Upper","Lower"
80   !
90   DIM Table(1:18,1:3)
100   INPUT "Enter number of segments (<=18)",Numb
110   FOR I=1 TO Numb
120     GOSUB Loadlimit
130   NEXT I
140   !
150   LOOP
160      INPUT "Do you want to edit? (Y/N)",An$
170   EXIT IF An$="N" OR An$="n"
180      INPUT "Enter segment number(<=18)",I
190   IF Numb<I THEN Numb=I
200      GOSUB Loadlimit
210   END LOOP
```

```
  :
610 Loadlimit: !
620   INPUT "ENTER SWEEP PARAMETER (Hz)",Table(I,1)
630   INPUT "ENTER UPPER LIMIT VALUE",Table(I,2)
640   INPUT "ENTER LOWER LIMIT VALUE",Table(I,3)
650   PRINT I;TAB(11);Table(I,1);TAB(27);Table(I,2);TAB(42);Table(I,3)
660   RETURN
```

Lines 60 and 70 print the limit table heads on the BASIC SCREEN.

Line 90 defines the table (array `Table(1:18,1:3)`) used to hold the limit values. It contains the sweep parameter, the upper limit value, and the lower limit value as follows:

Segment	Sweep Parameter	Upper Limit	Lower Limit
1	`Table(1,1)`	`Table(1,2)`	`Table(1,3)`
2	`Table(2,1)`	`Table(2,2)`	`Table(2,3)`
⋮	⋮	⋮	⋮

Lines 110 to 130 call the subroutine `Loadlimit` (line 610) to edit and print as many segments as you defined in line 100 (the analyzer can retain up to 18 segments).

```
 Segment   Swp.Prmtr(Hz)  Upper          Lower
  1          2.E+6          0             -10
  2          3.E+6          10            -20
  3          4.E+6          10            -10
```

The loop, lines 150 to 210, determines if you want to edit the table and confirms that the segment is in the table.

Set Limit Line

```
230   OUTPUT @Hp4396;"EDITLIML"                  Start to set the limit line
240   OUTPUT @Hp4396;"LIMCLEL"                   Clear existing limit table
250   FOR K=1 TO Numb                                                          Setting the
260     OUTPUT @Hp4396;"LIMSADD"                 Start to set a segment        limit line
270     OUTPUT @Hp4396;";LIMPRM ";Table(K,1)     Enter sweep parameter
280     OUTPUT @Hp4396;";LIMU ";Table(K,2)       Enter upper limit        Setting a
290     OUTPUT @Hp4396;"LIML ";Table(K,3)        Enter lower limit        segment
300     OUTPUT @Hp4396;"LIMSDON"                 Complete the segment
310   NEXT K
320   OUTPUT @Hp4396;"LIMEDONE"                  Complete the limit line
330   OUTPUT @Hp4396;"LIMILINE ON"               Set limit line ON
```

In this portion of the program, the limit table (edited using BASIC) is transferred to the analyzer.

Read the Limit Test Results

```
430   OUTPUT @Hp4396;"LIMITEST ON"
440   DIM Dt(1:801,1:4)
450   OUTPUT @Hp4396;"OUTPLIMF?"         ! \ Output test results.
460   ENTER @Hp4396 USING "%,K";Dt(*)    ! /
470   OUTPUT @Hp4396;"OUTPFAIP?"
480   ENTER @Hp4396;Failp
490   IF Failp=0 THEN Passed
500   PRINT "           FAIL POINTS             "
510   FOR I=1 TO Failp
520     PRINT
530     PRINT "Swp. prmtr : ";Dt(I,1)
540     PRINT "   Results       Upper         Lower    "
550     PRINT TAB(5);Dt(I,2);TAB(17);Dt(I,3);TAB(32);Dt(I,4)
560   NEXT I
570 Passed: !
580   DISP "Program End"
```

The OUTPLIMF? command in line 450 returns the limit test result for failed points. The test results are in the following order: sweep parameter, result (0 for fail, −1 for no test), upper limit, and lower limit.

The OUTPFAIP? command in line 470 returns the number of failed points. (When the limit test result is PASS, it returns 0 and the program goes to Passed.) Then the array Dt is printed with as many lines as the transferred data.

Lines 510 to 560 print the limit test result as follows:

```
Swp.Prmtr(Hz) : 1.1925E+7
   Result          Upper          Lower
     0              20             -40

Swp.Prmtr(Hz) : 1.2125E+7
   Result          Upper          Lower
     0              20             -40
```

What are other commands used to retrieve the test results?

Instead of reading the limit test results for failed points by using the OUTPLIMF? command, you can read out the test result using the following commands:

- At all measurement points: OUTPLIML?
- At marker position: OUTPLIMM?

Both commands return the sweep parameter, result (1 for pass, 0 for fail, −1 for no test), upper limit, and lower limit.

9

Using the List Sweep Function

This chapter describes how to use the list sweep function over GPIB.

To Set List Sweep

```
10   !
20   ! Figure 9-1. List Sweep
30   !
40   ASSIGN @Hp4396 TO 800  ! When iBASIC is used, change "717" to "800"
50   !
60   OUTPUT @Hp4396;"SA"
70   CLEAR SCREEN
80   PRINT "Segment";TAB(9);"Center(Hz)";TAB(20);"Span(Hz)";TAB(30);"Points";
90   PRINT TAB(39);"Power(dBm)";TAB(50);"RBW(Hz)"
100  !
110  DIM Table(1:31,1:5)
120  INPUT "Enter number of segments (<=31)",Numb
130  FOR I=1 TO Numb
140    GOSUB Loadlist
150  NEXT I
160    !
170  LOOP
180    INPUT "Do you want to edit? (Y/N)",An$
190  EXIT IF An$="N" OR An$="n"
200    INPUT "Enter segment number(<=31)",I
210    IF Numb<I THEN Numb=I
220    GOSUB Loadlist
230  END LOOP
240   !
250  OUTPUT @Hp4396;"EDITLIST"
260  OUTPUT @Hp4396;"CLEL"
270  FOR K=1 TO Numb
280    OUTPUT @Hp4396;"SADD"
290    OUTPUT @Hp4396;"CENT ";Table(K,1)
300    OUTPUT @Hp4396;"SPAN ";Table(K,2)
310    OUTPUT @Hp4396;"POIN ";Table(K,3)
320    OUTPUT @Hp4396;"POWE ";Table(K,4)
330    OUTPUT @Hp4396;"BW ";Table(K,5)
340    OUTPUT @Hp4396;"SDON"
350  NEXT K
360  OUTPUT @Hp4396;"EDITDONE"
370  OUTPUT @Hp4396;"SWPT LIST"
380 !
```

Figure 9-1. Sample Program : List Sweep (1/2)

```
390  INPUT "Connect DUT, and press Enter.",Dum$
400  OUTPUT @Hp4396;"CLES"
410  OUTPUT @Hp4396;"*SRE 4;ESNB 1"
420  ON INTR 8 GOTO Sweep_end  ! \ When iBASIC is used, change "7" to "8"
430  ENABLE INTR 8;2           ! /
440    OUTPUT @Hp4396;"SING"
450 Measuring:GOTO Measuring
460 Sweep_end:   !
470 DISP "Program End"
480  STOP
490  !
500 Loadlist: !
510  INPUT "Enter center frequency(Hz)",Table(I,1)
520  INPUT "Enter frequency span(Hz)",Table(I,2)
530  INPUT "Enter number of points",Table(I,3)
540  INPUT "Enter power level(dBm)",Table(I,4)
550  INPUT "Enter resolution band width(Hz)",Table(I,5)
560  PRINT I;TAB(11);Table(I,1);TAB(20);Table(I,2);TAB(30);Table(I,3);
570  PRINT TAB(40);Table(I,4);TAB(50);Table(I,5)
580  RETURN
590  END
```

Figure 9-1. Sample Program : List Sweep (2/2)

Edit List Table

```
70   CLEAR SCREEN
80   PRINT "Segment";TAB(9);"Center(Hz)";TAB(20);"Span(Hz)";TAB(30);"Points";
90   PRINT TAB(39);"Power(dBm)";TAB(50);"RBW(Hz)"
100  !
110  DIM Table(1:31,1:5)
120  INPUT "Enter number of segments (<=31)",Numb
130  FOR I=1 TO Numb
140    GOSUB Loadlist
150  NEXT I
160    !
170  LOOP
180    INPUT "Do you want to edit? (Y/N)",An$
190  EXIT IF An$="N" OR An$="n"
200    INPUT "Enter segment number(<=31)",I
210    IF Numb<I THEN Numb=I
220    GOSUB Loadlist
230  END LOOP
 :
500 Loadlist: !
510  INPUT "Enter center frequency(Hz)",Table(I,1)
520  INPUT "Enter frequency span(Hz)",Table(I,2)
530  INPUT "Enter number of points",Table(I,3)
540  INPUT "Enter power level(dBm)",Table(I,4)
550  INPUT "Enter resolution band width(Hz)",Table(I,5)
560  PRINT I;TAB(11);Table(I,1);TAB(20);Table(I,2);TAB(30);Table(I,3);
570  PRINT TAB(40);Table(I,4);TAB(50);Table(I,5)
580  RETURN
```

Lines 70 to 90 print the list table heads on the BASIC screen.

Line 110 defines the table (array `Table(1:31,1:5)`) used to hold the list values. It contains the center frequency, frequency span, number of points, power level, and resolution band width of each segment as follows:

Segment	**Center Frequency**	**Frequency Span**	**Number of Points**	**Power Level**	**Resolution Band Width**
1	`Table(1,1)`	`Table(1,2)`	`Table(1,3)`	`Table(1,4)`	`Table(1,5)`
2	`Table(2,1)`	`Table(2,2)`	`Table(2,3)`	`Table(2,4)`	`Table(2,5)`
⋮	⋮	⋮	⋮	⋮	⋮

Lines 130 to 150 call the subroutine `Loadlist` (line 500) to edit and print as many segments as you defined in line 120 (The analyzer can retain up to 31 segments).

```
Segment   Center(Hz)   Stop(Hz)   Points   Power(dBm)   RBW(Hz)
 1           100          20        100        0          100
 2           10000        1000      300        0          300
 3           1000000      1000      400        0          100
```

The loop, lines 170 to 230, determines if you want to edit the table and confirms that the segment is in the table.

Set List Table

```
250  OUTPUT @Hp4396;"EDITLIST"                 Start to set the list table
260  OUTPUT @Hp4396;"CLEL"                     Clear existing limit table
270  FOR K=1 TO Numb                                                          Setting the
280    OUTPUT @Hp4396;"SADD"                   Start to set a segment          list table
290    OUTPUT @Hp4396;"CENT ";Table(K,1)       Enter center frequency
300    OUTPUT @Hp4396;"SPAN ";Table(K,2)       Enter frequency span         Setting a
310    OUTPUT @Hp4396;"POIN ";Table(K,3)       Enter number of points        segment
320    OUTPUT @Hp4396;"POWE ";Table(K,4)       Enter power level
330    OUTPUT @Hp4396;"BW ";Table(K,5)         Enter resolution band width
340    OUTPUT @Hp4396;"SDON"                   Complete the segment
350  NEXT K
360  OUTPUT @Hp4396;"EDITDONE"                 Complete the list table
370  OUTPUT @Hp4396;"SWPT LIST"                Activate list sweep
```

In this portion of the program, the list table (edited using BASIC) is transferred to the analyzer.

What are other commands are used to set the list values?

- When setting segment frequencies, instead of setting the center/span definition by using the `CENT` *parameter* / `SPAN` *parameter* commands, you can define start/stop frequency by using:

  1. `STAR` *parameter* / `STOP` *parameter* commands
  2. `MKRSTAR` / `MKRSTOP` commands (Maker to start/stop)

- When setting the IF band width (with the analyzer in network analyzer mode), use the `BW` *parameter* command.

10

Using the Analyzer's I/O Port

This chapter describes how to use the I/O port of the analyzer with the GPIB. For general operation of the I/O port, see the *4396B Function Reference* manual.

The I/O port on the analyzer's rear panel communicates with external devices (for example, a handler on a production line).

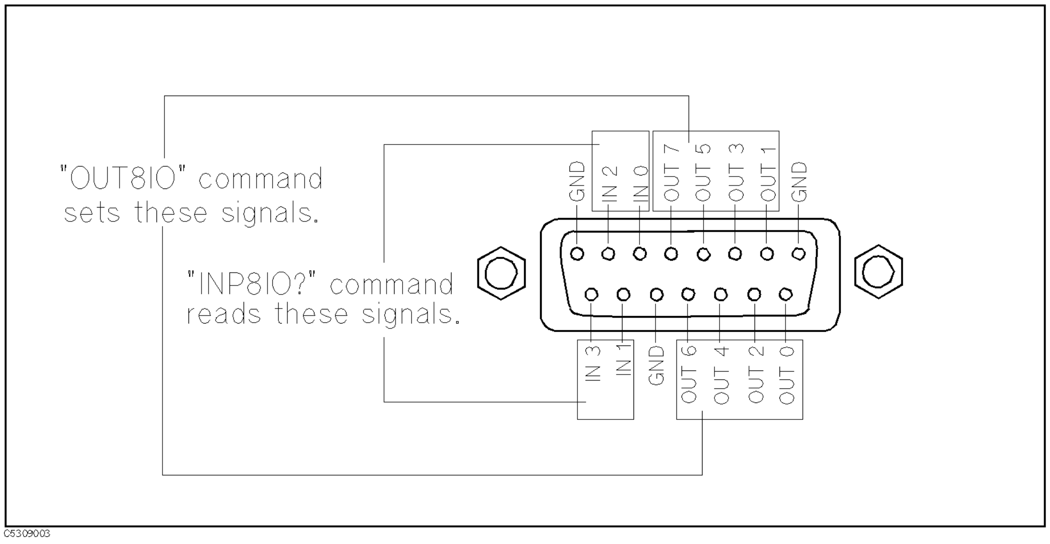

"OUT8IO" command sets these signals.
"INP8IO?" command reads these signals.
GND
IN 2
IN 0
OUT 7
OUT 5
OUT 3
OUT 1
GND
IN 3
IN 1
GND
OUT 6
OUT 4
OUT 2
OUT 0
C5309003

Figure 10-1. I/O Port

The I/O port consists of the following 15 TTL compatible signals:

8-bit output
4-bit input
3 grounds

The signals IN 0 to IN 3 and OUT 0 to OUT 7 can be read and set by GPIB commands.

iBASIC

The Instrument BASIC commands `READIO(15,0)` and `WRITEIO 15,0` can directly control the 8-bit I/O port without using GPIB commands. This operation is faster than using an GPIB command. For more information on these commands, see the *Using Instrument BASIC with the 4396B* manual.

To Synchronize External Handler with Analyzer

```
10 !
20 ! Figure 10_2. Synchronization of an External Handler
30 !             with the Analyzer
40 !
50  ASSIGN @Hp4396 TO 717 ! When iBASIC is used, change "717" to "800"
60  !
70  !
80  OUTPUT @Hp4396;"OUT8IO 8"
90  !
100 REPEAT                       !
110   OUTPUT @Hp4396;"INP8IO?"    !
120   ENTER @Hp4396;Inpio         ! Waiting Handler Response
130   A=BIT(Inpio,3)              !
140 UNTIL A=1                    !
150 !
160 !
170 END
```

**Figure 10-2.
Sample Program : Synchronization of an External Handler with the Analyzer**

Send Signal to the External Handler

```
80 OUTPUT @Hp4396;"OUT8IO 8"
```

The OUT8IO *parameter* command sets the 8-bit data value of the OUT 0-7 lines. The OUT8IO 8 command sets the OUT 3 line to TRUE (1).

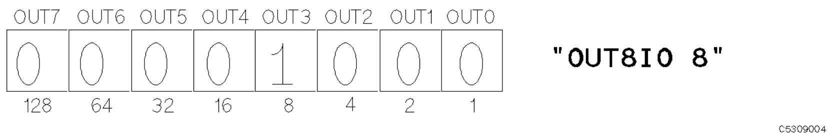


Figure 10-3. 8-Bit Data of OUT0-7

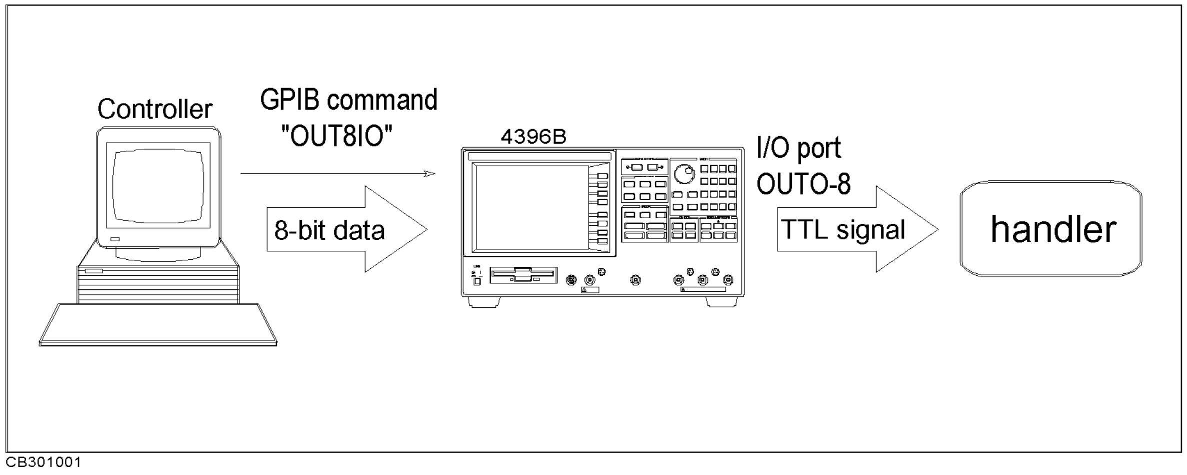


Figure 10-4. Sending Signal to an the External Handler

Read Signal from the External Handler

```
100 REPEAT                          !
110   OUTPUT @Hp4396;"INP8IO?"      !
120   ENTER @Hp4396;Inpio           ! Waiting Handler Response
130   A=BIT(Inpio,3)                !
140 UNTIL A=1                       !
```

The INP8IO? command returns the 4-bit data value of the IN 0-3 lines.

Lines 100 to 160 wait for the external handler to set signal on line IN 3 to TRUE (1).

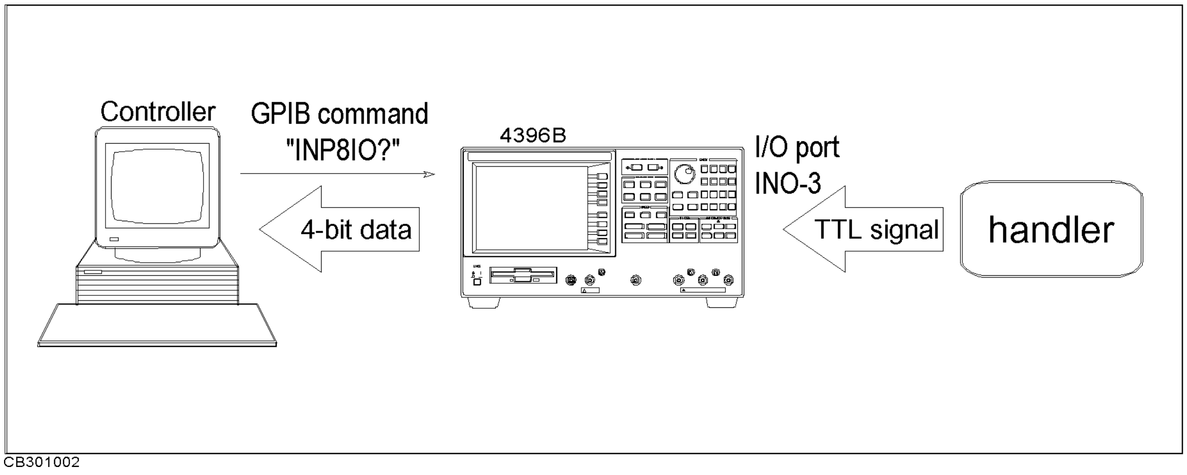


Figure 10-5. Reading Signal from the External Handler

11

Using Application Programs

This chapter provides the following sample programs:

- For your convenience when calculating the following spectrum analysis factors:
  - Total Harmonic Distortion (THD)
  - Adjacent Channel Power
  - Occupied Bandwidth
- For your convenience when using the file transfer function
  - File transfer from the 4396B to the external controller
  - File transfer from the external controller to the 4396B
  - Listing of the files in the current directory of the 4396B

Total Harmonic Distortion

Most transmitting devices and signal sources contain harmonics as shown in Figure 11-1.

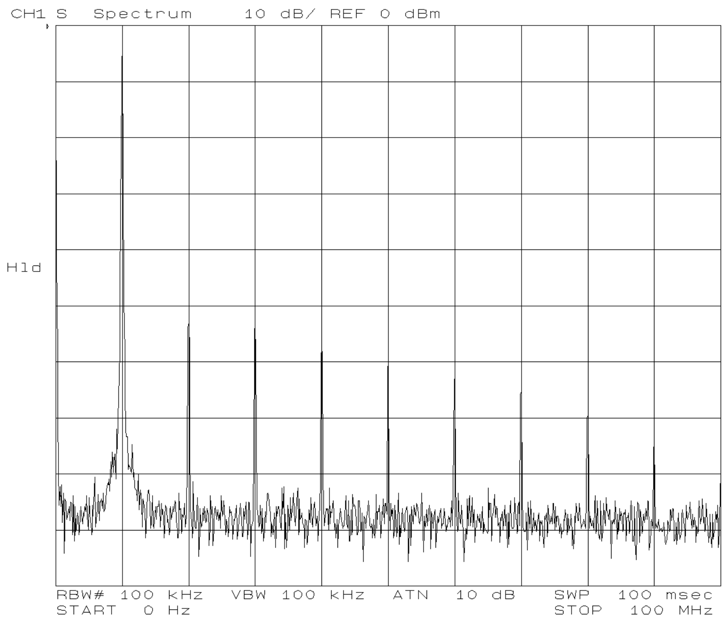


Figure 11-1. Harmonic Distortion in a Signal

This program computes the total harmonic distortion (THD) as defined by the following equation:

$$THD = \frac{\sqrt{V_2^2 + V_3^2 + \cdots}}{V_1} \times 100[\%] \qquad (11-1)$$

Where,

V_1	Fundamental [V]
V_2	The second harmonic [V]
V_3	The third harmonic [V]
$\vdots$	

THD takes into account the power in all the harmonics. Because an infinite number of the harmonics cannot be measured, a finite number will have to suffice.

Before running the program, measure the signal and display the fundamental and harmonics to be computed on the analyzer display (in the dBm format).

```
10   !
20   ! Figure 11-2. Total Harmonic Distortion
30   !
40     Vf=1
50     ASSIGN @Hp4396 TO 717  ! When iBASIC is used, replace "717" to "800"
60     OUTPUT @Hp4396;"CLES;*SRE 4;ESNB 96"
70     ON INTR 7 GOTO Done   ! \ When iBASIC is used,
80     ENABLE INTR 7;2       ! / replace "7" to "8"
90     OUTPUT @Hp4396;"STOP?"
100    ENTER @Hp4396;Fstop
110    OUTPUT @Hp4396;"PRSMKRS"
120    OUTPUT @Hp4396;"MKR ON;SEAM PEAK"
130    OUTPUT @Hp4396;"OUTPMKR?"
140    ENTER @Hp4396;Vf,Vf2,Ff    ! Fundamental
150    Vf=SQR(10^(Vf/10)*.05)     ! Vf in V
160    PRINT "Fundamental"
170    Fr=Ff
180    I=2
190    S=0
200  LOOP
210    Fh=Ff*I
220  EXIT IF Fstop<=Fh
230    OUTPUT @Hp4396;"DMKR TRAC;MKRPRM ";Fh-Fr/2
240    OUTPUT @Hp4396;"DMKR ON"
250    OUTPUT @Hp4396;"MKRPRM ";Fr
260    OUTPUT @Hp4396;"PARS ON;SEARSTR"
270    OUTPUT @Hp4396;"SEAM PEAK;DMKR OFF"
280    OUTPUT @Hp4396;"OUTPMKR?"
290    ENTER @Hp4396;Vh,Vh2,F
300    Vh=10^(Vh/10)*.05          ! Vh^2 in V^2
310    PRINT I;" harmonic"
```

Figure 11-2. Sample Program : Total Harmonic Distortion (THD) (1/2)

```
320     S=S+Vh
330     I=I+1
340   END LOOP
350  !
360 Done:   !
370    Thd=SQR(S)/Vf*100
380    PRINT "THD=";Thd;" %"
390    DISP "PROGRAM FINISHED"
400    END
```

Figure 11-2. Sample Program : Total Harmonic Distortion (THD) (2/2)

In line 120 the marker searches for the fundamental frequency.

In lines 200 to 340 the marker searches for the harmonics on the analyzer display and integrates the squares.

Line 370 calculates the THD and line 380 prints the result.

Adjacent Channel Power Calculation

The adjacent channel power measurement examines the leakage power transmitted into an adjacent channel (that is, the channel next to the carrier channel). This program calculates the adjacent channel power leakage ratio to the power of the transmitter (Pl−Pc, Ph−Pc) in dBc.

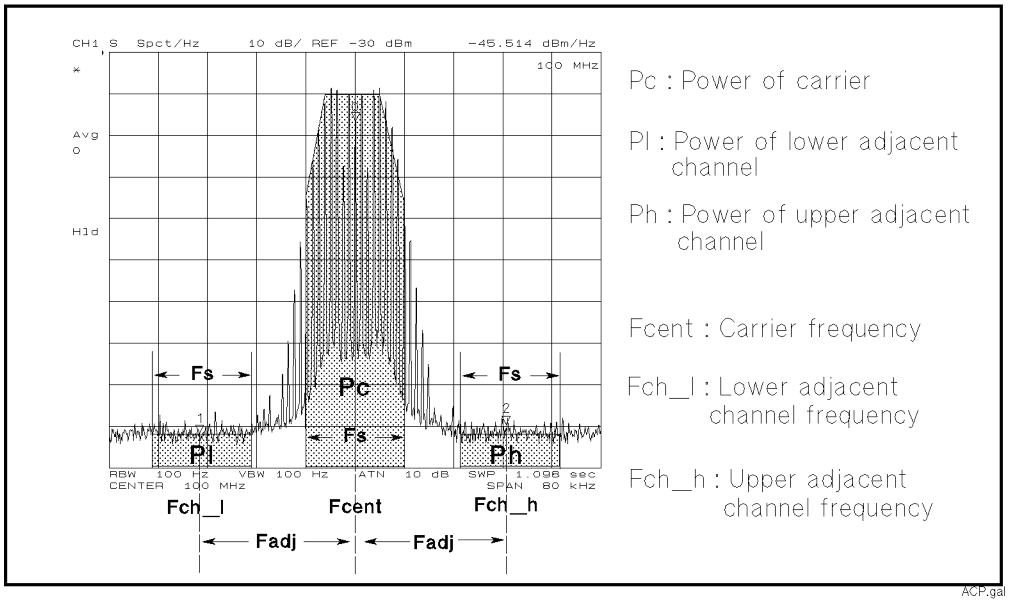


Figure 11-3. Adjacent Channel Power

Before running the program, set up the measurement, calibrate the analyzer, and connect the signal to the input port.

```
10 !
20 ! Figure 11-4. Adjacent Channel Power Calculation
30 !
40   ASSIGN @Hp4396 TO 717   ! When iBASIC is used, change "717" to "800"
50   Fadj=25000              ! Hz
60   Fs=16000                ! Hz
70   Rbw=100                 ! Resolution bandwidth, Hz
80   Nop=801                 ! Number of measurement points
90   Fspan=80000             ! Frequency Span, Hz
100  Avg=10                  ! Averaging factor
110  !
120  CLEAR SCREEN
130  INPUT "Enter carrier frequency(Hz).",Fcent
140  OUTPUT @Hp4396;"CENT ";Fcent
150  OUTPUT @Hp4396;"SPAN ";Fspan
160  OUTPUT @Hp4396;"BW ";Rbw
170  OUTPUT @Hp4396;"AVERFACT ";Avg
180  OUTPUT @Hp4396;"FMT NOISE;SAUNIT DBM;ATTAUTO ON;AVER ON"
190  OUTPUT @Hp4396;"HOLD;AVERREST"
200  !
210    INPUT "Connect input port and press Enter.",Dum$
220    DISP "MEASURING"
230    OUTPUT @Hp4396;"TRGS INT"
240    OUTPUT @Hp4396;"CLES"
250    OUTPUT @Hp4396;"*SRE 4;ESNB 1"
```

Figure 11-4. Sample Program : Adjacent Channel Power Calculation (1/2)

```
260     ON INTR 7 GOTO Sweep_end       ! \ When iBASIC is used,
270     ENABLE INTR 7;2                ! / change "7" to "8"
280       OUTPUT @Hp4396;"NUMG ";Avg
290 Measuring:GOTO Measuring
300 Sweep_end:     !
310     DISP "MEASUREMENT COMPLETE"
320     DIM D(1:801)
330     OUTPUT @Hp4396;"FORM3"
340     ASSIGN @Dt TO 717;FORMAT OFF   ! When iBASIC is used,
350     OUTPUT @Hp4396;"OUTPDTRC?"     ! change "717" to "800"
360     ENTER @Dt USING "%,8A";Dum$
370     ENTER @Dt;D(*)
380     ENTER @Dt USING "%,1A";Dum$
390     !
400   Fch_l=Fcent-Fadj
410   Fch_h=Fcent+Fadj
420   Pc=FNPower(D(*),Fspan,Fcent,Fs,Nop,Fcent)
430   Pl=FNPower(D(*),Fspan,Fch_l,Fs,Nop,Fcent)
440   Ph=FNPower(D(*),Fspan,Fch_h,Fs,Nop,Fcent)
450   !
460   OUTPUT @Hp4396;"MKR ON;SMKR1 ON;SMKR2 ON"
470   OUTPUT @Hp4396;"MKRPRM ";Fcent
480   OUTPUT @Hp4396;"SMKRPRM1 ";Fch_l
490   OUTPUT @Hp4396;"SMKRPRM2 ";Fch_h
500   PRINT "Carrier (MHz):",Fcent/1.E+6
510   PRINT "Power   (dBm):",Pc
520   PRINT
530   PRINT "Adjacent Channel Freq. Lo(Hz):",Fch_l
540   PRINT "                       Hi(Hz):",Fch_h
550   PRINT
560   PRINT "Adjacent Pow. Pl-Pc(dBc):",Pl-Pc
570   PRINT "              Ph-Pc(dBc):",Ph-Pc
580   DISP "PROGRAM FINISHED"
590   END
600   !
610   DEF FNPower(D(*),Fspan,Fch,Fs,Nop,Fcent)
620     Fdelta=Fspan/(Nop-1)
630     Ich=(Fch-Fcent)/Fdelta+401
640     I1=Ich-Fs/2/Fdelta
650     I2=Ich+Fs/2/Fdelta
660     IF I1<1 OR I2>Nop THEN
670       P=0
680       RETURN P
690     END IF
700     S=0
710     FOR I=I1 TO I2
720       S=S+10^(D(I)/10)          !  S in mW
730     NEXT I
740     P=S*(Fspan/(Nop-1))
750     P=10*LGT(P)                 ! P in dBm
760     RETURN P
770   FNEND
```

Figure 11-4. Sample Program : Adjacent Channel Power Calculation (2/2)

Lines 50 to 100 set the measurement coefficient, Fadj, Fs, frequency span, resolution bandwidth, number of measurement point, and averaging factor to typical values.

In lines 610 to 760, the subprogram `FNPower` performs summation of the powers by measurement points, in the area of center frequency, `Fch`, and the frequency span, `Fs`. The total power is calculated by the following equation:

$$PWR = 10\log_{10}\sum_{x=I1}^{x=I2}\frac{D(x)Fspan}{Nop}\ [\mathrm{mW}] \qquad (11-2)$$

Where,

$D(x)$	Power density spectrum [dBm/Hz]
$Fspan$	Measurement frequency span [Hz]
Nop	Number of measurement points
I1, I2	The measurement point of left and right edge of a channel bandwidth(Fs)

Occupied Power Bandwidth Calculation

This program calculates the occupied bandwidth of the carrier signal. It first computes the combined power of all signal responses contained in the trace. It then calculates the point for which 0.5 % of the total power lies to the right of the right maker and to the left of the left marker.

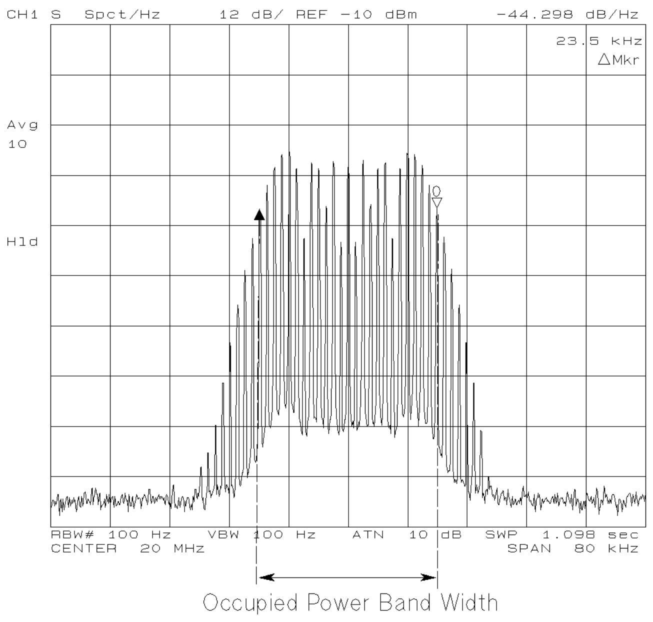


Figure 11-5. 99 % Occupied Power Bandwidth

Before running the program, set up the measurement, calibrate the analyzer, and connect the signal to the input port.

```
10 !
20 ! Figure 11-6. Occupied Power Bandwidth Calculation
30 !
40   ASSIGN @Hp4396 TO 717 ! When iBASIC is used, change "717" to "800"
50   Rbw=100         ! Resolution bandwidth, Hz
60   Nop=801         ! Number of measurement points
70   Fspan=80000.    ! Frequency Span, Hz
80   Avg=10          ! Averaging factor
90   !
100  INPUT "Enter carrier frequency(Hz).",Fcent
110  OUTPUT @Hp4396;"SPAN";Fspan
120  OUTPUT @Hp4396;"CENT";Fcent
130  OUTPUT @Hp4396;"BW";Rbw
140  OUTPUT @Hp4396;"FMT NOISE;DET POS"
150  OUTPUT @Hp4396;"SAUNIT DBM;ATTAUTO ON;AVER ON"
160  OUTPUT @Hp4396;"AVERFACT";Avg
170  !
180  INPUT "Connect input port and press Enter.",Dum$
190  OUTPUT @Hp4396;"HOLD"
200  OUTPUT @Hp4396;"CLES"
210  OUTPUT @Hp4396;"*SRE 4;ESNB 1"
220  ON INTR 7 GOTO Sweep_end     ! \ When iBASIC is used,
230  ENABLE INTR 7;2              ! / change "7" to "8"
240    OUTPUT @Hp4396;"TRGS INT"
250    OUTPUT @Hp4396;"NUMG";Avg
260 Measuring:GOTO Measuring
270 Sweep_end:  !  Get Data
280  DIM D(1:801)
290  DIM P(1:801)
300  OUTPUT @Hp4396;"FORM3"
310  ASSIGN @Dt TO 717;FORMAT OFF ! When iBASIC is used,
320  OUTPUT @Hp4396;"OUTPDTRC?"   ! change "717" to "800"
330  ENTER @Dt USING "%,8A";Dum$
340  ENTER @Dt;D(*)
350  ENTER @Dt USING "%,1A";Dum$
360  !
370  Power(D(*),P(*),Rbw,Nop,Fspan)
380  !
390  FOR I=1 TO Nop
400    A=P(I)/P(Nop)
410    IF A>.005 THEN Lower
420  NEXT I
430 Lower:I1=I
440  FOR I=Nop TO 1 STEP -1
450    A=P(I)/P(Nop)
460    IF A<.995 THEN Upper
470  NEXT I
480 Upper:I2=I
```

Figure 11-6. Sample Program : Occupied Power Bandwidth Calculation (1/2)

```
490     OUTPUT @Hp4396;"MKR ON"
500     OUTPUT @Hp4396;"MKRP ";I1
510     OUTPUT @Hp4396;"DMKR ON"
520     OUTPUT @Hp4396;"MKRP ";I2
530     OUTPUT @Hp4396;"OUTPMKR?"
540     ENTER @Hp4396;Val,Val2,Flh
550     PRINT "Occupied bandwidth :";
560     PRINT Flh;" Hz"
570   DISP "PROGRAM FINISHED"
580   END
590   !
600   SUB Power(D(*),P(*),Rbw,Nop,Fspan)
610     S=0
620     FOR I=1 TO Nop
630       S=S+10^(D(I)/10)     !  S in mW
640       P(I)=S
650     NEXT I
660   SUBEND
```

Figure 11-6. Sample Program : Occupied Power Bandwidth Calculation (2/2)

Lines 40 to 80 set the measurement coefficient, frequency span, resolution bandwidth, number of measurement points, and averaging factor to typical values.

Lines 390 to 430 search from the left for the point where the power is 0.5 % compared to the total power. Lines 440 to 480 do the same search from the right.

Lines 490 to 540 display the marker and Δ marker on the 0.5 % power point and read out the spacing of the markers.

Lines 600 to 660 (subprogram `Power`) perform a summation of the power at the measurement points. This summation is done in the area of the center frequency (`Fch`) and the frequency span (`Fs`). The same equation (11-2) is used in the "Adjacent Channel Power Calculation" example.

File Transfer Function

This section describes how to use the file transfer function, showing you a sample program.

The file transfer function uses the external controller to transfer files between the selected storage device of this instrument (memory disk or diskette) and an external storage device (such as hard disk). This function allows you to:

- Directly access data you want to use on the external controller.

  For example, you can transfer the file of an instrument screen to the external controller, print it on a printer connected to the external controller, and paste it onto a file in a word processor running on the external controller.

- Use external storage devices, which have larger capacity compared to the memory disk or a diskette.

  For example, if there are a great number of measurement conditions which require calibration, the amount of the setting data becomes extremely large, including calibration data. In this case, it is impractical to store all of these settings on the memory disk or a single diskette at a time. However, you can realize this functionality by transferring them to the external controller and then storing them on an external storage device.

- Perform remote measurement using the external controller with a few GPIB commands for basic measurement. You do not have to memorize further details (such as GPIB commands used for detailed settings).

  Preparation:

    Use the keys on the front panel to establish the setting required for your measurement. Store it on the storage device of the 4396B, then transfer the file to the external controller, and store it on an external storage device. Repeat this procedure for all of the settings required for your measurement.

  Measurement:

    Choose a necessary setting file from those stored and transfer it to the 4396B using the external controller. Then, recall the file to set the 4396B for the measurement and perform the measurement using the GPIB commands.

The storage device of the 4396B allows you to handle files listed below in the DOS format or the LIF format. For DOS format files, both binary files and ASCII files can be transferred. For LIF format files, only binary files can be transferred.

- Binary files
  - Instrument settings and internal data array (STATE)
  - Internal data arrays (DATA ONLY binary)
  - Graphic images (GRAPHICS)
- ASCII files
  - Internal data arrays (DATA ONLY ascii)
  - Instrument BASIC programs

File Transfer from 4396B to External Controller

This program transfers a specified file in the current directory of the 4396B to the current directory of the storage device connected to the external controller, giving a file name you desire.

When executed, this program first prompts you to enter a source file name, as shown below. Enter the name of a file you want to transfer.

```
ENTER SOURCE FILE NAME ON INSTRUMENT ?
```

Then, the program prompts you to enter a destination file name as shown below(in this example, SAMPLE.STA has been entered as the source file name). Enter the file name you want to give on the storage device. Note that a file with the same name will be overwritten, if it already exists.

```
ENTER SOURCE FILE NAME ON INSTRUMENT ?        SAMPLE.STA
ENTER DESTINATION FILE NAME ON CONTROLLER ?
```

```
10    !
20    !  Figure 11-7. File transfer (Instrument -> Controller)
30    !
40    DIM Src_file$[50],Dst_file$[50]
50    ASSIGN @Hp4396 TO 717
60    OUTPUT @Hp4396;"*rst"
70    !
80    PRINT "  ENTER SOURCE FILE NAME ON INSTRUMENT ?        ";
90    INPUT Src_file$
100   PRINT Src_file$
110   !
120   PRINT "  ENTER DESTINATION FILE NAME ON CONTROLLER ?   ";
130   INPUT Dst_file$
140   PRINT Dst_file$
150   !
160   Copy_from_instr(@Hp4396,Src_file$,Dst_file$)
170   !
180   END
190   !
200   !  copy_from_instrument
210   !
220   SUB Copy_from_instr(@Hp4396,Src_file$,Dst_file$)
230       DIM Len$[6],Img$[32],Dmy$[2]
240       !
250       ON ERROR GOTO Skip_purge
260       PURGE Dst_file$
270 Skip_purge:  OFF ERROR
280       CREATE Dst_file$,1
290       ASSIGN @Dst_file TO Dst_file$
300       !
```

Figure 11-7. Sample Program: File Transfer from 4396B to External Controller (1/2)

```
310      CLEAR @Hp4396
320      OUTPUT @Hp4396;"CLES"
330      OUTPUT @Hp4396;"ROPEN """;Src_file$;""""
340      IF FNCheck_error(@Hp4396,"<CPFI: ropen>")=-1 THEN SUBEXIT
350      !
360      LOOP
370          OUTPUT @Hp4396;"READ?"
380          ENTER @Hp4396 USING "#,2A";Dmy$
390          ENTER @Hp4396 USING "#,6A";Len$
400          Block_size=VAL(Len$)
410          !
420          IF Block_size=0 THEN
430              ENTER @Hp4396 USING "%,A";Dmy$
440              ASSIGN @Dst_file TO *
450              OUTPUT @Hp4396;"CLOSE"
460              SUBEXIT
470          END IF
480          !
490          ALLOCATE Dat$[Block_size]
500          Img$="#,"&VAL$(Block_size)&"A"
510          ENTER @Hp4396 USING Img$;Dat$
520          ENTER @Hp4396 USING "%,A";Dmy$
530          OUTPUT @Dst_file USING Img$;Dat$
540          DEALLOCATE Dat$
550          !
560          IF FNCheck_error(@Hp4396,"<CPFI: read>")=-1 THEN SUBEXIT
570      END LOOP
580  SUBEND
590  !
600  !  Instrument Error Check
610  !
620  DEF FNCheck_error(@Hp4396,Str$)
630      DIM Err$[64]
640      OUTPUT @Hp4396;"OUTPERRO?"
650      ENTER @Hp4396;Err$
660      IF Err$"+0,""No error""" THEN
670          PRINT "ERROR: ";Str$;"   ";Err$
680          RETURN -1
690      ELSE
700          RETURN 0
710      END IF
720  FNEND
```

Figure 11-7. Sample Program: File Transfer from 4396B to External Controller (2/2)

Lines 80 to 140 accept the entry of the source file name and the destination file name.
Line 160 calls the subprogram to transfer a file from the 4396B to the external controller.
Lines 250 to 290 prepare for writing to the destination file.
Lines 310 to 340 prepare for reading the source file to the external controller.
Line 370 executes the query command to read data.
Lines 380 to 400 read the part indicating the length of the fixed length block data (see Figure 11-8) to obtain the length of the data to be transferred.
Lines 420 to 470 check the data length. If the data length is 0, the transfer process is terminated.

Depending on the data length obtained in lines 500 to 520, the program adjusts the format and reads the data part.
Line 530 writes the data to the destination file.
The maximum length of data transferred at a time is 16 Kbytes. Therefore, if the size of the source file is greater than 16 Kbytes, the transfer routine, lines 370 to 560, is repeated until transferring all of the data is completed.
Lines 620 to 720 provide a function to check that no error has occurred in the 4396B.

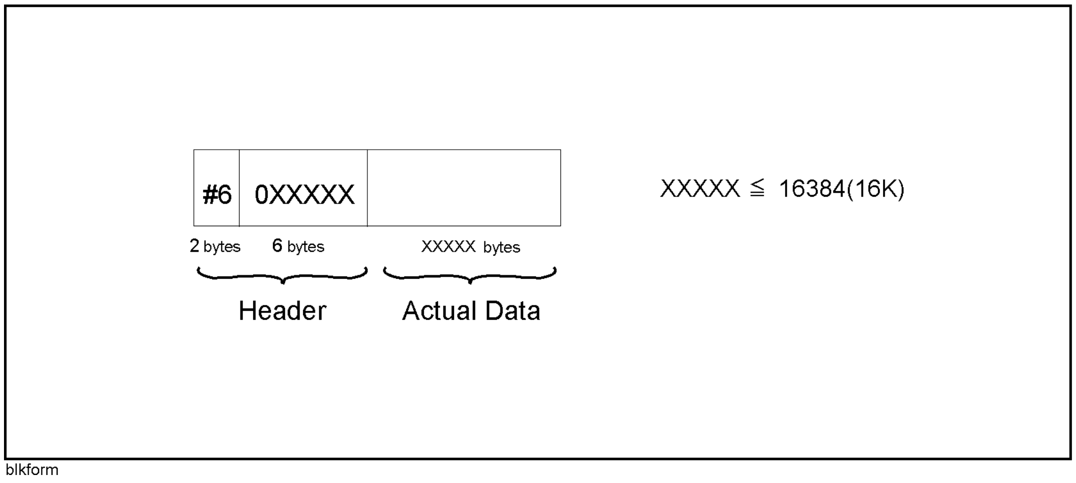


Figure 11-8. Fixed length block format

File Transfer from External Controller to 4396B

This program transfers a specified file in the current directory of the storage device connected to the external controller to the current directory of the selected storage device of the 4396B, giving a file name you desire.

This program, when executed, first prompts you to enter a source file name, as shown below. Enter the name of a file you want to transfer.

```
ENTER SOURCE FILE NAME ON CONTROLLER ?
```

Next, the program prompts you to enter the size of the source file as shown below (in this example, SAMPLE.STA has been entered as the source file name). Enter the size correctly in bytes.

```
ENTER SOURCE FILE NAME ON INSTRUMENT ?        SAMPLE.STA
ENTER SOURCE FILE SIZE ?
```

Then, the program prompts you to enter the destination file name, as shown below (in this example, the size of SAMPLE.STA is 12288 bytes). Enter the file name you want to give on the destination storage device. Note that a file with the same name will be overwritten, if it already exists.

```
ENTER SOURCE FILE NAME ON INSTRUMENT ?        SAMPLE.STA
ENTER SOURCE FILE SIZE ?                      12288
ENTER DESTINATION FILE NAME ON CONTROLLER ?
```

```
10   !
20   !  Figure 11-8. File transfer (Controller -> Instrument)
30   !
40   DIM Src_file$[50],Dst_file$[50]
50   ASSIGN @Hp4396 TO 717
60   OUTPUT @Hp4396;"*rst"
70   !
80   PRINT "  ENTER SOURCE FILE NAME ON CONTROLLER ?        ";
90   INPUT Src_file$
100  PRINT Src_file$
110  !
120  PRINT "  ENTER SOURCE FILE SIZE ?                      ";
130  INPUT Src_size
140  PRINT Src_size
150  !
160  PRINT "  ENTER DESTINATION FILE NAME ON INSTRUMENT ?   ";
170  INPUT Dst_file$
180  PRINT Dst_file$
190  !
200  Copy_to_instr(@Hp4396,Src_file$,Src_size,Dst_file$)
210  !
220  END
230  !
240  !  copy_to_instrument
250  !
260  SUB Copy_to_instr(@Hp4396,Src_file$,Src_size,Dst_file$)
270      DIM Img$[32]
280      Max_bsize=16384
290      !
300      ASSIGN @Src_file TO Src_file$
310      !
320      CLEAR @Hp4396
330      OUTPUT @Hp4396;"CLES"
340      OUTPUT @Hp4396;"WOPEN """;Dst_file$;""""
350      IF FNCheck_error(@Hp4396," <CPTI: wopen>")=-1 THEN SUBEXIT
360      Xfr_done=0
370      !
380      LOOP
390          SELECT (Src_size-Xfr_done)
400              CASE >Max_bsize
410                  Block_size=Max_bsize
420              CASE 0
430                  ASSIGN @Src_file TO *
440                  OUTPUT @Hp4396;"CLOSE"
450                  SUBEXIT
460              CASE ELSE
470                  Block_size=(Src_size-Xfr_done)
```

Figure 11-9. Sample Program: File Transfer from External Controller to 4396B (1/2)

```
480          END SELECT
490          Xfr_done=Xfr_done+Block_size
500          !
510          ALLOCATE Dat$[Block_size]
520          !
530          Img$="#,"&VAL$(Block_size)&"A"
540          ENTER @Src_file USING Img$;Dat$
550          !
560          Img$="8A,ZZZZZZ,"&VAL$(Block_size)&"A"
570          OUTPUT @Hp4396 USING Img$;"WRITE #6",Block_size,Dat$,END
580          DEALLOCATE Dat$
590          IF FNCheck_error(@Hp4396," <CPTI: write>")=-1 THEN SUBEXIT
600      END LOOP
610  SUBEND
620  !
630  !  Instrument Error Check
640  !
650  DEF FNCheck_error(@Hp4396,Str$)
660      DIM Err$[64]
670      OUTPUT @Hp4396;"OUTPERRO?"
680      ENTER @Hp4396;Err$
690      IF Err$"+0,""No error""" THEN
700          PRINT "ERROR: ";Str$;"   ";Err$
710          RETURN -1
720      ELSE
730          RETURN 0
740      END IF
750  FNEND
```

Figure 11-8. Sample Program: File Transfer from External Controller to 4396B (2/2)

Lines 80 to 180 accept the entry of the source file name and its size and the destination file name.
Line 200 calls the subprogram to transfer a file from the external controller to the 4396B.
Lines 340 to 350 prepare for writing the file to the destination storage device.
Lines 390 to 480 calculate the length of the data that has not been transferred based on the source file size previously entered and the length of the data that has been already transferred. If the length of the remaining data does not exceed 16 Kbytes, it is set as the transfer data length; otherwise, 16 Kbytes is set as the transfer data length. Note that, if the length of the data not transferred is 0 at this time, the transfer process is terminated.
Lines 530 to 540 read data, whose amount is specified by the transfer data length, from the source file.
Lines 560 to 570 write data to the destination file in the fixed length block format (see Figure 11-8).
The maximum length of data transferred at a time is 16 Kbytes. Therefore, if the size of the source file is greater than 16 Kbytes, the transfer routine, lines 390 to 590, is repeated until transferring all of the data is completed.
Lines 650 to 750 provide a function to check that no error has occurred in the 4396B.

Note To transfer a file from the external storage device to the 4396B, you must check the file size (number of bytes) in advance .

Displaying List of Files in Current Directory

This program displays the list of the files in the current directory.

```
10   !
20   !  Figure 11-9. File list
30   !
40   ASSIGN @Hp4396 TO 717
50   OUTPUT @Hp4396;"*rst"
60   !
70   Dir_instr(@Hp4396)
80   !
90   END
100  !
110  !  Dir_instr
120  !
130  SUB Dir_instr(@Hp4396)
140      DIM Stor_dev$[5],Curr_dir$[50],File_name$[13]
150      !
160      OUTPUT @Hp4396;"STODMEMO?"
170      ENTER @Hp4396;A
180      IF A=1 THEN
190        Stor_dev$="MEMO"
200      ELSE
210        Stor_dev$="DISK"
220      END IF
230      OUTPUT @Hp4396;"CWD?"
240      ENTER @Hp4396;Curr_dir$
250      PRINT "["&Stor_dev$&"]: "&Curr_dir$
260      PRINT "Size[byte]    File Name"
270      PRINT "-----------------------"
280      OUTPUT @Hp4396;"FNUM?"
290      ENTER @Hp4396;File_count
300      IF File_count>=1 THEN
310          FOR I=1 TO File_count
320              OUTPUT @Hp4396;"FNAME? ";I
330              ENTER @Hp4396;File_name$
340              OUTPUT @Hp4396;"FSIZE? """&File_name$&""""
350              ENTER @Hp4396;File_size
360              PRINT USING "XX,DDDDDD,XXXX,K";File_size,File_name$
370          NEXT I
380      END IF
390  SUBEND
```

Figure 11-10. Sample Program: Displaying List of Files in Current Directory of 4396B

Line 70 calls the subprogram to display the list of the files in the current directory.
Lines 160 to 250 check the storage device currently selected and its current directory name, and then display the result.
Lines 280 to 290 check the number of the files in the current directory.
If there are any files in the current directory, lines 300 to 380 check the name and size of every file and display them.

The following is the output result of the program, assuming that the selected storage device is the memory disk and the current directory, \TEST, contains 2 files, FILE1.STA (size: 24576 bytes) and FILE2.TIF (size: 16384 bytes) and 1 directory, DIR1. For size of a directory, -1 is displayed. To view the list of the files in DIR1, use the CHAD command to change the current directory to DIR1 and then execute this program again.

```
[MEMO]: \TEST
Size[byte]    File Name
------------------------
      -1    ..\
      -1    DIR1\
   24576    FILE1.STA
   16384    FILE2.TIF
```

12

If You Have a Problem

This chapter provides the information you need to correct the listed problems.

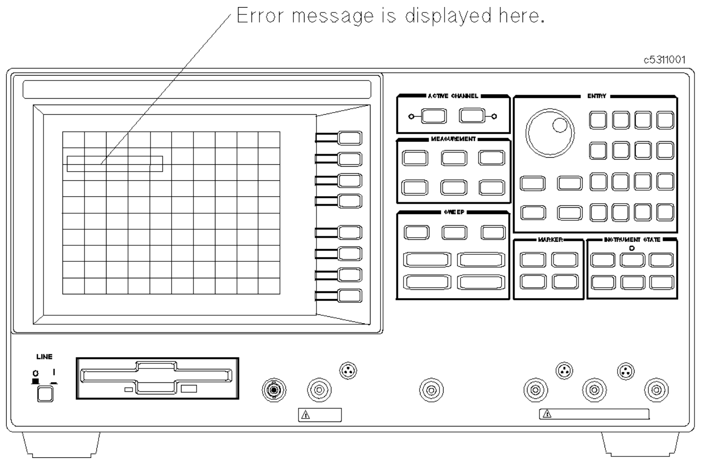
If There Is No Response From an Instrument on the GPIB Bus

- ☐ Check all GPIB addresses and cable connections.

  Most GPIB problems are caused by an incorrect address or a bad or loose GPIB cable.

If an Error Message is Displayed

- ☐ Check the error message on the analyzer's display.



- If "`GPIB error occurred`" is displayed:
  1. Get the error number and description using the `OUTPERRO?` command. (For information on how to use this command, see the "To Report Command Error Occurrence" in Chapter 3.)
  2. See "Messages" in the *GPIB Command Reference* manual.

- If any other message is displayed:

  See "Messages" in the *GPIB Command Reference* manual.

If the Disk Cannot Be Read

☐ Check the disk.

1. Put the disk into the disk drive and type as follows.

```
CAT
```

2. Press Return.

   If error message is displayed, the disk is corrupted or the disk format does not match. Use another disk.

COMPUTER

If you are using the external controller,

BASIC uses the LIF format, but doesn't use DOS format. Instrument BASIC uses both the LIF and the DOS format. Try again on using Instrument BASIC.

☐ Check the mass storage.

1. Put the disk into the disk drive and type as follows:

```
SYSTEM$("MSI")
```

2. Press Return.

```
:CS80, 700, 0      ←mass storage volume specifier
```

3. If the mass storage volume does not match your disk drive, use the `MSI` statement to set it to match.

iBASIC

If you are using Instrument BASIC:

To use the built-in disk drive, mass storage volume specifier must be `:,4`.

□ Check the file type.

1. Put the disk into the disk drive and type as follows:

```
CAT
```

2. Press (Return).

```
CAT
      ⋮
FILE NAME  PRO TYPE REC/FILE BYTE/REC   ADDRESS    DATE    TIME

FIG1_3        ASCII       6      256        34   22-Jun-92 11:00
FIG2_2        PROG        6      256        34   22-Jun-92 11:00
FIG2_3        ASCII       6      256        34   22-Jun-92 11:00
      ⋮
```

iBASIC

If you are using Instrument BASIC

Only an ASCII type program file can be saved and read.

Use the SAVE/GET commands to save and read ASCII files.

COMPUTER

If you are using BASIC

- To read ASCII type program, use GET command.

  The ASCII type program file can be saved and read using SAVE/GET commands.

- To read a PROG type program, use the READ command.

  The PROG type program file can be saved and read using the STORE/LOAD commands.

If the GPIB Command Does Not Work

□ Check the preceding GPIB command.

An GPIB command that requires execution time (such as changing format or calculating the calibration coefficients) can cause next GPIB command to fail.

If you are using such commands, insert the following command lines:

```
OUTPUT @Hp4396;"*OPC?"
ENTER @Hp4396;Dum
```

For details, see “To Wait For the Preceding Operation to Complete” in Chapter 3.

Manual Changes

Introduction

This appendix contains the information required to adapt this manual to earlier versions or configurations of the analyzer than the current printing date of this manual. The information in this manual applies directly to the 4396B Network/Spectrum Analyzer serial number prefix listed on the title page of this manual.

Manual Changes

To adapt this manual to your 4396B, see Table A-1 and Table A-2, and make all the manual changes listed opposite your instrument's serial number and firmware version.

Instruments manufactured after the printing of this manual may be different from those documented in this manual. Later instrument versions will be documented in a manual changes supplement that will accompany the manual shipped with that instrument. If your instrument's serial number is not listed on the title page of this manual or in Table A-1, it may be documented in a *yellow MANUAL CHANGES* supplement.

In additions to change information, the supplement may contain information for correcting errors (Errata) in the manual. To keep this manual as current and accurate as possible, Agilent Technologies recommends that you periodically request the latest *MANUAL CHANGES* supplement.

For information concerning serial number prefixes not listed on the title page or in the *MANUAL CHANGE* supplement, contact the nearest Agilent Technologies office.

Turn on the line switch or execute the `*IDN?` command by GPIB to confirm the firmware version. See the *GPIB Command Reference* manual for information on the `*IDN?` command.

Table A-1. Manual Changes by Serial Number

Serial Prefix or Number	Make Manual Changes

Table A-2. Manual Changes by Firmware Version

Version	Make Manual Changes
1.0X	Change 1

Serial Number

Agilent Technologies uses a two-part, nine-character serial number that is stamped on the serial number plate (see Figure A-1) attached to the rear panel. The first four digits and the letter are the serial prefix and the last five digits are the suffix.

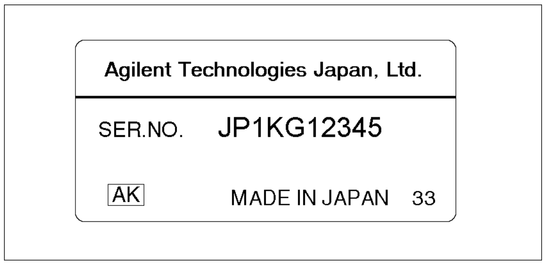


Figure A-1. Serial Number Plate

Change 1

The firmware revision 1.0X does not support the file transfer function. Please delete the descriptions about this function in this manual.

Index

A

active controller, 7-1
address, 1-2, 1-4
ASCII data transfer, 4-4
ASSIGN, 1-6
averaging, 2-3

B

binary data transfer, 4-6
BW, 9-3

C

CALIBRATION COEFFICIENT ARRAYS, 4-1, 4-5, 5-3, 5-5
CLEL, 9-3
CONT, 2-2

D

data arrays, 4-1
DATA ARRAYS, 4-1, 4-5, 5-5, 5-7
data format, 4-1
DATA TRACE ARRAYS, 4-1, 4-5, 5-5, 5-9
DATMEM, 5-5

E

EDITDONE, 9-3
EDITLIML, 8-3
EDITLIST, 9-3
ENABLE INTR, 3-4
ENTER, 1-8
*ESE, 3-5
ESNB, 3-3
*ESR?, 3-7
Event Status Enable Register, 3-5
Event Status Enable Register B, 3-3
Event Status Register B, 3-3

F

file transfer, 11-9
FMT, 5-5
FORM2, 4-7
FORM3, 4-7, 4-10
FORM4, 4-5
FORM5, 4-7
FORMAT OFF, 4-7, 4-10

G

group execution trigger, 2-4

H

HOLD, 2-2

I

INP8IO?, 10-3
INPDATA, 5-7
INPDTRC, 5-9
INPUCALC{1-12}, 5-5
INPUCALC{1-12}?, 5-3
INPUDATA, 5-5
INPUDTRC, 5-5
INPURAW{1-4}, 5-5
Instrument Event Status Register, 3-3
I/O port, 10-1

L

LIMCLEL, 8-3
LIMEDONE, 8-3
LIMILINE, 8-3
LIMITEST, 8-4
limit test, 8-1
LIML, 8-3
LIMPRM, 8-3
LIMSADD, 8-3
LIMSDON, 8-3
LIMU, 8-3
list sweep, 9-1

M

maker, 4-2
Manual Changes, A-1
MEMORY ARRAYS, 4-1, 4-5, 5-5
MEMORY TRACE ARRAYS, 4-1, 4-5, 5-5

O

ON INTR, 3-4
*OPC?, 3-2
OUT8IO, 10-2
OUTPCALC{1-12}?, 5-2
OUTPDATA?, 5-7
OUTPDTRC?, 4-5, 4-8, 4-10, 5-9
OUTPERRO?, 3-6
OUTPFAIP?, 8-4

OUTPLIMF?, 8-4
OUTPLIML?, 8-4
OUTPLIMM?, 8-4
OUTPMARK?, 4-3
OUTPSWPRM?, 4-5, 4-8, 4-10
OUTPUT, 1-4

P

pass control, 7-1
POIN, 9-3
POWE, 9-3
PRINALL, 6-1
PROGram:DEFine, 7-4
PROGram:DELete:ALL, 7-4
PROGram:EXECute, 7-3
PROGram[:SELected]:NUMBer, 7-6
PROGram subsystem, 7-3

Q

query, 1-8

R

RAW DATA ARRAYS, 4-1, 4-5, 5-5
remote mode, 1-4

S

SADD, 9-3
SAUNIT, 5-5
SAVC, 5-5
SDON, 9-3
select code, 1-4
serial number, A-2
Service Request Enable Register, 3-3, 3-5
service request (SRQ), 3-1
SING, 2-3
SPOLL, 3-7
*SRE, 3-3, 3-5
SRQ, 3-1
Standard Event Status Register, 3-5
Status Byte Register, 3-3, 3-5
SWPT, 9-3
system controller, 7-1

T

*TRG, 2-4
TRGS BUS, 2-4
TRGS INT, 2-2
TRIGGER, 2-4
trigger source, 2-1, 2-2
trigger system, 2-1

Agilent 4396B Network/Spectrum/Impedance Analyzer

Using Instrument BASIC with the 4396B

Agilent Technologies

Printed in JAPAN

Typeface Conventions

Bold	Boldface type is used when a term is defined. For example: **icons** are symbols.
Italics	Italic type is used for emphasis and for titles of manuals and other publications.
	Italic type is also used for keyboard entries when a name or a variable must be typed in place of the words in italics. For example: `copy` *filename* means to type the word `copy`, to type a space, and then to type the name of a file such as `file1`.
`Computer`	Computer font is used for on-screen prompts and messages.
(HARDKEYS)	Labeled keys on the instrument front panel and furnished keyboard are enclosed in ( ).
▨ SOFTKEYS ▨	Softkeys located to the right of the LCD are enclosed in ▨.

How to Use This Manual

This guide will help you learn how to effectively use Instrument BASIC (IBASIC) of the 4396B Network/Spectrum/Impedance Analyzer. It will help you to perform typical operations involving program creation, editing, and execution. It will also show you how to save and recall programs, and how to make the best use of the Instrument BASIC's front-panel and keyboard interface. Here is a brief guide to help you locate the necessary information in this manual.

- Chapter 2 introduces the analyzer's Instrument BASIC system and describes how to connect and use a keyboard.
- Chapter 3 and Chapter 4 show creating, getting, and saving programs to teach you front panel and keyboard operation.
- Chapter 5 introduces you to the editing environment.
- Chapter 6 provides application programs and useful techniques for developing programs.
- Chapter 7 describes interfacing features for graphics, external connector to trigger RUN/CONTinue of a program, and I/O port.
- Chapter 8 introduces special features for auto loading a program, and the On Key Label function (softkeys defined in a program). This chapter also describes techniques for speeding up your programs.
- Chapter 9 summarizes the unique features specified for the analyzer.
- Appendix A contains the information required to adept this manual to earlier versions or configurations of the analyzer than the current printing date of this manual.
- Appendix B provides references for BASIC commands specific to the analyzer's Instrument BASIC.
- Appendix C provides a handy reference guide to the analyzer's Instrument BASIC's key definitions for the mini-DIN keyboard.
- Appendix D describes the softkeys that are used for the Instrument BASIC operations.

Contents

1. **Welcome to Instrument BASIC**
How to Use This Manual 1-1

2. **Introduction to the System**
Overview of Instrument BASIC 2-1
Connecting the Keyboard 2-2
Using Instrument BASIC for the First Time 2-3
Allocating Screen Area for Instrument BASIC 2-3
Setting the Size of Memory Area for Instrument BASIC 2-4
Using the Keyboard 2-5
What can the Keyboard be Used for? 2-5
Performing Calculations 2-5
Entering Arguments to the Active Analyzer Functions 2-5
Entering Titles 2-6
Executing Commands 2-6
Using Softkeys 2-7
Entering BASIC Statements from the Front Panel Keys 2-7

3. **Writing and Running Programs**
Getting into/out of the EDIT Mode 3-1
Getting into the EDIT Mode 3-1
Getting out of the EDIT Mode 3-1
Writing Programs 3-2
Controlling the Analyzer 3-2
Running (Executing) Programs 3-5
Listing Programs 3-5
Listing on the Screen 3-5
Listing to the Printer 3-6
If You Need More Information 3-6

4. **Saving and Getting Programs**
Saving Programs (SAVE) 4-1
Listing File Names (CAT) 4-2
Listing to Screen 4-2
Listing to Printer 4-3
Getting Programs (GET) 4-3
If You Need More Information 4-3

5. Editing Programs
Getting Into/Out of the EDIT Mode 5-1
Getting Into the EDIT Mode using the Front Panel Keys 5-1
Entering the EDIT Mode from the Keyboard 5-1
Getting Out of the EDIT Mode 5-1
Editing Programs in the EDIT Mode 5-2
Deleting Characters 5-2
Back Space 5-2
Deleting Characters 5-2
Inserting Characters 5-2
Moving the Cursor 5-2
Scrolling Lines and Pages 5-2
Scrolling Lines 5-2
Scrolling Pages 5-2
Jumping from the Current Line 5-3
Jumping to a Specified Line 5-3
Jumping to the Top/Bottom of a Program 5-3
Inserting/Deleting/Recalling Lines 5-3
Clearing Line 5-3
Renumbering Program Line Numbers 5-3

6. Application Programs
Controlling the Analyzer Using Instrument BASIC 6-1
I/O Operation from Instrument BASIC 6-2
Data Transfer Using the I/O Port 6-2
Reading Data from the I/O Port 6-2
Writing Data to the I/O Port 6-2
Disk I/O for a Storage Unit 6-3
Saving Trace Data 6-3
Loading Trace Data 6-3
Using Instrument BASIC with an External Controller 6-4
Sharing One Printer Between Two Controllers 6-5
Loading Instrument BASIC Programs Using Softkeys 6-6

7. Program I/O
Graphics 7-1
Instrument BASIC Graphics Commands 7-2
Hard Copies 7-2
PRINT 7-2
Initial settings 7-2
Example of Graphics Programming 7-2
Using the External RUN/CONT Connector 7-3
File System Exceptions 7-3
Using the I/O Port in BASIC Programs 7-4

8. Special Features and Advanced Techniques
Autoloading and Running a Program Automatically (AUTOST) 8-1
On Key Label Function 8-1
Increasing Program Speed 8-2

9. Analyzer Specific Instrument BASIC Features
Available I/O Interfaces and Select Codes 9-1
Storage Unit 9-1
  Built-in Flexible Disk Drive 9-2
  RAM Disk Memory 9-2
GPIB Commands for Instrument BASIC 9-3

A. Manual Changes
Introduction A-1
Manual Changes A-1
Instruments Covered by This Manual A-2

B. BASIC Commands Specific to 4396B
BASIC Commands Not Implemented B-1
BASIC Commands Specific to 4396B B-1
DATE B-2
DATE$ B-3
EXECUTE B-4
READIO B-7
SET TIME B-8
SET TIMEDATE B-9
TIME B-10
TIME$ B-11
WRITEIO B-12

C. The Keyboard
Character Entry Keys C-2
Cursor-Control and Display-Control Keys C-2
Numeric Keypad C-3
Editing Keys C-3
Program Control Keys C-3
System Control Keys C-4
Softkeys and Softkey Control C-4
  Softkey Control Keys C-4
  Softkeys C-4
  Softkeys Accessed from (Shift) - (F9) Key C-5
    IBASIC Menu C-5
  Softkeys Accessed form (F10) Key C-6
Using (CTRL) Key in Edit Mode C-6

D. Softkeys Used for Instrument BASIC Operation
(System) D-2
  IBASIC D-2
  MEMORY PARTITION D-5
(Display) D-6
  DISPLAY ALLOCATION D-6
    Run Light Indications D-7

Index

Figures

2-1. Configuration Example of the Instrument BASIC System 2-2
6-1. Sending GPIB Command 6-1
6-2. Reading I/O Port 6-2
6-3. Writing Data to the I/O Port 6-2
6-4. Saving Trace data 6-3
6-5. Loading Trace Data 6-3
6-6. Transferring the Program to an External Controller (on the External Controller) 6-4
6-7. Sharing a Printer (Program for External Controller) 6-5
6-8. Sharing a Printer (Program for Instrument BASIC) 6-5
6-9. Loading Instrument BASIC Programs Using Softkeys 6-7
7-1. Screen Structure 7-1
7-2. RUN/CONT Trigger Signal 7-3
A-1. Serial Number Plate A-2
C-1. mini-DIN Keyboard C-1
D-1. IBASIC Menu D-2
D-2. Memory Partition Menu D-5
D-3. Display Allocation Menu D-6
D-4. Display Allocation D-7

Tables

A-1. Manual Changes by Serial Number A-1
A-2. Manual Changes by Firmware Version A-1
B-1. GPIB Commands for EXECUTE B-4

1

Welcome to Instrument BASIC

This guide will help you learn how to effectively use Instrument BASIC (IBASIC) of the 4396B Network/Spectrum/Impedance Analyzer. It will help you to perform typical operations involving program creation, editing, and execution. It will also show you how to save and recall programs, and how to make the best use of the Instrument BASIC's front-panel and keyboard interface.

If you are new to programming or to HP's dialect of BASIC, take the time to read this guide and perform the exercises. For many users, this will provide all the information that is needed to create and run programs.

How to Use This Manual

The tasks in each chapter, when performed in sequential order, demonstrate a typical use of Instrument BASIC and cover the most common tasks. Read the overview and try the sample tasks in each chapter to get you started. For more background information, you can read further into each chapter; otherwise, go to the next exercises and continue the session. You can refer back to the individual chapters for more information as necessary. Here is a brief guide to help you locate the necessary information in this manual and the other Instrument BASIC manuals.

- Chapter 2 introduces the analyzer's Instrument BASIC system and describes how to connect and use a keyboard.
- Chapter 3 and Chapter 4 show creating, getting, and saving programs to teach you front panel and keyboard operation.
- Chapter 5 introduces you to the editing environment.
- Chapter 6 provides application programs and useful techniques for developing programs.
- Chapter 7 describes interfacing features for graphics, external connector to trigger RUN/CONTinue of a program, and I/O port.
- Chapter 8 introduces special features for auto loading a program, and the On Key Label function (softkeys defined in a program). This chapter also describes techniques for speeding up your programs.
- Chapter 9 summarizes the unique features specified for the analyzer.
- Appendix A contains the information required to adept this manual to earlier versions or configurations of the analyzer than the current printing date of this manual.
- Appendix B provides references for BASIC commands specific to the analyzer's Instrument BASIC.
- Appendix C provides a handy reference guide to the analyzer's Instrument BASIC's key definitions for the mini-DIN keyboard.
- Appendix D describes the softkeys that are used for the Instrument BASIC operations.

Note

You should become familiar with the operation of the analyzer before attempting to control it using Instrument BASIC. See the following documents that are better suited to this task.

User's Guide
Task Reference
Function Reference
GPIB Programming Guide
GPIB Command Reference
Instrument BASIC Users Handbook

Note

This manual, *Using Instrument BASIC with the 4396B* , is not intended to teach the Instrument BASIC programming language; see the following document which is better suited to these tasks.

Instrument BASIC Users Handbook

The handbook consists of the following three parts:

Instrument BASIC Programming Techniques
Instrument BASIC Interfacing Techniques
Instrument BASIC Language Reference

IF you want to port HP 9000 Series 200/300 BASIC programs to Instrument BASIC, see Chapter 10, "Keyword Guide to Porting," in the *Instrument BASIC Programming Techniques*.

2

Introduction to the System

This chapter introduces the analyzer's Instrument BASIC (IBASIC) and describes how to connect and use a keyboard. Read this chapter before using Instrument BASIC with the analyzer for the first time. The topics covered in this chapter are:

- Overview of Instrument BASIC
- Connecting the keyboard
- Using Instrument BASIC for the first time
- Using the keyboard
- Entering BASIC Statements from the front panel keys

Overview of Instrument BASIC

Instrument BASIC (IBASIC) can be used for a wide range of applications from simple recording and playback of measurement sequences to remote control of other instruments.

Instrument BASIC is a complete system controller residing inside your analyzer. It communicates with your analyzer via GPIB commands and can also communicate with other instruments, computers, and peripherals over the GPIB interface.

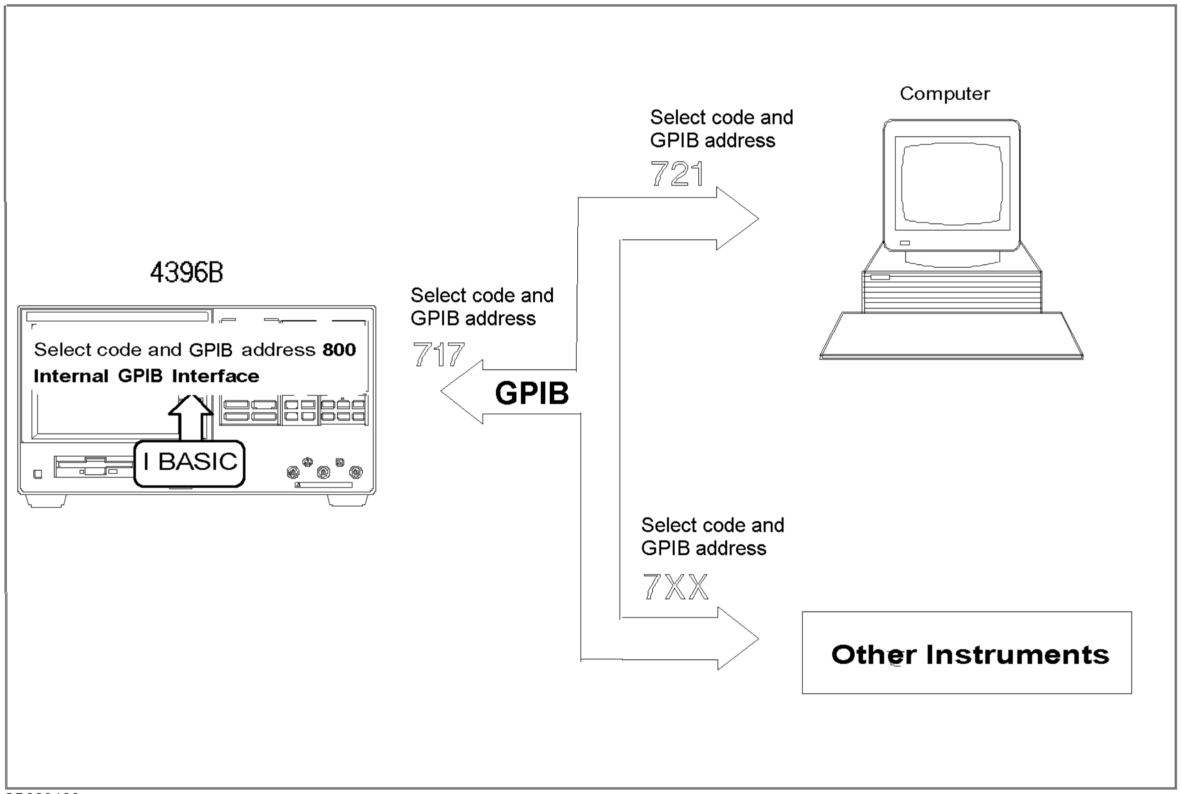

Computer
Select code and GPIB address 721
4396B
Select code and GPIB address 717
Select code and GPIB address 800
Internal GPIB Interface
GPIB
I BASIC
Select code and GPIB address 7XX
Other Instruments
CB300103

Figure 2-1. Configuration Example of the Instrument BASIC System

The Instrument BASIC's programming interface includes an editor and a set of programming utilities. The utilities allow you to perform disk I/O, renumber, secure, or delete all or part of your program.

The Instrument BASIC command set is similar to the command set of HP 9000 Series 200/300 BASIC. Therefore, Instrument BASIC programs can be run on any BASIC workstation with few if any changes. Porting information can be found in the *Instrument BASIC Programming Techniques* of the *Instrument BASIC Users Handbook*.

Connecting the Keyboard

Note Turn OFF the analyzer before inserting or removing the keyboard connector.

When you use Instrument BASIC, connect the furnished keyboard to the mini-DIN connector on the rear panel.

Using Instrument BASIC for the First Time

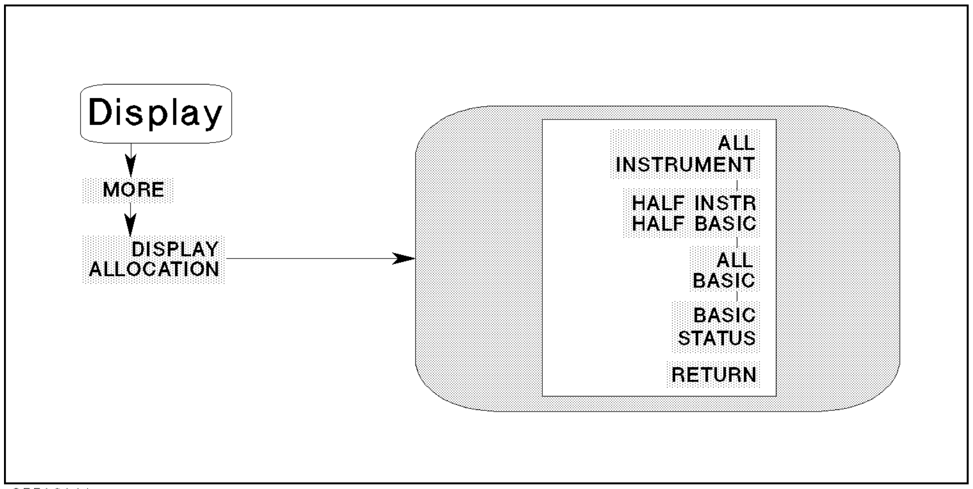
Allocating Screen Area for Instrument BASIC

Because all of the analyzer's screen is allocated for analyzer operation after power ON, you must allocate screen area for Instrument BASIC when you want to use it. The analyzer provides four display allocation types. Select one of them using DISPLAY ALLOCATION under (Display).

Let's try

1. Press the following key and softkeys:

   (Display) MORE DISPLAY ALLOCATION



C5502001

2. Press the following softkey.

   ALL BASIC

   The screen is cleared and all of the screen area is allocated for Instrument BASIC.

3. Press the following softkey.

   ALL INSTRUMENT

   The total screen area is reallocated as the analyzer display.

4. Press the following softkey:

   HALF INSTR HALF BASIC

   The screen area is allocated so that the upper half of the screen is used for the analyzer operation and the lower half is used for Instrument BASIC.

5. Press the following softkey:

   BASIC STATUS

   Three blank lines appear at the display line (lower area of the screen). This area is used by Instrument BASIC to input commands and to display messages.

More information on the display allocations for the Instrument BASIC area is described in "Display" in Appendix D.

Setting the Size of Memory Area for Instrument BASIC

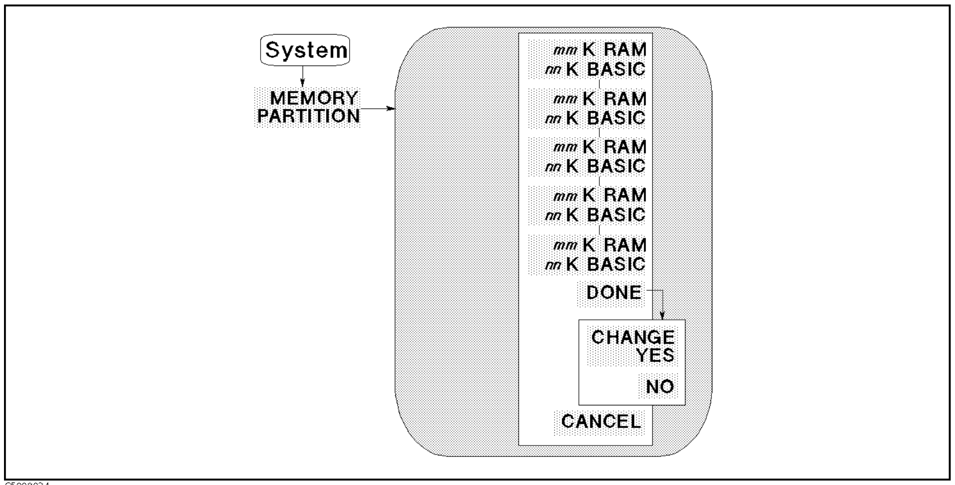
The size of the memory areas for the RAM disk memory and the variable of Instrument BASIC (excluding common variables) can be changed according to your application.

Caution When the memory partition is reconfigured, the analyzer goes to the initial settings. That is, the RAM disk memory is initialized and all the data saved in the RAM disk memory is destroyed, and the program on the BASIC editor is destroyed.

Let's try

1. Press the following key and softkey.

   System MEMORY PARTITION



2. Press the desired softkey and DONE.
3. CHANGE YES and NO softkey labels are displayed.

   Press CHANGE YES to change the memory partition.

   Press NO to cancel changing the memory partition.

Using the Keyboard

What can the Keyboard be Used for?

The mini-DIN keyboard can be used as follows:

- Performing calculations
- Entering arguments to the active analyzer functions
- Entering titles
- Executing commands
- Using softkeys

The following simple operations show you how to use these functions.

Performing Calculations

You can perform calculations while in any display allocation type except for ALL INSTRUMENT.

Let's try

1. Press the following key and softkeys:

   (Display) MORE DISPLAY ALLOCATION ALL BASIC

   The screen is cleared and a cursor appears at the bottom left of screen.

2. Type the following key from the keyboard:

   3*2 (Enter)

   The characters you enter are displayed at the current cursor position. After pressing (Enter), the system responds with the following answer at the bottom of screen:

   ```
   6
   ```

For more information, see "Numeric Computation" in the *Instrument BASIC Programming Techniques* of the *Instrument BASIC Users Handbook*.

Entering Arguments to the Active Analyzer Functions

The numeric keys on the keyboard can be used to input the arguments for an active analyzer function the same as using the front panel keys.

Let's try

1. Press the following key and softkeys:

   (Display) MORE DISPLAY ALLOCATION ALL INSTRUMENT

2. Then press the following key:

   (Start)

   The current start frequency is displayed on the screen and becomes the active analyzer function.

3. Type a value to change the frequency from the keyboard. For example, type this:

   100000

   The START value is cleared and the value you typed is displayed.

4. Then press the following key on the keyboard:

   Enter

   The START value is changed to 100 kHz.

5. Next, type the following value and key:

   `2E6` Enter

   After pressing Enter the active function value is changed to 2 MHz. You can use the character "`E`" and "`e`" in an exponential expression.

Pressing Backspace on the keyboard deletes the last entry. This performs the same function as pressing Back Space on the front panel.

Entering Titles

The character entry keys can be used to enter a title on the screen instead of using front panel operation.

Let's Try

1. Press the following key and softkey:

   Display `MORE` `TITLE`

   A cursor appears at the top left of the graticule.

2. Type in characters using the keyboard, the characters you type appear at the top of the graticule.

3. Press the following key to terminate entry:

   Enter

You can enter standard uppercase and lowercase letters for the title, using the Shift key to access the alternate case as usual. For more information on the character entry keys, see "Character Entry Keys" in Appendix C.

Executing Commands

You can type in and execute commands from the keyboard at all times except when:

- The display allocation is ALL INSTRUMENT.
- A command is being executed.
- The analyzer is in the EDIT mode.

At all other times, you can type in commands and press Enter to present them to the system for execution. The system parses the command and takes the appropriate action.

Let's Try

1. Press the following key and softkey:

   Display `MORE` `DISPLAY ALLOCATION` `HALF INSTR HALF BASIC`

2. To check the current mass storage, type the following command:

   `SYSTEM$("MSI")` Enter

3. The system returns:

```
:,4
```

Using Softkeys

Pressing f1 through f8 on the keyboard performs the same function as pressing a softkey on the front panel.

Entering BASIC Statements from the Front Panel Keys

The analyzer's Instrument BASIC allows you to enter and execute statements from the front panel keys (if the external mini-DIN keyboard is not connected).

Press the following key and softkeys from the front panel:

System IBASIC MORE [1/3] MORE [2/3] COMMAND ENTRY

The Command Entry menu is displayed on the softkey menu area, and the active entry area displays the letters, the digits 0 through 9, and some special characters including mathematical symbols. Three sets of letters can be scrolled using the step keys, ⇧ and ⇩. To enter a statement, press the step keys for the desired letter set, rotate the knob until the arrow "↑" points at the first letter, then press SELECT LETTER . Repeat this until the complete statement is entered, then press DONE to execute the statement.

3

Writing and Running Programs

This chapter describes how to write, execute (run), and list programs. The example program in this chapter also describes how to control the analyzer from an Instrument BASIC program. Topics covered in this chapter are:

- Getting into/out of the EDIT mode
- Writing programs
- Running (Executing) programs
- Listing programs

Getting into/out of the EDIT Mode

When you write a program, you must be in the EDIT mode. For more information about the EDIT mode, see Chapter 5.

Getting into the EDIT Mode

Press the following key and softkeys from the front panel:

(System) `IBASIC` `Edit`

The system enters the EDIT mode. You can also get into the EDIT mode from the keyboard. Type and press as follows:

`EDIT` and press (Enter)

Getting out of the EDIT Mode

Press the following softkey from the front panel:

`END EDIT`

The system exits the EDIT mode. If `END EDIT` does not appear on the softkey menu, press (System) `IBASIC` from the front panel, `END EDIT` will appear at the bottom of the menu.

You can also get out of the EDIT mode from the keyboard as follows:

Press (Shift) - (Alt) - (F4), (ESC), or (Home)

Writing Programs

Controlling the Analyzer

Instrument BASIC can control the analyzer (itself) through the "internal" GPIB bus. This means that an analyzer with Instrument BASIC includes both a controller and an analyzer in the same instrument. They are connected through an internal GPIB bus.

Note

The select code of the internal GPIB interface is 8, and the GPIB address of the analyzer can be any number from 0 to 30. In this manual, we use "800" for the device selector of the analyzer.

For more information on GPIB addresses and device selectors, see "Device Selectors" in the *Instrument BASIC Interfacing Techniques* of the *Instrument BASIC Users Handbook* and "Available I/O Interfaces and Select Codes" in Chapter 9.

You can write the program by using the keyboard or by pressing keys and softkeys from the front panel procedure without using the external keyboard. Using the keyboard is very useful when you write a larger and more complex program, or type comments in a program. For detailed information on how to use the keyboard, see Appendix C.

Let's Try

The following example program selects the following measurement settings:

Active Channel Block	Channel 1 (default)
Measurement Block	Network Analyzer A/R LOG MAG format (default) Display scale to AUTO
Sweep Block	Center frequency: 70 MHz Span frequency: 100 kHz

1. Turn the analyzer ON.
2. Press the following key and softkeys to display the network analyzer mode softkeys:

   (Meas) `ANALYZER TYPE` `NETWORK ANALYZER` `RETURN`

3. Press the following key and softkeys from the front panel:

   (System) `IBASIC` `Edit`

   The system enters the EDIT mode. The cursor appears at line number 10, which is the default line number of the first program line, as follows:

```
10 _
```

4. Press the following softkey:

   `ASSIGN @Hp4396`

The commands are automatically entered at the current cursor position like this:

```
10 ASSIGN @Hp4396 TO 800_
```

5. Press the following key:

   (x1)

   The system reads the entire line.

```
10 ASSIGN @Hp4396 TO 800
20 _
```

6. Press the following softkey:

   OUTPUT @Hp4396

   The following characters are displayed on the screen:

```
10 ASSIGN @Hp4396 TO 800
20 OUTPUT @Hp4396;""
```

7. Press the following key and softkey to preset the analyzer:

   (Preset)

   The GPIB command to preset the analyzer ";PRES" is automatically entered at the current cursor position like this:

```
10 ASSIGN @Hp4396 TO 800
20 OUTPUT @Hp4396;";PRES"
```

   Then press (x1).

Note ";" preceding the GPIB command is automatically added, when you write the program by pressing keys. ";" is a separator to send more than one command in the same message.

8. Press the following key to select measurement parameter as A/R:

   OUTPUT @Hp4396 (Meas) ANALYZER TYPE NETWORK ANALYZER RETURN A/R

   The program code is automatically generated:

```
10 ASSIGN @Hp4396 TO 800
20 OUTPUT @Hp4396;";PRES"
30 OUTPUT @Hp4396;";NA;MEAS AR"
```

Then enter (×1).

9. Press the following keys and softkeys to set the center frequency and frequency span:

   (System) IBASIC OUTPUT @Hp4396 (Center) 70(M/μ) (Span) 100 (k/m) (×1)

```
10 ASSIGN @Hp4396 TO 800
20 OUTPUT @Hp4396;";PRES"
30 OUTPUT @Hp4396;";NA;MEAS AR"
40 OUTPUT @Hp4396;";CENT 70E6;SPAN 100E3"
50 _
```

10. Then press the following keys and softkeys to execute the auto scale function:

    (System) IBASIC OUTPUT @Hp4396 (Scale Ref) AUTO SCALE (×1)

```
10 ASSIGN @Hp4396 TO 800
20 OUTPUT @Hp4396;";PRES"
30 OUTPUT @Hp4396;";NA;MEAS AR"
40 OUTPUT @Hp4396;";CENT 70E6;SPAN 100E3"
50 OUTPUT @Hp4396;";AUTO"
60 _
```

11. To terminate the program, the END command should be entered. Press the following softkey and key:

    (System) IBASIC END (×1)

```
10 ASSIGN @Hp4396 TO 800
20 OUTPUT @Hp4396;";PRES"
30 OUTPUT @Hp4396;";NA;MEAS AR"
40 OUTPUT @Hp4396;";CENT 70E6;SPAN 100E3"
50 OUTPUT @Hp4396;";AUTO"
60 END
70 _
```

12. Press the following softkey to exit the EDIT mode:

    END EDIT

The screen returns to the analyzer display.

Running (Executing) Programs

Press the following key and softkeys from the front panel to execute the program:

(System) IBASIC Run

The system executes the program. You can execute the RUN statement from the keyboard. Type and press as follows:

RUN (Enter)

Listing Programs

The system can list the program on the screen and to a printer.

Listing on the Screen

You can list a program on the screen as follows:

Let's Try

1. Because the system lists a program in the print area, the Print Area must be allocated on the screen. For example:

   (Display) MORE DISPLAY ALLOCATE ALL BASIC

   All of the screen area is allocated for the print area.

2. Type as follows:

   LIST (Enter)

   The system lists the program as follows:

```
10 ASSIGN @Hp4396 TO 800
20 OUTPUT @Hp4396;";PRES"
30 OUTPUT @Hp4396;";NA;MEAS AR"
40 OUTPUT @Hp4396;";CENT 70E6;SPAN 100E3"
50 OUTPUT @Hp4396;";AUTO"
60 END
```

Listing to the Printer

Note For hard copy output, an parallel cable must connect the analyzer to the printer.

Let's Try

1. Set the output device to a printer as follows:

   `PRINTER IS PRT` (Enter)

2. Type and press as follows:

   `LIST` (Enter)

   The program is listed on the printer.

3. Set the output device to LCD as follows:

   `PRINTER IS LCD` (Enter)

If You Need More Information

This chapter is an introduction to using Instrument BASIC. For more information, see the following chapters and documents:

For more information on	See
EDIT mode	Chapter 5
Keyboard and softkeys	Appendix C
Display Allocation	"(Display)" in Appendix D
Instrument BASIC commands	*Instrument BASIC Language Reference* of the *Instrument BASIC Users Handbook*
GPIB commands	*GPIB Command Reference*

Saving and Getting Programs

This chapter describes how to save and get programs to or from the built-in flexible disk drive and RAM disk memory. Topics of this chapter are:

- Saving programs (SAVE)
- Listing file names (CAT)
- Getting programs (GET)

If you are using the disk drive for the first time, see "To Save and Recall" in Chapter 6 of the *Task Reference*.

Note Instrument BASIC on the analyzer cannot communicate with an external disk drive.

Note The analyzer can use either LIF (Logical Interchange Format) or DOS (Disk Operating System) formatted disks. The instrument automatically detects the disk format. It can use most of the same operations for either disk format.

Saving Programs (SAVE)

1. To use the built-in disk drive, insert a 2DD disk or 2HD disk into the disk drive.
2. If you are using a flexible disk for the first time, set the disk format to LIF or DOS and initialize the disk. See "To Save and Recall" in Chapter 6 of the *Task Reference* for the procedure.

Note When the analyzer is turned on, the RAM disk memory is automatically initialized by the format that is set by `FORMAT [ ]` under `FILE UTILITY` under (Save). If you want to change the disk format, initialize it. See "To Save and Recall" in Chapter 6 of the *Task Reference* for the procedure.

3. If the display allocation is ALL INSTRUMENT, change the allocation. For example:

   (Display) `MORE` `DISP ALLOCATION` `ALL BASIC`

4. Specify the system mass storage device as follows:

   When you want to use the built-in disk drive, type in `MSI ":INTERNAL,4"` or `MSI ":,4"`, then press (Enter).

   When you want to use the RAM disk memory, type in `MSI ":MEMORY,0"` or `MSI ":,0"`, then press (Enter).

5. Press the following key among the 3 menus which leads to the (Shift) - (F9) key. And type in the filename to which you will store the program as follows:

`FILE UTILITY` `SAVE` *file_name* (Enter)

You can also save the file from the keyboard. Type and press as follows:

`SAVE` *file_name* (Enter)

The program is stored on the disk.

Note

If you get the error −257, "`File name error`", a file on the disk already has the name you are trying to use. In this case, you have three choices:

- Pick a new file name that doesn't already exist. To determine which file names are already being used, use the "`CAT`" command (see below).
- Replace an existing file, use the "`RE-SAVE`" statement.
- Purge the old file using the `PURGE` command, then save the new one.

Listing File Names (CAT)

Listing to Screen

Press the following key and softkeys:

1. If the display allocation is ALL INSTRUMENT or BASIC STATUS, change the allocation to either HALF INSTRument HALF BASIC or ALL BASIC. For example:

   (Display) `MORE` `DISP ALLOCATION` `ALL BASIC`

2. Press the following key among the 3 menus which leads to the (Shift) - (F9) key:

   (Menu) (on the keyboard) `FILE UTILITY` `CAT` (Enter)

   You can list from the keyboard as follows:

   Type in `CAT` then press (Enter).

   The file names stored on the disk are listed on the screen.

Note

Because the CAT statement outputs 80 columns to a line and the maximum number of columns to a screen is 61, each line is wrapped at the 62th column. If you do not want the list to wrap around, execute the following statement before executing the CAT command.

```
PRINTER IS LCD;WIDTH 80
```

CAT will list the file names with no wrap around, but anything after the 62th column in the output cannot be seen.

Listing to Printer

Note For hard copy output, an parallel cable must connect the analyzer to the printer.

1. Set the output device to be a printer as follows:

   `PRINTER IS PRT` Enter

2. Press the following key among the 3 menus which leads to the Shift - F9 key:

   `FILE UTILITY` `CAT` Enter

   You can list from the keyboard. Type and press as follows:

   `CAT` Enter

   The program is listed on the printer.

3. Get the output device back to LCD:

   `PRINTER IS LCD` Enter

Getting Programs (GET)

You can retrieve a program from the disk as follows:

1. If the display allocation is ALL INSTRUMENT, change the allocation to either HALF INSTRument HALF BASIC or ALL BASIC. For example:

   Display `MORE` `DISP ALLOCATION` `ALL BASIC`

2. Press the following key among the 3 menus which leads to the Shift - F9 key and type the filename you want to retrieve:

   `FILE UTILITY` `GET` *file-name* Enter

   You can get the file from the keyboard. Type and press as follows:

   `GET` *file_name* Enter

If You Need More Information

This chapter is an introduction to saving and retrieving programs on a disk. For more information, see the following chapters and documents:

For more information on	See
IBASIC menu	"IBASIC Menu" in Appendix C.
Initializing a disk	"Save Menu" in Chapter 8 of the *Function Reference*.

5

Editing Programs

This chapter describes how to edit programs using the EDIT mode. The topics covered in this chapter are:

- Getting into/out of the EDIT mode
- Editing programs in the EDIT mode
- Renumbering program line numbers

Getting Into/Out of the EDIT Mode

Getting Into the EDIT Mode using the Front Panel Keys

Pressing the following key and softkey allows you to enter the EDIT mode immediately, irrespective of Display Allocation.

(System) IBASIC Edit

Entering the EDIT Mode from the Keyboard

Use the following keys to enter the EDIT mode with the cursor positioned at the specified line number. The *line_number* can be omitted. Press the following key among the 3 menus which leads to the (Shift) - (F9) key.

EDIT *line_number* (Enter)
or type as follows:
EDIT *line_number* (Enter)

To use the keyboard, the Keyboard Input Line must be allocated on the screen. If it is not, press (Display) MORE DISPLAY ALLOCATION and select any allocation except ALL INSTRUMENT.

Getting Out of the EDIT Mode

The EDIT mode is exited by pressing (Shift) - (Alt) - (F4), (ESC), and (Home) from the keyboard (or by pressing the END EDIT softkey).

Editing Programs in the EDIT Mode

This section describes how to edit a program while in the EDIT mode, the topics are:

- Deleting characters
- Inserting characters
- Moving the cursor
- Scrolling lines and pages
- Jumping lines
- Inserting/deleting/recalling lines
- Clearing lines

See Appendix C for more information on functions of each key.

Deleting Characters

There are two functions you can use to delete characters: "Back space" and "Delete characters."

Back Space

Pressing Back space on the front panel (or on the keyboard) erases the character to the left of the cursor and moves the cursor left to the position of the erased character.

Deleting Characters

Pressing Delete char from the keyboard deletes the character at the cursor's position.

Inserting Characters

The EDIT mode is always in the insert mode. Characters you type at the keyboard are inserted before the current cursor position. (Pressing Insert performs no function.)

Moving the Cursor

The following key operations allow you to move the cursor horizontally along a line:

From the front panel	**From the keyboard**
Turning the knob	Pressing ◀ and ▶

Scrolling Lines and Pages

Scrolling Lines

The following key operations enable you to scroll lines up and down:

From the front panel	**From the keyboard**
Pressing ⇧ and ⇩	pressing ▲ and ▼

Scrolling Pages

Pressing Page Up and Page Down from the keyboard causes the display to scroll up and down in one page increments.

Jumping from the Current Line

Jumping to a Specified Line

You can specify a line by using a line number or a label name when jumping from the current line as follows:

`GOTO LINE` *line_number* (Enter)

or

`GOTO LINE` *label_name* (Enter)

If the label specified is not defined in the program, an error will occur.

Jumping to the Top/Bottom of a Program

Pressing the following keys allows you to jump to top or bottom of the program:

(Shift)-(▲)

(Shift)-(▼)

Inserting/Deleting/Recalling Lines

(Shift) - (Insert) inserts a new line above the current cursor position.

(Shift) - (Delete) deletes the line at which the cursor is.

`RECALL LINE` recalls the last deleted line.

Clearing Line

Pressing (Shift) - (End) clears a line from the current cursor position to the end of the line.

Renumbering Program Line Numbers

The REN command allows you to renumber the program currently in memory. You should execute the REN command after exiting the EDIT mode. Press the following key among the 3 menus which leads to the (Shift) - (F9) key.

`RENumber` (Enter)

or

REN (Enter)

You can specify the starting value, increment value, beginning line number, and the ending line number when renumbering a program as follows:

`RENumber` *starting_value*, *increment* IN *beginning_line_number*, *ending_line_number* (Enter)

or type as follows:

REN *starting_value*, *increment* IN *beginning_line_number*, *ending_line_number* (Enter)

line_label can be also use instead of *line_number*. For more information, see the *Instrument BASIC Language Reference* of the *Instrument BASIC Users Handbook*.

6

Application Programs

This chapter describes Instrument BASIC programming using examples. The examples correspond to actual measurement situations. These Instrument BASIC examples will supply useful information for developing the analyzer's Instrument BASIC application programs. The topics covered in this chapter are:

- Controlling the analyzer using Instrument BASIC
- I/O operation from Instrument BASIC
- Using Instrument BASIC with an external controller
- Sharing one printer between two controllers
- Loading Instrument BASIC programs using softkeys

Controlling the Analyzer Using Instrument BASIC

Instrument BASIC allows you to easily control the analyzer. This chapter describes the basic techniques for using Instrument BASIC to control the analyzer.

Note

Two quotes, in succession, will embed a quote within a string when a quotation mark needs to be in a string.

For example:

```
100 OUTPUT @Hp4396;";TITL ""This is a test."""
```

or

```
100 Title$="This is a test."
110 OUTPUT @Hp4396;";TITL """;Title$;""""
```

Sends string, `;TITL "This is a test."`, to the analyzer. (TITL displays a title.)

The analyzer and the Instrument BASIC in the analyzer should be regarded as two separate instruments interfaced by an internal GPIB bus. So, to distinguish between the internal and external GPIB interfaces, use select code "8" for the internal GPIB interface (the external select code is "7"). For more information on GPIB commands, see *GPIB Command Reference* and the *GPIB Programming Guide*. This program sends the GPIB command by using the GPIB interface from Instrument BASIC to the analyzer.

```
10 ASSIGN @Hp4396 TO 800  ! Assign GPIB path to the analyzer
20 OUTPUT @Hp4396;";PRES" ! Preset the analyzer.
30 END
```

Figure 6-1. Sending GPIB Command

Note For sample programs to control the analyzer, see the *GPIB Programming Guide*.

I/O Operation from Instrument BASIC

This paragraph describes the input/output operations using the I/O port and the storage unit. The following programs are covered in this section:

- Data transfer using the I/O port
  - Reading data from the I/O port
  - Writing data to the I/O port
- Disk I/O for a storage unit
  - Saving trace data
  - Loading trace data

Data Transfer Using the I/O Port

The following two examples show input and output operations of the I/O port. The `READIO` and `WRITEIO` commands of Instrument BASIC directly control the I/O port and are faster than the `INP8IO?` and `OUTP8IO` GPIB commands.

Reading Data from the I/O Port

This program shows how to directly read a specific data bit from the I/O port.

```
10 A=READIO(15,0)
20 PRINT A
30 END
```

Figure 6-2. Reading I/O Port

Writing Data to the I/O Port

This program shows an example of writing data to the I/O port. When you use the output port of the I/O port, the output data must be decimal data. Binary-expressed data is useful to set each bit ON or OFF. If you want to set bits of the output port using binary data, use the `IVAL` or `DVAL` command of Instrument BASIC. This command allows you to convert data from binary to decimal. The following example shows how to write binary data to the I/O port by using the `DVAL` command.

```
10 Bin_dat$="11111111"
20 Decimal_dat=DVAL(Bin_dat$,2) ! Convert binary data to decimal
30 WRITEIO 15,0;Decimal_dat
40 END
```

Figure 6-3. Writing Data to the I/O Port

Disk I/O for a Storage Unit

The analyzer has a built-in disk drive and RAM disk memory. You can save or get data using these disks with Instrument BASIC.

Saving Trace Data

This program saves the analyzer's current raw measurement data to an arbitrarily named file.

```
10 DIM File_name$[10]
20 ASSIGN @Hp4396 TO 800
30 INPUT "ENTER FILE NAME (up to 10 Characters)",File_name$
40 OUTPUT @Hp4396;"SAVDDAT """;File_name$;""""
50 END
```

Figure 6-4. Saving Trace data

Loading Trace Data

This program loads trace data from the built-in disk drive into the "`Dat`" array.

```
10 DIM Dat(1:201,1:2)
20 DIM File_name$[10]
30 ASSIGN @Hp4396 TO 800
40 MSI ":INTERNAL,4"
50 INPUT "ENTER FILE NAME (without EXT.)",File_name$
60 File_name$=File_name$&".DAT"
70 ASSIGN @File TO File_name$ ! Open target file
80 ENTER @File USING "16X,#"
90 ENTER @File USING "12X,#"
100 ENTER @File;Dat(*)        ! Load data from file
110 ASSIGN @File TO *         ! Close file
120 PRINT Dat(*)
130 END
```

Figure 6-5. Loading Trace Data

Line 40 selects the internal flexible disk drive. When you want to use the RAM disk memory, change `":INTERNAL,4"` to `":MEMORY,0"`.

Line 60 adds an extension `".DAT"` to file name for the DOS format file. When you want to use LIF format file, change `".DAT"` to `"_D"`.

Using Instrument BASIC with an External Controller

This program transfers a program from the Instrument BASIC memory to the external controller's disk through the GPIB interface. This program must be executed on the external controller.

Note

For other topics listed below, see Chapter 7, "Controlling Instrument BASIC from Remote," of the *GPIB Programming Guide*.

- Passing control between controllers
- Transferring a program to Instrument BASIC
- Running Instrument BASIC program from an external controller program
- Referring to an external controller's data array contents

```
10 DIM A$[10000]
20 ASSIGN @Hp4396 TO 717
30 OUTPUT @Hp4396;"*RST"
40 OUTPUT @Hp4396;":PROG:DEF?"
50 ENTER @Hp4396 USING "#,2A";Head$
60 B=VAL(Head$[2])
70 FOR I=1 TO B
80   ENTER @Hp4396 USING "%,A";Head$
90 NEXT I
100 ENTER @Hp4396 USING "-K";A$ ! Transfer the program
110 !
120 INPUT "File name?",File_name$
130 CREATE ASCII File_name$,1
140 ASSIGN @File TO File_name$
150 OUTPUT @File;A$
160 ASSIGN @File TO *
170 END
```

Figure 6-6.
Transferring the Program to an External Controller (on the External Controller)

Lines 50 to 90 read the file header: `#NMM ... M`.

The first byte is always "#".
`N` specifies the number of bytes that defines the program size.
`MM ... M` is program size in byte (`N` digits).

See `:PROGram[:SELected]:DEFine` in Chapter 2 of the *GPIB Command Reference* for more information.

Note

The program to be uploaded must be in either the paused or stopped state.

Sharing One Printer Between Two Controllers

This program shows an example of sharing one printer between two controllers. The analyzer and the external controller use the printer in sequence. The external controller uses the printer first. The following is assumed:

- Two controllers and one printer on the same GPIB bus
- Figure 6-7 is executed on the external controller
- Figure 6-8 is in the Instrument BASIC editor

```
10    Hp4396=717
20    Printer=PRT
30    !
40    OUTPUT Hp4396;":PROG:STAT RUN" ! Make Instrument BASIC run state
50    !
60    PRINTER IS Printer
70    PRINT "This line is printed out from ext. controller."
80    !
90    PASS CONTROL Hp4396
100   !
110   ON ERROR GOTO Not_active
120 Not_active:   ! Waiting until control is back
130   !
140   PRINT "This line is printed out from ext. controller again."
150   PRINTER IS LCD
160   END
```

Figure 6-7. Sharing a Printer (Program for External Controller)

```
10 ASSIGN @Hp4396 TO 800
20 Printer=PRT
30 !
40 PRINTER IS Printer
50 !
60 ON ERROR GOTO Not_active
70 Not_active:!
80 !
90 PRINT "This line is printed from IBASIC."
100 !
110 PASS CONTROL 721
120 PRINTER IS LCD
130 END
```

Figure 6-8. Sharing a Printer (Program for Instrument BASIC)

Loading Instrument BASIC Programs Using Softkeys

This program displays up to eight program file names in the analyzer's softkey label area. One of the programs can be selected and executed by just pressing a softkey. This feature lets you execute a program without using the keyboard. You only need to select the softkey of the program you want to execute.

You can name this program file, "AUTOST", so it will be executed automatically when the analyzer is turned ON.

When you want to recall this program again after the execution of an object file, you simply add the command `GET "AUTOST"` just before the `END` statement line of your object program code.

```
10 ASSIGN @Hp4396 TO 800
20 DIM Dir$(1:200)[80],File$(1:200)[10]
30 !
40 CAT TO Dir$(*)
50 !
60 File_number=0
70 REPEAT
80 File_number=File_number+1
90 File$(File_number)=Dir$(File_number+7)[1,10]
100 UNTIL File$(File_number)="" OR File_number>200
110 !
120 Max_page=INT((File_number-1)/6)+1
130 Npage=1
140 OUTPUT @Hp4396;"USKEY" ! SYSTEM key
150 OUTPUT @Hp4396;"KEY 0"  ! IBASIC key
160 OUTPUT @Hp4396;"KEY 6"  ! ON KEY LABEL key
170 Head: !
180 Page=(Npage-1)*6
190 ON KEY 1 LABEL File$(Page+1) GOSUB Jump1
200 ON KEY 2 LABEL File$(Page+2) GOSUB Jump2
210 ON KEY 3 LABEL File$(Page+3) GOSUB Jump3
220 ON KEY 4 LABEL File$(Page+4) GOSUB Jump4
230 ON KEY 5 LABEL File$(Page+5) GOSUB Jump5
240 ON KEY 6 LABEL File$(Page+6) GOSUB Jump6
250 ON KEY 7 LABEL "NEXT PAGE" GOTO Jump7
260 ON KEY 8 LABEL "PREV. PAGE" GOTO Jump8
270 !
280 LOOP
290 END LOOP
300 !
310 Jump1:GET File$(Page+1)
320 Jump2:GET File$(Page+2)
330 Jump3:GET File$(Page+3)
340 Jump4:GET File$(Page+4)
350 Jump5:GET File$(Page+5)
360 Jump6:GET File$(Page+6)
370 Jump7:IF Npage<Max_page THEN Npage=Npage+1
380 GOTO Head
390 Jump8:IF Npage>1 THEN Npage=Npage-1
400 GOTO Head
410 !
420 END
```

Figure 6-9. Loading Instrument BASIC Programs Using Softkeys

7

Program I/O

This chapter describes how to write programs that use the LCD, the I/O port, the external RUN/CONT connector in the analyzer, and the DOS file system.

Topics covered in this chapter are:

- Graphics
- Using the external RUN/CONT connector
- File system exceptions
- Using the I/O port in BASIC programs

Graphics

Instrument BASIC adds graphics capability to the analyzer. You can draw pictures on the LCD display independent of the grids and traces.

The analyzer has two screens, the instrument screen and the graphics screen. These two screens are always displayed together on the LCD and are not separately selectable. The instrument screen consists of a trace display area and a softkey label area. The Instrument BASIC editor is also displayed on the trace display area. The graphics screen covers the entire instrument screen as shown in Figure 7-1. The graphics screen is like an independent transparent overlay in front of the instrument screen. Therefore, you can draw figures in both the trace display and softkey label areas.

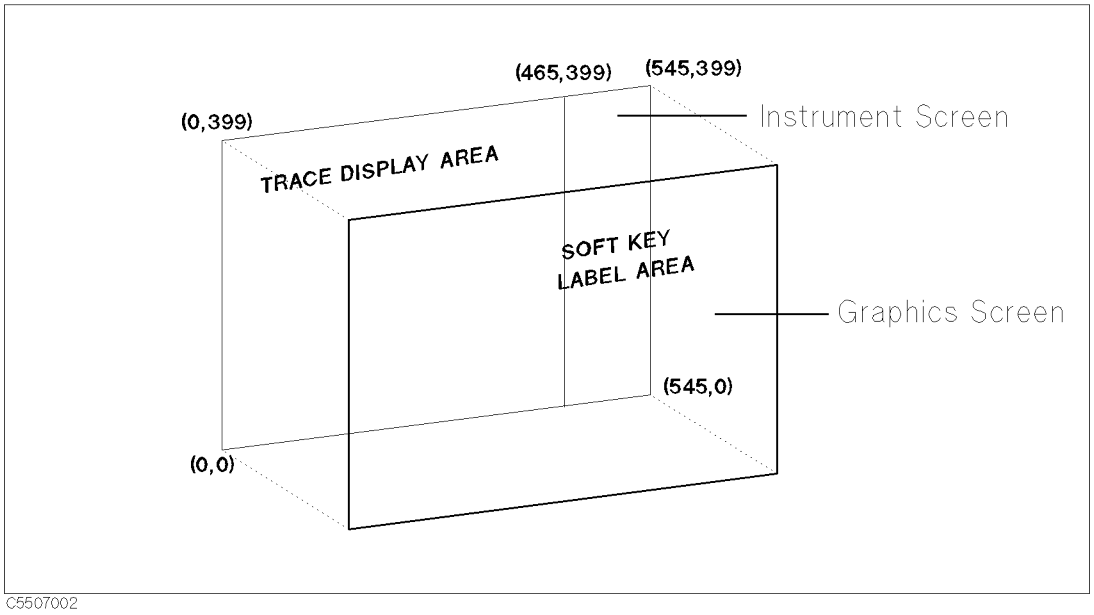


Figure 7-1. Screen Structure

Each point on the graphics screen is addressable using a coordinate address as shown in Figure 7-1. The bottom left corner is the origin (0,0) and the top right corner is the maximum horizontal and vertical end points (393,299). The MOVE and DRAW statement parameters are specified using these coordinate values. Because the aspect ratio of a graphics screen is 1, you need not adjust the aspect ratio when drawing figures.

Instrument BASIC Graphics Commands

The analyzer's Instrument BASIC has three graphics commands; MOVE, DRAW, and GCLEAR.

`MOVE`	Moves the pen from its current position to the specified coordinates.
`DRAW`	Draws a line from the current pen position to the specified coordinates.
`GCLEAR`	Clears the graphics screen, moves the pen from its current position to the origin (0,0), and selects pen 1.

Note

The total times of executing the `MOVE` and `DRAW` commands is up to 1933, even if the pen position is not changed.

Hard Copies

Graphics hard copies can be obtained with the printing function.

PRINT

`PRINT` under (Copy) prints a display image on a printer. See "Copy Menu" in Chapter 8 of the *Function Reference*.

Initial settings

When power is turned ON, the default settings are as follows:

- MOVE 0,0

Example of Graphics Programming

This section describes an example of a simple program for drawing lines on the graphics screen.

- Drawing a Straight Line

  The following Instrument BASIC program will draw a line from coordinate (50,200) to coordinate (300,200) on the display.

```
GCLEAR                  ! INITIALIZE GRAPHICS MODE
MOVE 50,200             ! MOVE PEN TO COORDINATE  (50,200)
DRAW 300,200            ! DRAW A LINE TO COORDINATE (300,200)
END
```

- Drawing a Circle

  Trying to express all graphical images using only straight lines is tedious, slow, and difficult. This example describes a subprogram you can use to draw a circle. It can draw a circle by passing the center coordinates and the radius as arguments to the following subroutine. This subroutine can be used as a base for drawing arcs, setting different values for Theta, etc.

```
SUB Drawcircle(Centx,Centy,R)    !
  DEG                            ! USE DEGREES FOR ANGLE EXPRESSIONS
```

```
    X=Centx+R                    !
    Y=Centy                      !
    MOVE X,Y                     ! MOVE PEN TO INITIAL POINT
    For Theta=1 to 360           !
      X=INT(COS(Theta)*R+Centx)  ! NEXT X COORDINATE ON CIRCLE
      Y=INT(SIN(Theta)*R+Centy)  ! NEXT Y COORDINATE ON CIRCLE
      DRAW X,Y                   ! DRAW LINE TO NEXT POINT ON CIRCLE
    NEXT Theta                   ! UNTIL STARTING POINT IS REACHED
  SUBEND                         !
```

Using the External RUN/CONT Connector

You can use the RUN or CONT commands in a program by inputting a TTL-compatible signal to the External RUN/CONT connector on the rear panel. At the negative-going edge of a pulse more than 20 μs wide (T_p) in the LOW state will trigger RUN or CONT.

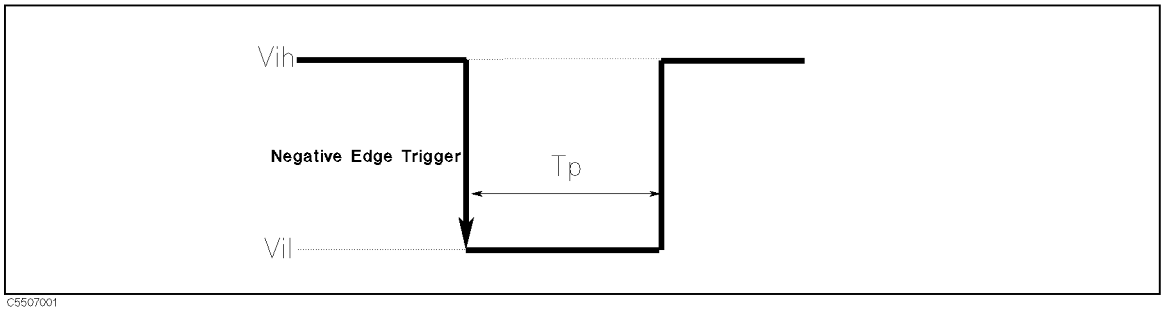


Figure 7-2. RUN/CONT Trigger Signal

File System Exceptions

The analyzer supports both the LIF and DOS file formats. When using an LIF format disk, the CREATE and CREATE DIR commands will generate an error.

Because the analyzer does not support an external disk drive, the MASS STORAGE IS (MSI) statement cannot specify volumes other than the built-in disk drive (volume specifier "INTERNAL,4", the default volume) and RAM disk memory (volume specifier "MEMORY,0").

Using the I/O Port in BASIC Programs

The Instrument BASIC can directly control the I/O port without using GPIB commands. This is faster than using the `INP8IO?` and `OUTP8IO` GPIB commands.

`READIO(15,0)`	Reads the 4-bit data from the I/O Port and returns a decimal value.
`WRITEIO 15,0;`*data*	Outputs the decimal value of the 8-bit data to the OUT 0 to 7 lines of the I/O port. The OUT 0 signal is the LSB (least significant bit), while the OUT 7 signal is the MSB (most significant bit).

See Appendix B for more information on `READIO` and `WRITEIO` commands.

For more information on the I/O port, see "I/O port" in Chapter 12 of the *Function Reference*.

8

Special Features and Advanced Techniques

The topics covered in this chapter are :

- Autoloading and running a program automatically (AUTOST)
- On Key Label function
- Increasing program speed

Autoloading and Running a Program Automatically (AUTOST)

The analyzer allows you to create a special program file called AUTOST. This program is automatically loaded and run every time the analyzer is turned ON.

When you use this capability, the disk on which you saved AUTOST must be inserted in the disk drive before the analyzer is turned ON.

The system first checks to see if there is an AUTOREC file on the disk. If there is, the system reads the AUTOREC file to set up the analyzer, and then loads and runs the AUTOST program. (For more information on AUTOREC, see "Auto Recall Function" in Appendix C of the *Function Reference.)*

On Key Label Function

The Instrument BASIC allows you to define softkeys from within a program. The softkey labels you define will appear when pressing the (Shift) - (F10) key on the Keyboard. The labels are displayed while running the program.

The `ON KEY` statement is used to define the softkeys. For example:

```
......
100 ON KEY 1 GOTO 150
110 ON KEY 2 LABEL "Print" GOSUB Report
......
```

The `KEY` statement is used to display the softkey labels defined. The following set of statements is the same as the key strokes (System) `IBASIC` `ON KEY LABELS` :

```
......
200 OUTPUT @Hp4396;"KEY 47"      ! SYSTEM key
210 OUTPUT @Hp4396;"KEY 0"       ! IBASIC softkey
220 OUTPUT @Hp4396;"KEY 7"       ! ON KEY LABELS softkey
......
```

For more information on the ON KEY statement, see the *Instrument BASIC Language Reference* of the *Instrument BASIC Users Handbook.*

Example programs for ON KEY LABEL keys are shown in Chapter 6.

Increasing Program Speed

Because the analyzer's CPU interleaves processing measurements and executing a program, program execution speed depends on the measurement conditions. The display process also requires processing time.

To increase program speed (increase throughput), set the analyzer to the following conditions:

- If you do not need to measure the DUT when executing a program, set TRIGGER MODE to HOLD.
- If you need to measure the DUT but do not need to display the traces on the screen, set DISPLAY ALLOCATION to ALL BASIC.
- If you need to measure the DUT and display traces, but do not need to use the marker function, preset all markers.
- When you use the I/O port, use the `READIO` and `WRITEIO` commands to input or output data to the port directly.
- If you change channels in a program, set Dual Channel to ON before changing channels to avoid the setup time for the channel.

For example, when you change channels in a program, set Dual Channel to ON and Display Allocation to All BASIC to decrease the switching time between channels 1 and 2.

9

Analyzer Specific Instrument BASIC Features

This chapter lists and summarizes the Instrument BASIC features specific to the analyzer. Details of each feature are described in the previous chapters and in the appendixes.

This chapter covers the following topics:

- Available I/O interfaces and select codes
- Storage units
- GPIB commands for Instrument BASIC

Available I/O Interfaces and Select Codes

Available interfaces and their select codes in the analyzer's Instrument BASIC are listed in the following table:

Select Codes	Devices
1	LCD
2	Keyboard
7	External GPIB interface
8	Internal GPIB interface

Note The analyzer does not have an RS-232C interface.

Storage Unit

The analyzer has two types of storage units: the built-in flexible disk drive and the RAM disk memory. Both the disk drive and RAM disk memory support the LIF and DOS formats.

To switch the system's storage units between the disk in the disk drive and the RAM disk under control of Instrument BASIC,

`MSI ":INTERNAL,4"` or `MSI ":,4"` for the built-in disk drive
`MSI ":MEMORY,0"` or `MSI ":,0"` for the RAM disk memory

Note

When you want to manage the storage units using the following GPIB commands, use the `STODDISK` command (for the built-in disk drive) or the `STODMEMO` command (for the RAM disk memory) to specify the storage unit.

• CHAD	• INID	• RESAVD	• SAVDGRA
• CRED	• PURG	• SAVDASC	• SAVDSTA
• DISF	• RECD	• SAVDDAT	

To copy a file between the disk and RAM disk, use an `FILC` command.

Note

The `FILC` command cannot be used to copy a file if the format (LIF or DOS) of the disk in the built-in disk drive is different from that of the RAM disk.

Use the front panel key or enter an GPIB command to initialize the storage unit. (For the procedure for initialization using the front panel, see Chapter 6 of the *Task Reference.*) When using an GPIB command to initialize the storage unit, execute the following procedure:

```
10  ASSIGN @Hp4396 TO 800
20  OUTPUT @Hp4396;"STODDISK"  ! Selects the built-in disk drive
30  OUTPUT @Hp4396;"DISF DOS"  ! Selects the DOS format
40  OUTPUT @Hp4396;"INID"      ! Initializes the disk
50  END
```

Built-in Flexible Disk Drive

The analyzer's Instrument BASIC has the following disk drive limitations:

- Disk types which can be initialized by the analyzer's Instrument BASIC `INITIALIZE` statement is 1.44 MByte (2HD). 720 Kbyte (2DD) and 270 Kbyte disks cannot be initialized.
- The only `INITIALIZE` format option is the default (256 byte/sector).
- DOS formats supported. The DOS formats supported are:

  720 Kbyte, 80 tracks, double-sided, 9 sectors/track
  1.44 Mbyte, 80 tracks, double-sided, 18 sectors/track
- HFS format is not supported.
- External disk drives are not supported.

RAM Disk Memory

A part of the RAM of the analyzer can be used as a virtual disk drive; RAM disk memory. RAM disk memory can be operated in the same way as the internal disk drive.

When the analyzer is turned OFF, the data saved in the RAM disk is lost, and the RAM disk memory is automatically initialized by the format that is set by `FORMAT [ ]` under `FILE UTILITIES` under (Save).

GPIB Commands for Instrument BASIC

The PROGram subsystem commands of the analyzer's GPIB commands are used to control Instrument BASIC. The PROGram subsystem commands do the following:

- Download the program from an external controller to the analyzer
- Upload the program from the analyzer to an external controller
- Delete the program on the BASIC editor of the analyzer
- Execute the program on the BASIC editor of the analyzer
- Set or query the variables and arrays in the program on the BASIC editor of the analyzer
- Set or query the state of the program on the BASIC editor of the analyzer

See the *GPIB Command Reference* for more information and the *Programming Guide* for their usage example.

Note The PROGram subsystem commands can be used from an external controller only.

Manual Changes

Introduction

This appendix contains the information required to adapt this manual to earlier versions or configurations of the analyzer than the current printing date of this manual. The information in this manual applies directly to the 4396B Network/Spectrum/Impedance Analyzer serial number prefix listed on the title page of this manual.

Manual Changes

To adapt this manual to your 4396B, see Table A-1 and Table A-2, and make all the manual changes listed opposite your instrument's serial number and firmware version.

Instruments manufactured after the printing of this manual may be different from those documented in this manual. Later instrument versions will be documented in a manual changes supplement that will accompany the manual shipped with that instrument. If your instrument's serial number is not listed on the title page of this manual or in Table A-1, it may be documented in a *yellow MANUAL CHANGES* supplement.

In additions to change information, the supplement may contain information for correcting errors (Errata) in the manual. To keep this manual as current and accurate as possible, Agilent Technologies recommends that you periodically request the latest *MANUAL CHANGES* supplement.

For information concerning serial number prefixes not listed on the title page or in the *MANUAL CHANGES* supplement, contact the nearest Agilent Technologies office.

Turn on the line switch or execute the `*IDN?` command by GPIB to confirm the firmware version. See the *GPIB Command Reference* manual for information on the `*IDN?` command.

Table A-1. Manual Changes by Serial Number

Serial Prefix or Number	Make Manual Changes
JP1KE	

Table A-2. Manual Changes by Firmware Version

Version	Make Manual Changes

Instruments Covered by This Manual

Agilent Technologies uses a two-part, nine-character serial number that is stamped on the serial number plate (see Figure A-1) attached to the rear panel. The first four digits and the letter are the serial prefix and the last five digits are the suffix.

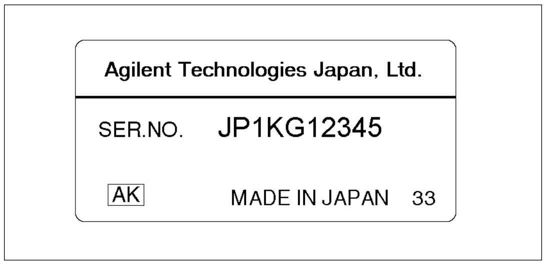


Figure A-1. Serial Number Plate

B

BASIC Commands Specific to 4396B

BASIC Commands Not Implemented

The following commands are listed in the *Instrument BASIC Language Reference* of the *Instrument Users Handbook*, but are not implemented in the analyzer's Instrument BASIC.

- `OFF CYCLE`
- `ON CYCLE`

Note

`GCLEAR` and `ON TIMEOUT` commands are available, but the following supplementary items are added.

- GCLEAR

  Moves the pen to (0,0).

- OFF TIMEOUT and ON TIMEOUT

  The interface select code is 7 only.

BASIC Commands Specific to 4396B

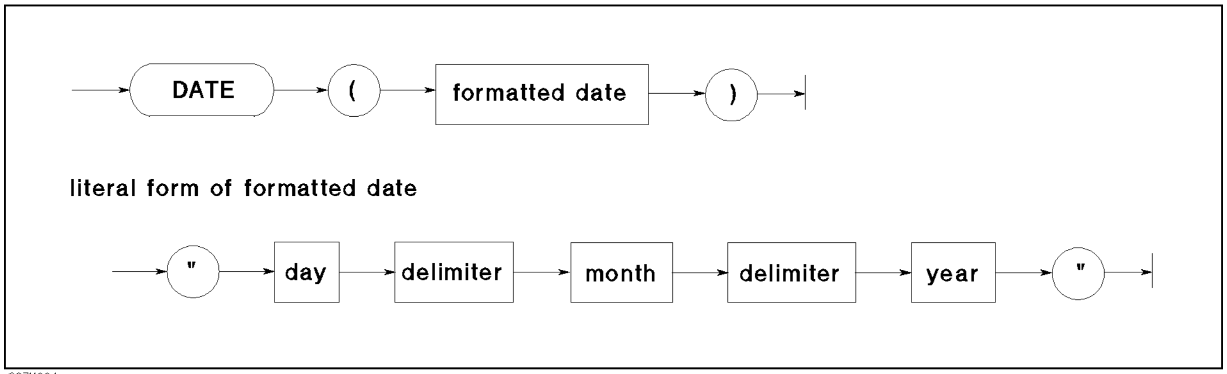
The following commands are *not* listed in the *Instrument BASIC Language Reference* of the *Instrument BASIC Users Handbook*, but are available in the analyzer's Instrument BASIC.

- `DATE`
- `DATE$`
- `EXECUTE`
- `READIO`
- `SET TIME`
- `SET TIMEDATE`
- `TIME`
- `TIME$`
- `WRITEIO`

DATE

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN ...	Yes

This command converts data formatted as (DD MMM YYY) into the numeric value used to set the clock.



C2711004

Item	Description	Range
formatted date	string expression	(see drawing and text)
day	integer constant	1 to end-of-month
month	Literal (letter case ignored)	JAN, FEB, MAR, APR, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC
year	integer constant	1900 to 2079

- Example Commands

```
PRINT DATE("21 MAY 1991")
SET TIMEDATE DATE("1 Jan 1991")
Days=(DATE("1 JAN 1991")-DATE("11 NOV 1990")) DIV 86400
```

- Semantics

Using a value from the `DATE` command as the argument for SET TIMEDATE will set the clock to midnight on the date specified. The results from the `DATE` and `TIME` commands must be combined to set the date and time of day.

If the `DATE` command is used as an argument for SET TIMEDATE to set the clock, the date must be in the range: 1 Mar 1900 to 4 Aug 2079.

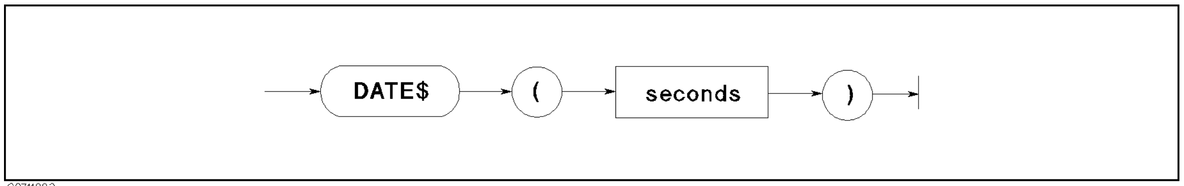
Specifying an invalid date, such as the thirty-first of February, will cause an error.

Leading blanks or non-numeric characters are ignored. ASCII spaces are recommended as delimiters between the day, month, and year. However, any non-alphanumeric character except the negative sign (−), may be used as the delimiter.

DATE$

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN ...	Yes

This command formats the number of seconds into a date (DD MMM YYY).



C2711003

Item	Description	Range
seconds	numeric expression	−4.623683256E+12 to 4.6534263350399E+13

- Example Commands

```
PRINT DATE$(TIMEDATE)
DISP DATE$(2.111510608E+11)
```

- Semantics

The date returned is in the form: DD MMM YYYY, where DD is the day of the month, MMM is the month, and YYYY is the year.

The day is a blank filled to two character positions. Single ASCII spaces delimit the day, month, and year.

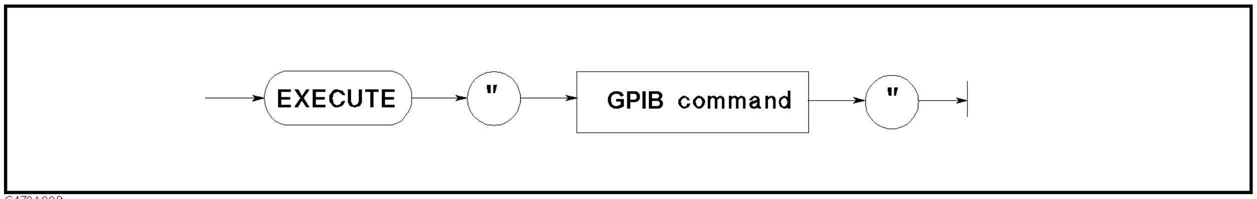
The first letter of the month is capitalized and the rest are lowercase charters.

Years less than the year 0 are expressed as negative years.

EXECUTE

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN ...	Yes

This command executes specific GPIB commands faster than the `OUTPUT` statement.



C470A009

Item	Description	Range
GPIB command	string expression	refer to Table B-1

Table B-1. GPIB Commands for EXECUTE

ANAOCH1	NUMLMAX?	PEAN?
ANAOCH2	NUMLMIN?	POLE?
ANAODATA	OUTPMAX?	RPLENV?
ANAOMEMO	OUTPMEAN?	RPLHEI?
ANARANG	OUTPMIN?	RPLLHEI?
ANARFULL	OUTPMINMAX?	RPLMEA?
EQUCPARA?	OUTPCERR?	RPLPP?
EQUCPARS?	OUTPCFIL?	RPLRHEI?
EQUCPARS4?	OUTPFILT?	RPLVAL?
EQUCO?	OUTPRESF?	SING
EQUM	OUTPRESO?	TARL?
LMAX?	OUTPRESR?	TARR?
LMIN?	OUTPXFIL?	THRR
NEXPK?	PEAK?	

- Example Commands

```
EXECUTE "SING"
EXECUTE "ANAOCH1"
```

- Semantics
  - Handling GPIB command parameters and a query command's return value when an GPIB command is executed by the `execute` command:

    To transfer GPIB command parameters, use a `WRITEIO` command. This command must be executed before the `EXECUTE` command. One `WRITEIO` command is required to transfer one parameter. For example, to transfer two `ANARANGE` command parameters to the EXECUTE command, write the program as follows:

    ```
    WRITEIO 8,0; 100E6
    WRITEIO 8,1: 200E6
    EXECUTE "ANARANG"
    ```

    To receive a query command's return value, use a `READIO` function. The `READIO` function returns only one specified return value. For example, four return values (`Za`, `Fa`, `Zr`, and `Fr`) of the query command `OUTPRESO?` must be received by writing the program as follows:

    ```
    EXECUTE "OUTPRESO?"
    Za=READIO(8,0)
    Fa=READIO(8,1)
    Zr=READIO(8,2)
    Fr=READIO(8,3)
    ```

  - GPIB Commands that work differently when executed by the `EXECUTE` command:

    When the following GPIB command is executed by the `EXECUTE` command, it works differently than when it is executed by the `OUTPUT` command.

`SING`	The 4396B executes `EXECUTE "SING"` to sweep once. Execution of the next statement is suppressed until the sweep is completed. Therefore, the completion of the sweep need not be supervised using a status register.

Note

When both an external controller and Instrument BASIC are used at the same time, the EXECUTE command occasionally does not terminate normally. If the external controller queries the instrument while the 4396B is sweeping when triggered by EXECUTE "SING", the EXECUTE command does not terminate normally (In the worst case, a system halt occurs).

To avoid this problem, it is necessary to use an SRQ interrupt technique that uses the status register. In this case, the external controller waits to send query commands until Instrument BASIC completes the execution of the EXECUTE command.

- For External Controller

```
 10  REAL Meas(1)
 20  ASSIGN @Hp4396 TO 717
 30  OUTPUT @Hp4396;"*CLS; *OPC?"
 40  ENTER @Hp4396;Opc
 50  OUTPUT @Hp4396;"OSPT 0"
 60  ON INTR 7 GOTO Anaend
 70  OUTPUT @Hp4396;"PROG:STAT RUN"
 80  !
 90 Start:!
100  OUTPUT @Hp4396;"*CLS; *OPC?"
110  ENTER @Hp4396;Opc
120  ENABLE INTR 7;2
130 Waiting: GOTO Waiting
140 Anaend:!
150  OUTPUT @Hp4396;"PROG:NUMB? MEAS"
160  ENTER @Hp4396;Meas(*)
170  PRINT Meas(*)
180  OUTPUT @Hp4396;"*CLS; *OPC?"
190  OUTPUT @Hp4396;"PROG:STAT CONT"
200  GOTO Start
210  END
```

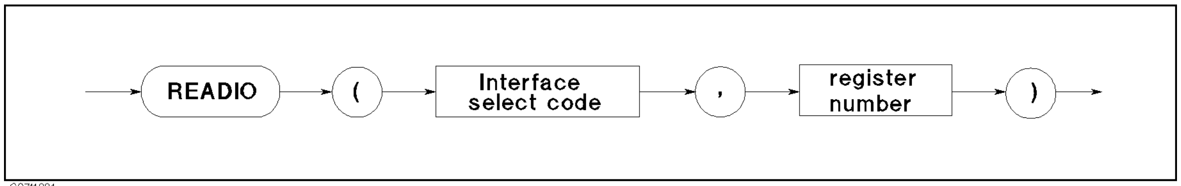
- For Instrument BASIC

```
 10 REAL Meas(1)
 20 ASSIGN @Hp4396 TO 800
 30 OUTPUT @Hp4396;"OSE 16384;OSNT 16384;*SRE 128"
 40 !
 50 LOOP
 60  EXECUTE "ANAOCH1;ANAODATA;ANARFULL"
 70  EXECUTE "SING"
 80  EXECUTE "OUTPMAX?"
 90  Meas(0) = READIO(8,0)
100  Meas(1) = READIO(8,1)
110  PAUSE
120 END LOOP
```

READIO

Keyboard Executable	Yes
Programmable	Yes
In an IF . . . THEN . . .	Yes

This command reads the contents of the register used for an I/O port or EXECUTE command.



C2711001

Item	Description	Range
select code	numeric expression	8: EXECUTE register 15: I/O port
register number	numeric expression	0 to 800 (Select code 8) 0: I/O port

- Example Commands

```
Ioport=READIO(15,0)

100  EXECUTE "OUTPRESO?"
110  Za=READIO(8,0)
120  Fa=READIO(8,1)
130  Zr=READIO(8,2)
140  Fr=READIO(8,3)
```

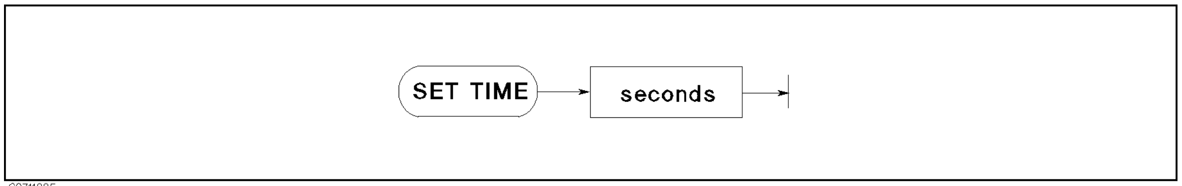
- Semantics

  The EXECUTE command stores the query command's return values in registers. The `READIO` command reads a return value from one of these registers. Return values are sequentially stored in registers 0 to 3. For example, when `EXECUTE"OUTPURESO?"` is executed, four return values `Za`, `Fa`, `Zr`, and `Fr` are stored in four registers, register 0 to register 3. `Za` is stored in register 0, `Fa` in register 1, `Zr` in register 2, and `Fr` in register 3. To read a return value stored by the `READIO` command, specify the appropriate register number. For more information on `EXECUTE` command, see the example in the "EXECUTE" command.

SET TIME

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN ...	Yes

This command resets the time-of-day given by the real-time clock.



C2711005

Item	Description	Range
seconds	numeric expression, rounded to the nearest hundredth	0 to 86399.99

- Example Commands

```
SET TIME 0
SET TIME Hours*3600+Minutes*60
```

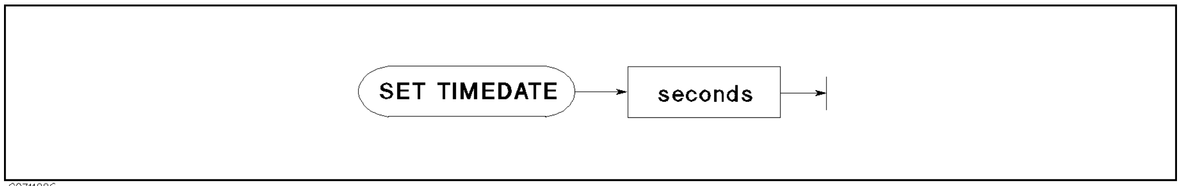
- Semantics

  This command changes only the time within the current day, not the date. The new clock setting is equivalent to (TIMEDATE DIV 86400)$\times$86400 plus the specified setting.

SET TIMEDATE

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN ...	Yes

This command resets the absolute seconds (time and day) given by the real-time clock.



C2711006

Item	Description	Range
seconds	numeric expression, rounded to the nearest hundredth	2.08662912E+12 to 2.143252224E+11

- Example Commands

```
SET TIMEDATE TIMEDATE+86400
SET TIMEDATE Strange_number
```

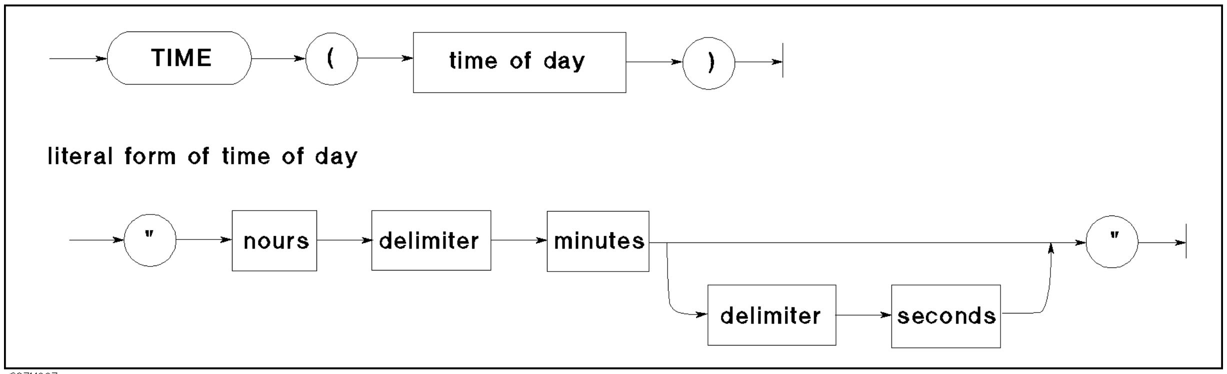
- Semantics

  The volatile clock is set to 2.08662912E+11 (midnight March 1, 1900) at power-on. If there is a battery-backed (non-volatile) clock, then the volatile clock is set to its value at power-up. If the computer is linked to an SRM system (and has no battery-backed clock), then the volatile clock is set to the SRM clock value when the SRM and DCOMM binaries are loaded. The clock values represent Julian time, expressed in seconds.

TIME

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN ...	Yes

This command converts data formatted as time of day (HH:MM:SS), into the number of seconds past midnight.



C2711007

Item	Description	Range
time of day	string expression representing the time in 24 hour format	(set drawing)
hours	literal	0 to 23
minutes	literal	0 to 59
seconds	literal; default = 0	0 to 59.99
delimiter	literal; single character	(see text)

- Example Commands

```
Seconds=TIME(T$)
SET TIME TIME("8:37:20")
ON TIME TIME("12:10") GOSUB Lunch
```

- Semantics

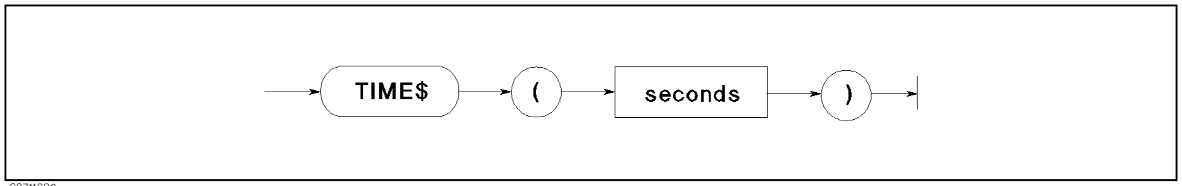
This command returns a positive integer, in the range 0 to 86399, equivalent to the number of seconds past midnight.

While any number of non-numeric characters may be used as a delimiter, a single colon is recommended. Leading blanks and non-numeric characters are ignored.

TIME$

Keyboard Executable	Yes
Programmable	Yes
In an IF . . . THEN . . .	Yes

This command converts the number of seconds past midnight into a string representing the time of day (HH:MM:SS).



C2711008

Item	Description	Range
seconds	numeric expression, truncated to the nearest second; seconds past midnight	0 to 86399

- Example Commands

```
DISP "The time is:  ";TIME$(TIMEDATE)
PRINT TIME$(45296)
```

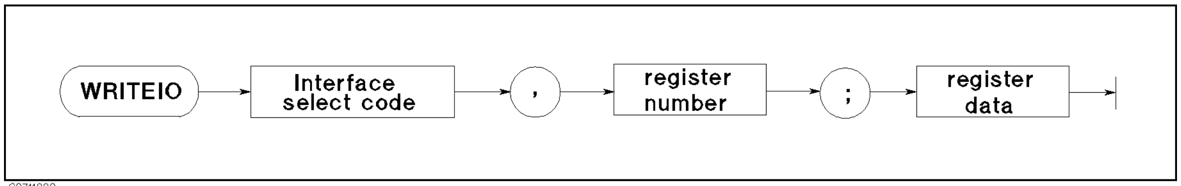
- Semantics

  TIME$ takes the time in seconds and returns the time of day in the form HH:MM:SS, where HH represents hours, MM represents minutes, and SS represents seconds. A module 86400 is performed on the parameter before it is formatted as a time of day.

WRITEIO

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN ...	Yes

This statement writes register data in decimal notation to a specified EXECUTE command parameter register or to a specified I/O port.



C2711002

Item	Description	Range
select code	numeric expression	8: EXECUTE register 15: I/O port
register number	numeric expression	0 to 800 (Select code 8) 0: I/O port
register data	numeric expression	−2147483648 to +2147483647 0 to 255: I/O port

- Example Commands

```
WRITEIO 15,0;12

100  WRITEIO 8,0; 100E6
110  WRITEIO 8,1; 200E6
120  EXECUTE "ANARANG"
```

- Semantics
  - How to write data to the I/O port:

    When writing data to an I/O port, use `15,0` as the select code and the register number that corresponds with the register. The range of register data for the I/O port is 0 through 255.

  - How to write GPIB command parameters when the `EXECUTE` command is used:

    The `EXECUTE` command uses the data stored in a register (select code 8) as a parameter. To store this parameter, the `WRITEIO` command must be executed before the `EXECUTE` command. The `WRITEIO` command stores one parameter in one register like the `READIO` command. For an GPIB command that requires multiple parameters, use as many `WRITEIO` commands as the number of parameters. For more information on the `EXECUTE` command, see the “EXECUTE” command.

The Keyboard

Mini-DIN keyboard is following key layout.

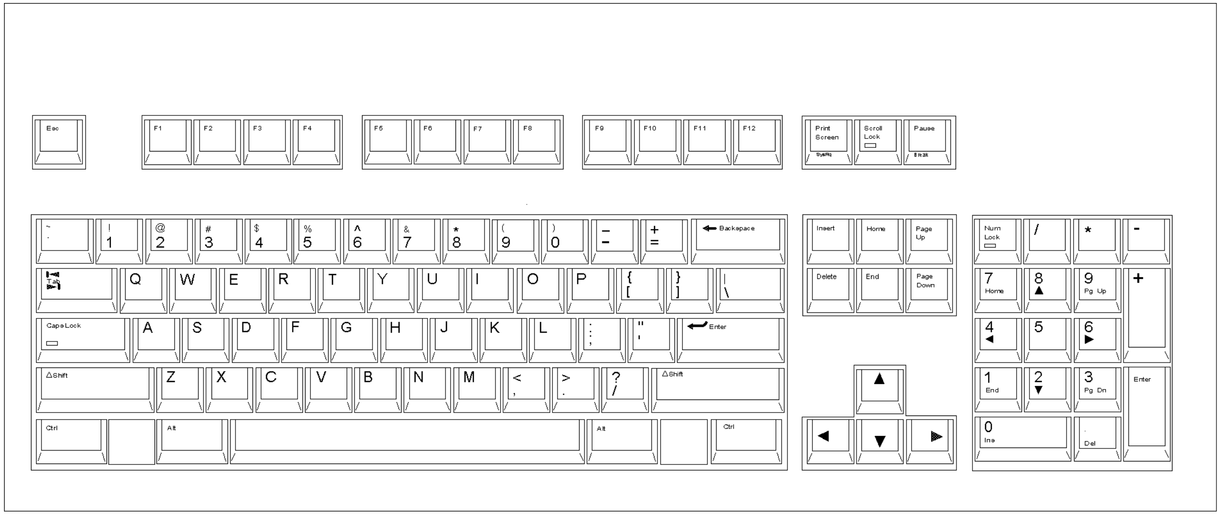


Figure C-1. mini-DIN Keyboard

Character Entry Keys

The character entry keys are arranged in the familiar QWERTY typewriter layout, but with additional features.

Caps	Sets the unshifted keyboard to either upper-case (which is the default after power ON) or lower-case (normal typewriter operation).
Shift	You can enter standard upper-case and lower-case letters, using the Shift key to access the alternate case.
Enter	Has three functions: ■ When a running program prompts you for data, respond by typing in the requested data and then press Enter. This signals the program that you have provided the data and that it can resume execution. ■ When typing in program source code, the Enter key is used to store each line of program code. ■ After typing in a command, the Enter key causes the command to be executed.
CTRL	In the EDIT mode, CTRL allows you to control the editor in the same as using the cursor-control, display-control, and editing keys. For more detail, see "Using CTRL Key in Edit Mode".
Backspace	Erases the character to the left of the cursor and moves the cursor to the erased character's position on the line.
Tab	Performs no function.

Cursor-Control and Display-Control Keys

▲ ▼	Allow you to scroll lines up and down in the print display area. Shifted, these keys cause the display to scroll toward the top or bottom of the display.
▶ ◀	Allow you to move horizontally along a line. Shifted, these keys allow you to "jump" to the left and right limits of the current line.
Page Up Page Down	Cause the display to scroll up or down in one page increments.
Home	Performs no function.

Numeric Keypad

The numerical keypad provides a convenient way to enter numbers and perform arithmetic operations. Just type in the arithmetic expression you want to evaluate, then press (Enter). The result is displayed in the lower-left corner of the screen.

(Enter)	Performs the same function as the (Enter) key. The numerical keypad serves the same function as the numerical keypad on the front panel of the analyzer.
(Num Lock)	Performs no function. Pressing the (Num Lock) key causes the LED ON/OFF, but the keys are performes as the numerical keypad only.

Editing Keys

(Insert)	Performs no function. The Instrument BASIC is always in the insert mode. The characters you type are always inserted to the left of the cursor.
(Shift) - (Insert)	Inserts a new line above the cursor's current position (edit mode only).
(Delete)	Deletes the character at the cursor's position.
(Shift) - (Delete)	Deletes the line containing the cursor (edit mode only).
(End)	Delete the line containing the cursor except the line number.
(Shift) - (End)	Clears from the current cursor position to the end of the line.
(Home)	Clears the entire alpha screen. In EDIT mode, this exits the EDIT mode.

Program Control Keys

The following keys allow you to control execution of the program stored in the analyzer's memory.

(Pause)

(Pause) or (Alt) - (F4) pauses program execution after the current line. Pressing `Continue` in the System menu resumes program execution from the point where it paused.

(Shift) - (Alt) - (F4) stops program execution after the current line. To restart the program, press `Run` in the System menu.

When in the editor mode, (Shift) - (Alt) - (F4) exits the edit mode.

(Ctrl) - (Break)

(Ctrl) - (Break) resets program execution immediately without erasing the program from memory (**BASIC RESET**).

Pauses program execution when the computer is performing or trying to perform an I/O operation. Press (Alt) - (F5) instead of (Pause) or (Alt) - (F4) when the computer is hung up during an I/O operation, because (Pause) or (Alt) - (F4) works only after the computer finishes the current program line.

System Control Keys

Shift - Page Up (Recall)	Shift - Page Up (Recall) recalls the last line the you entered, executed, or deleted. Several previous lines can be recalled this way. Recall is particularly handy to use when you mistype a line. Instead of retyping the entire line, you can recall it, edit it using the editing keys, and enter or execute it again. Shift-Page Down moves forward through the recall stack.
Alt - F3 (Run)	Starts a program running from the beginning.
Alt - F2 (Continue)	Resumes program execution from the point where it paused.
F12 (IBASIC)	Allows you to type BASIC commands on Keyboard Input Line. If Display Allocation is ALL INSTRUMENT, pressing this key changes the Display Allocation to BASIC STATUS. Shift-F12 changes Display Allocation to ALL INSTRUMENT.

Softkeys and Softkey Control

There are eight softkeys (labeled f1 through f8). The softkey labels are indicated on the right of the analyzer's screen.

Softkey Control Keys

Pressing the following:

F9	Leads to the IBASIC menu, which controls programs and the editor.
Shift - F9	leads to the BASIC menu from which to control a BASIC program. This menu is the same menu displayed when pressing SYSTEM IBASIC from the front panel. In the edit mode, pressing F10 leads to the Edit System menu, which provides softkeys to conveniently enter BASIC commands. Shift - F10 (User) leads to the ON KEY LABEL menu, which are user defined softkeys in a BASIC program. (For information on getting to this menu through the Instrument BASIC, see "On Key Label Function" in Chapter 8.)

Softkeys

F9 and F10 keys leads to the IBASIC menu. Pressing a softkey performs the command labeled or produces a sequence of characters on the keyboard input line (or on the "current line" in the EDIT mode).

Pressing the softkeys on the front panel of the analyzer performs the same functions as pressing the f1 through f8 function keys.

Softkeys Accessed from (Shift) - (F9) Key

IBASIC Menu

Pressing the following:

Step	Produces the command "Step" on the keyboard input line. Step executes a program at every line.
Continue	Produces the command "Continue" on the keyboard input line. Resumes program execution from the point where it paused.
Run	Produces the command "Run" on the keyboard input line. Immediately executes a program.
Pause	Produces the command "Pause" on the keyboard input line. Pauses program execution after the current program line is executed.
Stop	Produces the command "Stop" on the keyboard input line. Stops program execution after the current line. To restart the program, press Run.
EDIT	Produces the command "EDIT" on the keyboard input line. After EDIT is entered, pressing (Enter) enters the edit mode.
ON KEY LABELS	Leads to a softkey menu defined during program execution, if the softkey menu has been defined.
CAT	Produces the command "CAT". CAT lists the contents of a mass storage directory.
SAVE	Produces the command "SAVE". SAVE creates an ASCII file and copies program lines as strings into that file.
RE-SAVE	Produces the command "RE-SAVE". RE-SAVE creates a specified ASCII file if it does not exist; otherwise, it rewrites a specified ASCII file by copying program lines as strings into that file.
GET	Produces the command "GET". GET reads the specified ASCII file and attempts to store the strings into memory as program lines.
PURGE	Produces the command "PURGE". PURGE deletes a file or directory from the directory of a mass storage media.
INITIALIZE	Produces the command "INITIALIZE". INITIALIZE prepares mass storage media for use by the computer. When INITIALIZE is executed, any data on the media is lost.
MSI []	Produces the command "MSI []" on the keyboard input line. MSI [] specifies the mass strage. INTERNAL specifies the internal flexible disk, MEMORY specifies the RAM disk.
SCRATCH	Produces the command "SCRATCH". The SCRATCH erases the program in memory. After SCRATCH is entered, pressing (Enter) executes the command.
RENumber	Produces the command "REN". REN renumbers all of the program lines currently in memory.
LIST	Produces the command "LIST". Lists the program on the screen.
COMMAND ENTRY	Leads to the Command entry menu, which allows you to execute the Instrument BASIC commands from the front panel keys.
CLEAR I/O	Produces the command "CLEAR I/O". Pauses I/O operation program. To restart the program, press Continue.

`RESET` Produces the command "`RESET`". Aborts the program.

Softkeys Accessed form F10 Key

F10 key allows you to access three different softkey flows dependent on conditions as follows:

- Pressing F10 accesses the Program Control menu
- In editor mode, pressing F10 accesses the Edit System menu
- Pressing Shift-F10 accesses the On Key Label menu.

The menus listed above are described in "Instrument BASIC Menu" in Chapter 8 of the *Function Reference*.

Using CTRL Key in Edit Mode

In the edit mode, pressing CTRL, holding it down and pressing another key, allows you to control the editor in the same way as pressing control keys such as ▲, ▼, Insert line, etc.

If you press . . .	It performs . . .
CTRL-a	Moves the cursor to beginning of line, (the same function as Shift-◀).
CTRL-b	Moves cursor backward one character, (the same function as ◀).
CTRL-d	Deletes a character, (the same function as Delete).
CTRL-e	Moves the cursor to end of the line, (the same function as Shift-▶).
CTRL-f	Moves cursor forward character along a line, (the same function as ▶).
CTRL-g	Allows you to move the cursor to any line number or label, after press CTRL-g, type a line number or label name and press Enter, the cursor moves to the specified line, (the same function as `GOTO LINE`).
CTRL-h	Deletes backward one character, (the same function as Back Space).
CTRL-j	Performs the same function as Enter.
CTRL-k	Deletes a line from the cursor's current position to the end of the line.
CTRL-m	Performs the same function as Enter.
CTRL-n	Moves the cursor to the next line, (the same function as ▼).
CTRL-o	Inserts a new line above the cursor's current position, (the same function as Shift - Insert).
CTRL-p	Moves the cursor to the previous line, (the same function as ▲).

D

Softkeys Used for Instrument BASIC Operation

The following softkeys are available with the Instrument BASIC:

System	IBASIC controls Instrument BASIC.
	MEMORY PARTITION changes size of memory areas for Instrument BASIC and the RAM disk.
Display	DISPLAY ALLOCATION allocates the BASIC screen area on the display.

System

IBASIC

Displays the following softkeys to control Instrument BASIC (IBASIC):

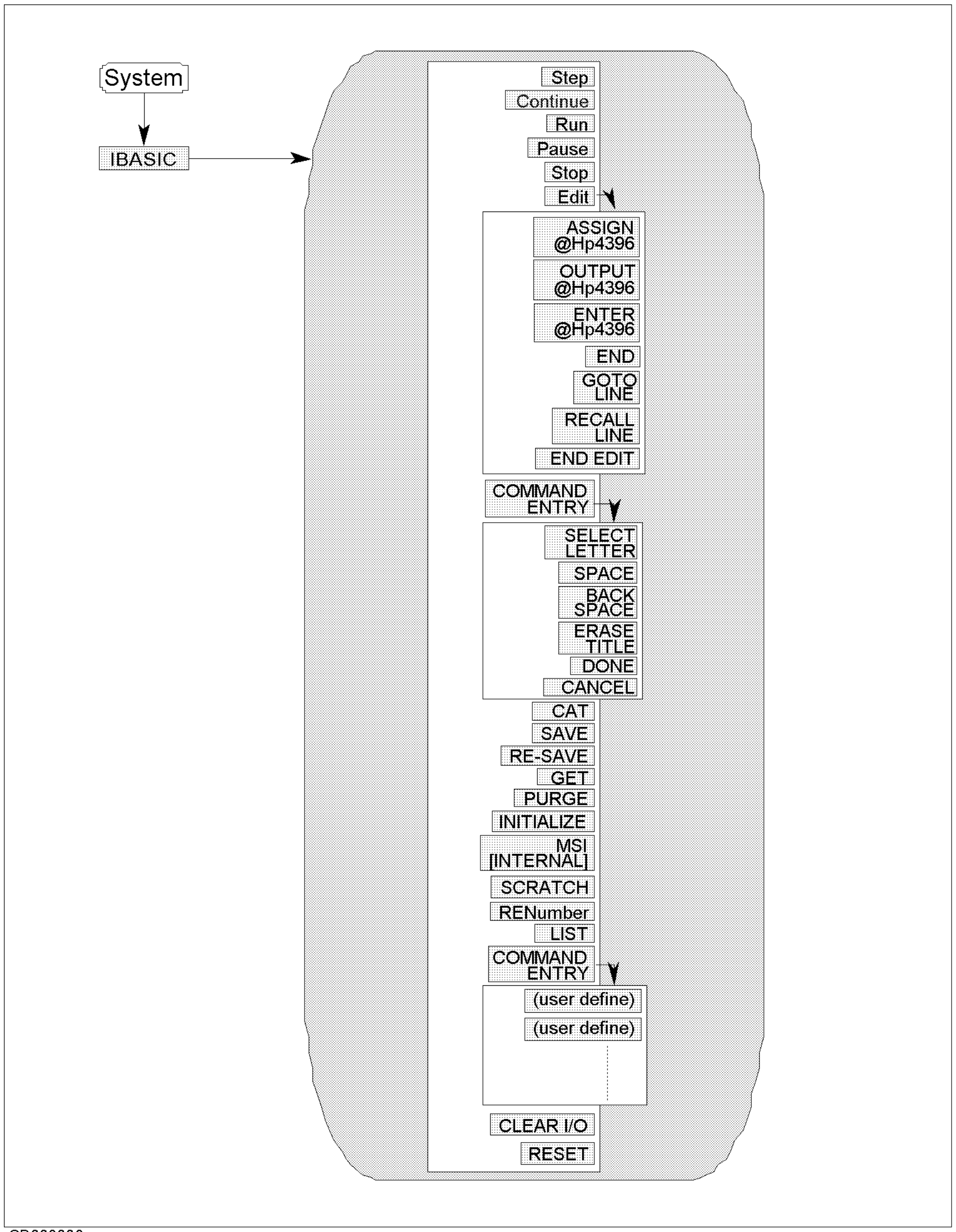


Figure D-1. IBASIC Menu

Step	Allows you to execute one program line at a time. This is particularly useful for debugging.
Continue	Resumes program execution from the point where it paused.

Run	Starts a program from its beginning.
Pause	Pauses program execution after the current program line is executed.
Stop	Stops program execution after the current line. To restart the program, press Run.
Edit	Enters into the EDIT mode.
ASSIGN @Hp4396	Produces the command "ASSIGN @Hp4396 TO 800" at the cursor's current position.
OUTPUT @Hp4396	Produces the command "OUTPUT @Hp4396;""" at the cursor's current position.
ENTER @Hp4396	Produces the command "ENTER @Hp4396;" at the cursor's current position.
END	Produces the command "END".
GOTO LINE	Allows you to move the cursor to any line number or to a label. After pressing GOTO LINE, type a line number or a label and then press Return. The cursor moves to the specified line or label.
RECALL LINE	Recalls the last deleted line.
END EDIT	Exits the edit mode.
ON KEY LABELS	Leads to a softkey menu defined during program execution, if the softkey menu has been defined.
CAT	Produces the command "CAT". CAT lists the contents of a mass storage directory.
SAVE	Produces the command "SAVE". SAVE creates an ASCII file and copies program lines as strings into that file.
RE-SAVE	Produces the command "RE-SAVE". RE-SAVE creates a specified ASCII file if it does not exist; otherwise, it rewrites a specified ASCII file by copying program lines as strings into that file.
GET	Produces the command "GET". GET reads the specified ASCII file and attempts to store the strings into memory as program lines.
PURGE	Produces the command "PURGE". PURGE deletes a file or directory from the directory of a mass storage media.
INITIALIZE	Produces the command "INITIALIZE". INITIALIZE prepares mass storage media for use by the computer. When INITIALIZE is executed, any data on the media is lost.
MSI []	Produces the command "MSI []" on the keyboard input line. MSI [] specifies the mass strage. INTERNAL specifies the internal flexible disk, MEMORY specifies the RAM disk.
SCRATCH	Produces the command "SCRATCH". The SCRATCH erases the program in memory. After SCRATCH is entered, pressing Enter executes the command.
RENumber	Produces the command "REN". REN renumbers all of the program lines currently in memory.
LIST	Produces the command "LIST". Lists the program on the screen.
COMMAND ENTRY	Leads to the Command entry menu, which allows you to execute the Instrument BASIC commands from the front panel keys.

`SELECT LETTER`	Selects the character pointed to by "↑".
`SPACE`	Inserts a space.
`BACK SPACE`	Deletes the last character entered.
`ERASE TITLE`	Deletes all characters entered.
`DONE`	Terminates command entry, and executes the command you entered.
`CANCEL`	Cancels command entry and returns to the BASIC menu.
`CLEAR I/O`	Produces the command "`CLEAR I/O`". Pauses I/O operation program. To restart the program, press `Continue`.
`RESET`	Produces the command "`RESET`". Aborts the program.

MEMORY PARTITION

Changes size of memory areas for Instrument BASIC and the RAM disk memory.

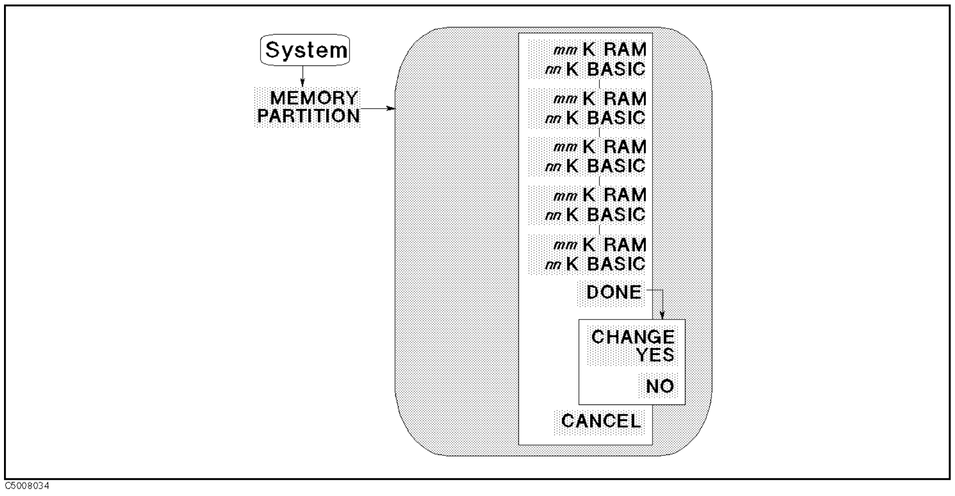


Figure D-2. Memory Partition Menu

mm K RAM nn K BASIC	Selects memory partition so that *mm* Kbyte is for RAM disk memory and *nn* Kbyte is for Instrument BASIC.
DONE	Terminates selecting memory partition and displays the following softkey labels.
CHANGE YES	Executes to change memory partition to one selected.
NO	Cancels to change memory partition and return to the previous softkey menu.

Caution When the memory partition is reconfigured, the analyzer goes to the initial settings. That is, the RAM disk memory is initialized and all the data saved in the RAM disk memory is destroyed, and the program on the BASIC editor is destroyed.

DISPLAY ALLOCATION

Displays the following menu to allocate the BASIC screen area on the display.

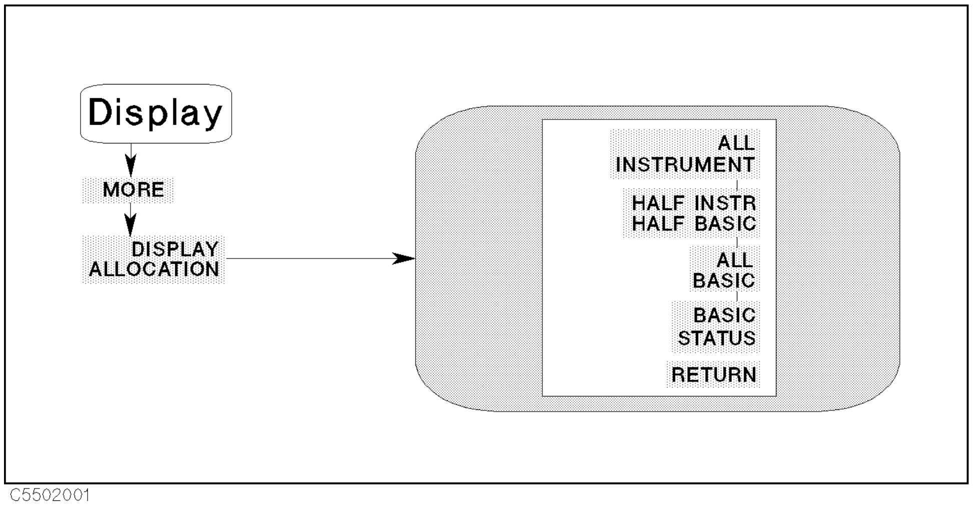


Figure D-3. Display Allocation Menu

ALL INSTRUMENT	Selects a full screen single screen or two half-screen graticules.
HALF INSTR HALF BASIC	Selects two half-screens, one graticule display above the Instrument BASIC display.
ALL BASIC	Selects a full screen single Instrument BASIC display.
BASIC STATUS	Selects a full screen graticule and three status lines for Instrument BASIC under the graticule.

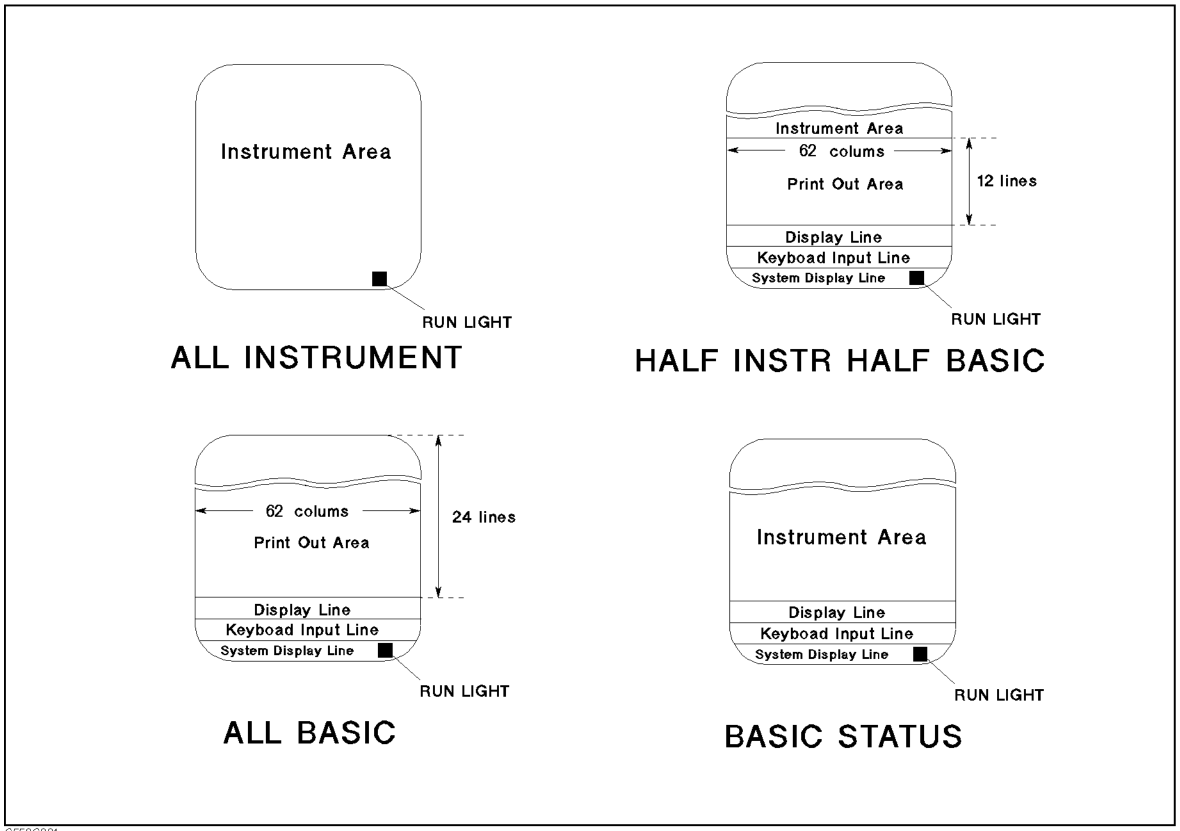


Figure D-4. Display Allocation

The following table lists the number of lines and columns in the BASIC print area for each display allocation. It also shows the keyboard input line status for each allocation. When the keyboard input line is available, you can execute BASIC commands from the keyboard.

Display Allocation	BASIC Print Area		Keyboard Input Line
	Columns	Lines	
All Instrument	0	0	not available
Half Instrument Half BASIC	62	12	available
ALL BASIC	62	24	available
BASIC Status	0	0	available

The analyzer can be connected to an external monitor. For information on the recommended monitor, see Chapter 9 of the *Function Reference*.

Run Light Indications

⊔ (blank)	Program stopped; can execute commands; CONTINUE not allowed.
_	Program paused; can execute commands; CONTINUE is allowed.
?	BASIC program waiting for input from keyboard; cannot execute commands.
*	This indication has two possible meanings: ■ Program running; CANNOT execute commands. CONTINUE not allowed. ■ System executing commanded entered from keyboard; CANNOT enter commands.

Index

A

ALL BASIC, 2-3
ALL INSTRUMENT, 2-3
arrow key
  ▼, C-2
  Home, C-2
  ◀, C-2
  ▶, C-2
  ▲, C-2
AUTOREC, 8-1
AUTOST, 8-1
auto start, 8-1

B

back space, 5-2
Backspace, C-2
BASIC STATUS, 2-3
built-in flexible disk drive, 9-2

C

Caps, C-2
CAT, 4-2
character
  deleting, 5-2
  inserting, 5-2
character entry key, C-2
command
  executing, 2-6
COMMAND ENTRY, 2-7
CREATE, 7-3
CREATE DIR, 7-3
CTRL, C-2
cursor
  moving, 5-2

D

DATE, B-2
DATE$, B-3
Delete, C-3
disk drive, 9-2
DISPLAY ALLOCATION, 2-3
DOS file system, 7-3

E

EDIT mode, 3-1, 5-1
End, C-3
Enter, C-2, C-3
"ERROR −257", 4-2
EXECUTE, B-4
external monitor, D-7
external RUN/CONT connector, 7-3
  trigger signal, 7-3

F

file name
  listing, 4-2
"File name error.", 4-2
flexible disk drive, 9-2
format option, 9-2

G

GET, 4-3
GPIB address, 3-2
graphics, 7-1

H

HALF INSTR HALF BASIC, 2-3
hard copy, 3-6
Home, C-3

I

IBASIC, 2-1
INITIALIZE, 9-2
Insert, C-3
Instrument BASIC, 2-1
"INTERNAL,4", 7-3
I/O interfaces, 9-1
I/O port, 7-4

K

keyboard, 2-5, C-1
  connecting, 2-2

L

label function, 8-1
LIF file system, 7-3
line
  clearing, 5-3
  deleting, 5-3

inserting, 5-3
jumping, 5-3
recalling, 5-3
renumber, 5-3
scrolling, 5-2
list, 3-5
on the screen, 3-5
to printer, 3-6

M

manual changes, A-1
MASS STORAGE IS, 7-3
"MEMORY,0", 7-3
mini-DIN keyboard, C-1
MSI, 7-3

N

Num Lock, C-3

O

OFF CYCLE, B-1
ON CYCLE, B-1
On Key Label function, 8-1

P

Page Down, C-2
Page Up, C-2
PEN, B-1
PRINTER IS, 3-6
program
executing, 3-5
getting, 4-3
listing, 3-5
running, 3-5
saving, 4-1
writing, 3-2
program speed
increasing, 8-2

R

RAM disk memory, 9-2
READIO, 7-4, B-7
RUN/CONT connector, 7-3
trigger signal, 7-3
run light indication, D-7

S

screen area
allocating, 2-3
select code, 3-2, 9-1
serial number, A-2
SET TIME, B-8
SET TIMEDATE, B-9
Shift, C-2
Shift - Delete, C-3
Shift - End, C-3
Shift - Insert, C-3
storage unit, 9-1

T

Tab, C-2
TIME, B-10
TIME$, B-11
title
entering, 2-6

W

WIDTH, 4-2
WRITEIO, 7-4, B-12

REGIONAL SALES AND SUPPORT OFFICES

For more information about Agilent Technologies test and measurement products, applications, services, and for a current sales office listing, visit our web site: http://www.agilent.com/find/tmdir. You can also contact one of the following centers and ask for a test and measurement sales representative. *11/29/99*

United States:
Agilent Technologies
Test and Measurement Call Center
P.O.Box 4026
Englewood, CO 80155-4026
(tel) 1 800 452 4844

Canada:
Agilent Technologies Canada Inc.
5150 Spectrum Way
Mississauga, Ontario
L4W 5G1
(tel) 1 877 894 4414

Europe:
Agilent Technologies
Test & Measurement
European Marketing Organization
P.O.Box 999
1180 AZ Amstelveen
The Netherlands
(tel) (31 20) 547 9999

Japan:
Agilent Technologies Japan Ltd.
Call Center
9-1, Takakura-Cho, Hachioji-Shi,
Tokyo 192-8510, Japan
(tel) (81) 426 56 7832
(fax) (81) 426 56 7840

Latin America:
Agilent Technologies
Latin American Region Headquarters
5200 Blue Lagoon Drive, Suite #950
Miami, Florida 33126
U.S.A.
(tel) (305) 267 4245
(fax) (305) 267 4286

Australia/New Zealand:
Agilent Technologies Australia Pty Ltd
347 Burwood Highway
Forest Hill, Victoria 3131
(tel) 1-800 629 485 (Australia)
(fax) (61 3) 9272 0749
(tel) 0 800 738 378 (New Zealand)
(fax) (64 4) 802 6881

Asia Pacific:
Agilent Technologies
24/F, Cityplaza One, 1111 King's Road,
Taikoo Shing, Hong Kong
(tel) (852)-3197-7777
(fax) (852)-2506-9284