HP 35665A Dynamic Signal Analyzer

Using HP Instrument BASIC with the HP 35665A

For Instruments with Firmware Revision
A.01.08

HP Part Number: 35665-90052
Microfiche Part Number: 35665-90252
Printed in U.S.A.

Print Date: September 1993

8600 Soper Hill Road Everett, Washington 98205-1298 U.S.A.

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MANUAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

WARRANTY

A copy of the specific warranty terms applicable to your Hewlett-Packard product and replacement parts can be obtained from your local Sales and Service Office.

Use of this manual and flexible disc(s) or tape cartridge(s) supplied for this pack is restricted to this product only. Additional copies of the programs can be made for security and back-up purposes only.

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the government is subject to restrictions as set forth in subdivision (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at 52.227-7013.

HEWLETT-PACKARD COMPANY

3000 Hanover St.

Palo Alto, CA 94303

Guide to HP 35665A Documentation

If you are thinking about...	And you want to...	Then read...
◆ Unpack and install the HP 35665A	Install the HP 35665A Dynamic Signal Analyzer	*HP 35665A Installation and Verification Guide*
	Do operation verification or performance verification tests	*HP 35665A Installation and Verification Guide*
◆ Getting started	Make your first measurements with your new analyzer	*HP 35665A Quick Start Guide*
	Review measurement basics	*HP 35665A Concepts Guide*
	Learn what each key does	*HP 35665A Operator's Reference* or use the analyzer's [**Help**] key
◆ Making measurements	Learn how to make typical measurements with the HP 35665A	*HP 35665A Operator's Guide*
	Understand each of the analyzer's instrument modes	*HP 35665A Concepts Guide*
◆ Creating automated measurements	Learn the HP Instrument BASIC interface	*Using HP Instrument BASIC with the HP 35665A*
	Record keystrokes for a particular measurement	*HP 35665A Operator's Guide*
	Program with HP Instrument BASIC	*HP Instrument BASIC User's Handbook*
◆ Remote operation	Learn about the HP-IB	*HP-IB Programming with the HP 35665A*
	Learn how to program with HP-IB	*HP-IB Programming with the HP 35665A*
	Find specific HP-IB commands	*HP 35665A HP-IB Commands: Quick Reference*
◆ Servicing the analyzer	Adjust, troubleshoot, or repair the analyzer	*HP 35665A Service Guide*

Table of Contents

Chapter 1: Introduction

Welcome to HP Instrument BASIC 1-1
HP Instrument BASIC Applications 1-2
Using HP Instrument BASIC 1-2
How to Use This Manual 1-3
Typographical Conventions 1-4
Other Sources of Information 1-4
Need Assistance? 1-5

Chapter 2: Recording Programs

Keystroke Recording 2-1
What is Keystroke Recording? 2-1
HP Instrument BASIC Programs and the HP-IB Buffer 2-2
What's in a Recorded Program 2-3
The OUTPUT Statement 2-3
The ASSIGN Statement 2-4
HP-IB Commands 2-4
How Recording Works 2-5
Operations That Are Not Recorded 2-6
Front Panel Operations Without HP-IB Commands 2-6
HP Instrument BASIC Operations 2-7
Operations Requiring Additional Programming 2-8
Operations Not Available From The Front Panel 2-8
Avoiding Recording Errors 2-9
Use Preset 2-9
Selecting Specific Parameters 2-9
Use GPIB Echo 2-10
Program Buffers and the Active Program 2-11
Selecting the Active Program 2-11
Changing a Program Label 2-12

Chapter 3: Controlling Programs

Running and Continuing a Program 3-2
Pausing a Program 3-5
Stopping a Program 3-6

Chapter 4: Saving and Recalling Programs

Transferring Programs 4-1
Disk Formats and File Systems 4-2
File Types 4-2
DOS Conventions 4-3
Using a DOS Disk to Transfer Data With a PC 4-3
LIF Conventions 4-4
Using a LIF Disk to Transfer Data with an HP BASIC computer 4-4
Program Buffers 4-5
Memory 4-5
Front Panel Operation versus Keyword Statements 4-6
The [**Save / Recall**] Menu 4-6
The Keyword Statements (SAVE, RE-SAVE and GET) 4-6
Saving a Program to Disk 4-7
Recalling a Program from Disk 4-8
Appending Program Files from Disk 4-9
Autoloading a Program 4-11

Chapter 5: Developing Programs

Overview 5-1
Using the HP Instrument BASIC editor 5-3
Using the HP Instrument BASIC Editor With a Keyboard 5-4
Using the [EDIT] Softkeys 5-8
Getting Around in the Program 5-9
Entering Program Lines 5-10
Renumbering, Copying and Moving Lines 5-10
Inserting Spaces 5-10
Inserting Lines 5-11
Recalling Deleted Lines 5-11
Using the Front-Panel Alpha Keys 5-12
Recording into an Existing Program 5-16
Removing Program Text 5-16
Using [UTILITIES] 5-18
MEMORY SIZE 5-19
AUTO MEMORY 5-19
SCRATCH 5-20
RENUMBER 5-22
SECURE 5-23
Using [PRINT PROGRAM] 5-25
Using [DISPLAY SETUP] 5-25

Chapter 6: Debugging Programs

Overview 6-2
Using [EXAMINE VARIABLE] 6-4
Examining Strings 6-4
Examining Arrays 6-4
Setting Breakpoints 6-5
Using [SINGLE STEP] 6-6
Using [RUN], [CONTINUE], and [LAST ERROR] 6-7
Using [**RESET**] 6-7

Chapter 7: Graphics and Display Techniques

Using the Partitions 7-1
Allocating Partitions 7-1
De-Allocating Partitions 7-2
Using Text 7-3
Using Graphics 7-5

Chapter 8: Interfacing with the HP-IB

Introduction 8-1
Communicating with Devices 8-2
HP-IB Device Selectors 8-2
Moving Data Through the HP-IB 8-3
General Structure of the HP-IB 8-3
Examples of Bus Sequences 8-5
General Bus Management 8-6
REMOTE 8-7
LOCAL LOCKOUT 8-8
LOCAL 8-9
TRIGGER 8-10
CLEAR 8-10
ABORT 8-11
HP-IB Service Requests 8-11
Passing and Regaining Control 8-14
The HP Instrument BASIC HP-IB Model 8-15
External and Internal Busses 8-15
Service Request Indicators 8-16
Status Registers 8-17
HP Instrument BASIC as the Active Controller 8-18
Passing Active Control to the Instrument 8-19
HP Instrument BASIC as a Non-Active Controller 8-21
Interfacing with an External Controller 8-22
Transferring Data Between Programs 8-23
Downloading and Uploading Programs 8-27

Chapter 9: Interfacing with the RS-232-C Serial Port

Introduction 9-1
RS-232-C Serial Interface 9-2
Asynchronous Data Communication 9-3
What HP Instrument BASIC Supports 9-3
Establishing the Connection 9-5
Configuring the RS-232-C Port 9-8
Character Length 9-8
Number of Stop Bits 9-8
Speed (Baud Rate) 9-9
Parity 9-9
Handshaking 9-10
Transferring Data 9-12
Entering and Outputting Data 9-12
Outbound Data Messages 9-12
Inbound Data Messages 9-13
Error Detection 9-14
The Device State Register 9-15
Event-Initiated Branching 9-16
Command Reference 9-17

Chapter 10: Example Programs

ARBSOURC 10-2
GRPDELAY 10-7
MANARM 10-14
OPC_SYNC 10-16
OPCQSYNC 10-17
RPNCALC 10-18
TWO_CTLR 10-26
TWO_CTLR 10-29
WAI_SYNC 10-31

Chapter 11: Instrument-Specific HP Instrument BASIC Features

Introduction 11-1
Global Exceptions 11-2
HP-IB, GPIO and RS-232-C Interfaces 11-2
Display and Keyboard Interfaces 11-3
Disk I/O 11-6
Miscellaneous Command Differences 11-13
Specific Exceptions 11-14

Index

1

Introduction

Welcome to HP Instrument BASIC

This manual will help you learn about using your HP Instrument BASIC software on the HP 35665A. It shows you how to use the programming, editing and debugging features of HP Instrument BASIC. It also describes how to save and recall programs and how the HP 35665A implements HP Instrument BASIC features.

If you would like to get acquainted with the language, work through the exercises in chapter 12 of the *HP 35665A Operator's Guide*. The exercises demonstrate keystroke recording, editing, saving and recalling HP Instrument BASIC programs. These example tasks provide a good overview of the HP Instrument BASIC environment in the HP 35665A.

An additional aid is on-line help which provides key-specific information on HP Instrument BASIC features. This help is accessed in the same manner as it is for other features of the HP 35665A. Press the [**Help**] hardkey followed by the desired hardkey or softkey or use the index.

HP Instrument BASIC Applications

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

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

Using HP Instrument BASIC

You need not be proficient in a programming language to successfully use HP Instrument BASIC. With keystroke recording, HP Instrument BASIC automatically builds an executable program by capturing measurement sequences as they are performed. With little or no editing of this generated code, you can put your program to work immediately controlling and automating your HP 35665A.

HP Instrument BASIC's programming interface includes an editor, a debugging program, and a set of programming utilities. The utilities allow you to set memory size as well as renumber, secure or delete your program. The remaining softkeys allow you to run or continue a program, print a listing or configure the display.

You can have up to five programs in memory at one time. Each has its own softkey that runs the program and appears in the [**BASIC**] menu. You can customize the program softkey label. You can obtain a program listing by pressing a softkey in one of the HP Instrument BASIC menus.

The HP Instrument BASIC command set is similar to the command set of HP Series 200/300 BASIC. HP Instrument BASIC programs can run on any HP BASIC workstation with few, if any, changes. Refer to chapter 8, "Interfacing with the HP-IB," for information on interfacing the HP Series 200/300 BASIC and HP Instrument BASIC environments. Porting information is located in the "HP Instrument BASIC Language Reference" section of the *HP Instrument BASIC Users Handbook.*

How to Use This Manual

Read chapters 1 through 4 to learn how to record, run, save and recall programs with a minimum of editing and programming. This information is generally adequate for those who only need HP Instrument BASIC to record their measurement tasks.

Read chapter 5, "Developing Programs," and chapter 6, "Debugging Programs" to learn how to edit programs with the front panel or with a keyboard.

Read chapter 7, "Display and Graphics Techniques," to understand how HP Instrument BASIC's graphics features apply to the HP 35665A.

Read chapter 8, "Interfacing with the HP-IB," to understand how the HP Instrument BASIC controller in the HP 35665A interfaces with external devices (such as plotters) and external controllers (such as HP Series 200/300 controllers).

Read chapter 9, "Interfacing with the RS-232-C Serial Port," to understand how the HP Instrument BASIC controller in the HP 35665A interfaces with RS-232-C compatible devices.

Refer to chapter 10 for example programs written in HP Instrument BASIC to run on the HP 35665A.

Chapter 11 couples this manual with the *HP Instrument BASIC Users Handbook.* The handbook serves users of HP Instrument BASIC on all instrument platforms. It contains three sections:

- "HP Instrument BASIC Programming Techniques"
- "HP Instrument BASIC Interfacing Techniques"
- "HP Instrument BASIC Language Reference"

Chapter 11 clarifies which parts of the handbook do not apply to the HP 35665A.

Typographical Conventions

The following conventions are used in this manual when referring to various parts of the HP Instrument BASIC and HP 35665A operating environments:

[**Hardkey**]	Brackets [ ] surrounding a bold-faced name indicate the name of a hardkey on the front panel of the HP 35665A.
[SOFTKEY]	Brackets [ ] surrounding a name indicate the name of a softkey.
[SOFTKEY **ON** OFF]	Bolded selection in a softkey indicates the state *after* the softkey is pressed.
[**Hardkey**] [SOFTKEY] [SOFTKEY]	A series of hardkeys and softkeys represents the path to a given softkey or menu.
[*Key*]	Brackets [ ] surrounding an italic typeface indicate the name of a key on the keyboard which can be used to edit HP Instrument BASIC programs.
Italic	Italic typeface is used when referring to the name of a different manual. It is also used to emphasize a particular word or phrase.
<element>	Angle brackets are used to signify a syntax element in a statement.

Other Sources of Information

- *HP 35665A Operator's Guide*
- *HP 35665A Operator's Reference*
- *HP Instrument BASIC Users Handbook*:
  - HP Instrument BASIC Programming Techniques
  - HP Instrument BASIC Interfacing Techniques
  - HP Instrument BASIC Language Reference

Need Assistance?

If you need assistance, contact your nearest Hewlett-Packard Sales and Service Office listed in the HP Catalog, or contact your nearest regional office listed at the back of this guide. If you are contacting Hewlett-Packard about a problem with your HP 35665A Dynamic Signal Analyzer, please provide the following information:

- Model number: HP 35665A
- Serial number and firmware version:
  (To locate the analyzer's serial number and firmware version, press [**System Utility**] [S/N VERSION].)
- Options:
- Date the problem was first encountered:
- Circumstances in which the problem was encountered:
- Can you reproduce the problem?
- What effect does this problem have on you?

2

Recording Programs

Keystroke Recording

Of all the available methods of creating HP Instrument BASIC programs, the easiest is keystroke recording. It requires only a couple of steps to set up and run a program. It can be accomplished with very little knowledge of programming.

You can record your program into any one of the five available locations in memory. Only one of these memory locations is active at any one time. You can select any one of the five memory locations as the currently active program.

What is Keystroke Recording?

Keystroke recording is a way to automatically create HP Instrument BASIC measurement sequence programs.

To enabling recording, press:

[**BASIC**]
[INSTRUMNT BASIC]
[ENABLE RECORDING]

Press the normal key sequences of a measurement on the analyzer. To stop recording, press the [**BASIC**] hardkey. To run the program, press the appropriate [RUN PROGRAM] softkey in the [**BASIC**] menu.

HP Instrument BASIC programs communicate with the analyzer over an internal bus. HP Instrument BASIC uses the same set of commands that external controllers use for remote operation of the instrument. Keystroke recording works by finding the bus command, called an HP-IB command, that fits each operation you perform from the front panel. It builds a program line that duplicates that operation when executed.

All program lines built by keystroke recording are entered into the analyzer's program memory. If the memory location does not contain any code, a complete executable program is inserted. If program statements exist in the memory location when recording is turned on, the recorded statements are inserted into the existing code. Chapter 5, "Developing Programs," describes how to record into existing programs.

Note

An introduction to keystroke recording with HP Instrument BASIC is included in chapter 12 of the *HP 35665A Operator's Guide*.

HP Instrument BASIC Programs and the HP-IB Buffer

Recorded programs work by sending HP-IB commands to the analyzer. The analyzer queues the HP-IB commands into its input buffer. An HP Instrument BASIC program generally outputs the commands much faster than the analyzer can execute them. The program often completes before the analyzer finishes executing the commands in the input buffer. The analyzer continues to process these commands until the buffer is empty.

This can be a problem if you are not aware of the possible delay. For example, it may not be obvious that the program has completed, since the analyzer is still functioning. This could cause confusion if you try to pause and continue a program that has actually finished.

You can clear the input buffer by inserting the statement "CLEAR 8" at the beginning of your program. Refer to chapter 5 for more information on developing and editing programs.

What's in a Recorded Program

Any program created with keystroke recording is composed of three fundamental HP Instrument BASIC statements:

- ASSIGN
- OUTPUT
- END

The following simple program demonstrates these statements:

```
1 ASSIGN @Hp35665a TO 800
2 OUTPUT @Hp35665a;"FREQ:SPAN:FULL"
10 END
```

There is only one ASSIGN statement at the beginning of a program and only one END statement at the end, but in a typical program there are many OUTPUT statements. The OUTPUT statement does the actual work of controlling the HP 35665A.

The OUTPUT Statement

The HP Instrument BASIC statement

```
OUTPUT  <destination>; <data>
```

essentially tells the internal computer to send some information (*data*) to a device at a specific address (*destination*). The destination can be a device selector (a number), or a name representing a number, called a path name. The data can take several forms but in recorded HP Instrument BASIC programs it is a string containing instructions to the analyzer.

The following command represents a typical OUTPUT statement generated from a recording session:

```
OUTPUT @Hp35665a;"FREQ:SPAN:FULL"
```

The OUTPUT command is followed by a name representing the device selector (@Hp35665a), followed by a semicolon, followed by the data. The data is in quotes ("FREQ:SPAN:FULL") and contains an instruction to the analyzer.

The ASSIGN Statement

The destination in an OUTPUT statement specifies the address of the device. In recorded programs this address is represented by the I/O path name "@Hp35665a." The following line appears in all recorded programs before any OUTPUT statements:

```
ASSIGN @Hp35665a TO 800
```

The ASSIGN statement substitutes an I/O path name (a variable name preceded by the @ symbol) for a device selector number. After the above ASSIGN statement, the program line:

```
OUTPUT @Hp35665a;"FREQ:SPAN:FULL"
```

is equivalent to:

```
OUTPUT 800;"FREQ:SPAN:FULL"
```

The device selector 800 specifies the *host instrument* as the destination of any data sent by the OUTPUT command. The program communicates with the analyzer via select code 8, the internal HP-IB interface. This select code is used solely for communication between HP Instrument BASIC programs and the analyzer. The analyzer responds to any address on the internal interface from 800 to 899. (800 is typically used.)

HP-IB Commands

The data sent to the analyzer by the OUTPUT command is called an HP-IB command. Many of the HP 35665 HP-IB commands conform to SCPI—the Standard Commands for Programmable Instruments The HP-IB command is found in quotes following the device selector path name and semicolon:

```
2 OUTPUT @Hp35665a;"FREQ:SPAN:FULL"
```

The HP-IB commands used in HP Instrument BASIC are the same ones used to remotely control the analyzer from an external computer. External computers communicate with the analyzer over the *external bus* while HP Instrument BASIC programs communicate with it over the *internal bus*. In our example, "FREQ:SPAN:FULL" tells the analyzer to set the start frequency to its minimum value and the stop frequency to its maximum value.

Note

Many, but not all of the HP 35665A's HP-IB commands conform to SCPI. Refer to *HP-IB Programming with the HP 35665A* for a complete description of HP-IB commands, including compliance to SCPI.

For more information on interfacing HP Instrument BASIC with a bus, see chapter 8, "Interfacing with the HP-IB."

How Recording Works

To fully understand HP Instrument BASIC recording, it is important to understand the relationship between the analyzer's front panel operation and the program that is generated to emulate that operation.

Note

HP-IB commands entered in a program during a recording session do not necessarily have a one-to-one correlation with the actual keys that are pressed during that session.

It is important to know that HP-IB commands correspond to an operation—not to the front panel's hardkeys and softkeys. It may take several keystrokes to perform an operation. Keystroke recording generates the appropriate HP-IB command *after* you have pressed a valid sequence of keys.

In other words, the functional operation of the analyzer is recorded, not the exact series of keystrokes.

For example, recording the key sequence:

[**Freq**]
[FULL SPAN]

requires two keystrokes but produces only one command. The command, "FREQ:SPAN:FULL," is generated after the sequence is completed. Keystroke recording automatically formats this operation into the statement:

```
OUTPUT @Hp35665a;"FREQ:SPAN:FULL"
```

and inserts it into the program.

If you accidentally press the wrong key in a sequence, it may not appear in the recorded program. It also means that you cannot exactly mimic keystrokes to leave the analyzer in a specific front-panel state. The analyzer's state appears only as a natural consequence of a *completed operation.*

For example, in the above example, pressing [**Freq**] in a recording session has the effect of bringing up the [**Freq**] menu. However, it does not, by itself, generate a line of code. You could not, therefore, set the analyzer to display the [**Freq**] menu.

Operations That Are Not Recorded

Although in most situations keystroke recording works automatically, there are some operations that are not captured or are only partially captured using this method. These operations fall into one of the following areas:

- Front panel operations with no corresponding HP-IB command such as help text operations, HP-IB controller status, RPG (knob) operations and transitional key sequences.
- Operations requiring additional programming steps, such as passing control to the analyzer for plotting or special handling of measurement operations which arm a trigger.
- HP-IB operations with no equivalent front panel operations such as HP-IB query commands.

Front Panel Operations Without HP-IB Commands

There are some front panel operations which have no corresponding HP-IB commands.

The help text available through the [Help] hardkey has no corresponding HP-IB command. Help cannot be accessed from the HP-IB. Therefore, the keystroke is not recorded.

You cannot remotely change the analyzer's controller status. This has two significant consequences:

- You cannot remotely change the state of the HP-IB interface. For example, you cannot change the analyzer from Addressable Only to the System Controller.
- You cannot remotely abort an I/O operation when the analyzer has active control of the HP-IB interface. I/O operations are for printing, plotting or using an external disk drive.

Any front-panel key sequences that perform these operations do not generate an HP-IB command. They are not keystroke recorded.

You must use the numeric keys to enter a numeric value. Even if a front panel operation allows you to increment or to decrement a value by turning the knob, the entry is not recorded.

During a measurement sequence it may take several key presses to reach an operation that generates a command. The transitional sequences between actual instrument events are not recorded.

Any default settings you do not select while recording are not recorded.

Note

It is important to remember instrument settings not specifically selected or changed *are not recorded.*

Since default states are not recorded, you must actively select them to generate a program statement. An alternate method is to make sure the analyzer is in the *same exact state* when the program runs as it was when the program was recorded. This is discussed later in this chapter in "Avoiding Recording Errors."

HP Instrument BASIC Operations

Softkeys under the [**BASIC**] key cannot be recorded because pressing this key turns off keystroke recording. In addition, [**Save/Recall**] operations that refer to an external disk drive or to another HP Instrument BASIC program are not recorded. You can, however, record all the other save and recall operations which do not refer to HP Instrument BASIC programs or to an external disk drive.

Although operations in the [**BASIC**] menus cannot be recorded, many do have corresponding HP-IB commands that allow an external controller to control and communicate with internal HP Instrument BASIC programs. See the example program, TWO_CTRL, in chapter 10 and *HP-IB Programming with the HP 35665A* for more information.

Operations Requiring Additional Programming

Some operations that work well when performed from the front panel have special circumstances that need additional attention when used in an HP Instrument BASIC program. These operations are synchronization and active control.

Synchronization

You must always anticipate timing and synchronization when one event must complete before another can occur. One example of this is when you need to detect a state in the instrument before issuing the next command. For example, you may want your program to manually arm the trigger for several measurements, but only after each measurement has successfully completed. You can record the command to set the analyzer to manual arm mode, and the command to manually arm the trigger, by pressing key sequences. However, to detect when the analyzer has completed a measurement, you must edit the program and include a routine that waits for a status register to indicate the event has occurred. (For an example of this kind of program, see the MAN_ARM program in chapter 10.)

Active Control

The [START PLOT/PRNT] operation, as well as any external disk drive operation, requires the analyzer to be the active controller on the external bus. This means that the analyzer must be set as the System Controller before the program runs; or, an external controller must pass active control to the analyzer. The instrument's active control of the external interface is automatically passed to the HP Instrument BASIC program when it begins running. Active control must be passed back to the analyzer before it can execute the print/plot or external disk operations.

Although you can keystroke record operations involving an external disk drive or [START PLOT/PRNT], you cannot successfully run the generated program. You need to add program lines to first pass active control to the analyzer and then wait for the active control of the bus to be passed back to the HP Instrument BASIC program. See "Passing and Regaining Control" in chapter 8 for an example of passing control to the analyzer.

Operations Not Available From The Front Panel

Operations such as querying the analyzer's status, transferring data over the external bus or the RS-232-C port, and setting and clearing status registers are not available from the analyzer's front panel. These operations cannot be keystroke recorded. They are useful for HP-IB programming using HP Instrument BASIC. See *HP-IB Programming with the HP 35665A* for a description of these types of operations. Refer to chapter 9, "Interfacing with the RS-232-C Serial Port," for information about using the RS-232-C interface.

Avoiding Recording Errors

This section describes ways to minimize the mistakes you can make when using keystroke recording.

Use Preset

You should preset the analyzer before recording a measurement sequence and again before running the recorded program. This sets the instrument to its default state. It avoids the risk of creating a program that depends on instrument settings that were present at the time of the keystroke recording but may be different when the program runs.

To include the command that presets the analyzer, press [**Preset**] [Do Preset] immediately after enabling keystroke recording. This inserts the following line before all the other OUTPUT statements in your program:

```
OUTPUT @Hp35665a;"SYST:PRES"
```

This sets the analyzer to its default state.

Selecting Specific Parameters

You may not want to preset the analyzer before a recorded program runs because you are recording a section of a larger measurement sequence. In this case, be sure to activate every instrument setting you need in your automated sequence. For example, if you want the format to be SINGLE, press [**Disp Format**] and then [SINGLE], even though SINGLE is already the default setting. This generates a program line which specifically sets the format to SINGLE.

In some cases you may have to select another setting first and then re-select the original setting in order to generate the correct program line. For example, you want to generate a program line to set trace A as the active trace. While you are recording you discover that A is already the active trace. Press [**Active Trace**] twice — once to select B and then again to select A. You can delete any unwanted program lines generated by this procedure.

Use GPIB Echo

You can review the HP-IB commands before you actually record them. While this is not essential, it can be very useful when you are in doubt as to what a particular key sequence will record, or precisely when a key sequence corresponding to an HP-IB command is completed.

GPIB Echo is a facility that allows you to view HP-IB commands corresponding to any operation executed from the front panel. A command appears in the upper left corner of the display (the third line) as you complete any key sequence that has a matching HP-IB command. See figure 2-1. This command is the same as those generated in your recorded program during a recording session.

At power-up, the default status of GPIB Echo is off. To turn on GPIB Echo, press:

[Local/HP-IB]
[GPIB ECHO **ON** OFF]

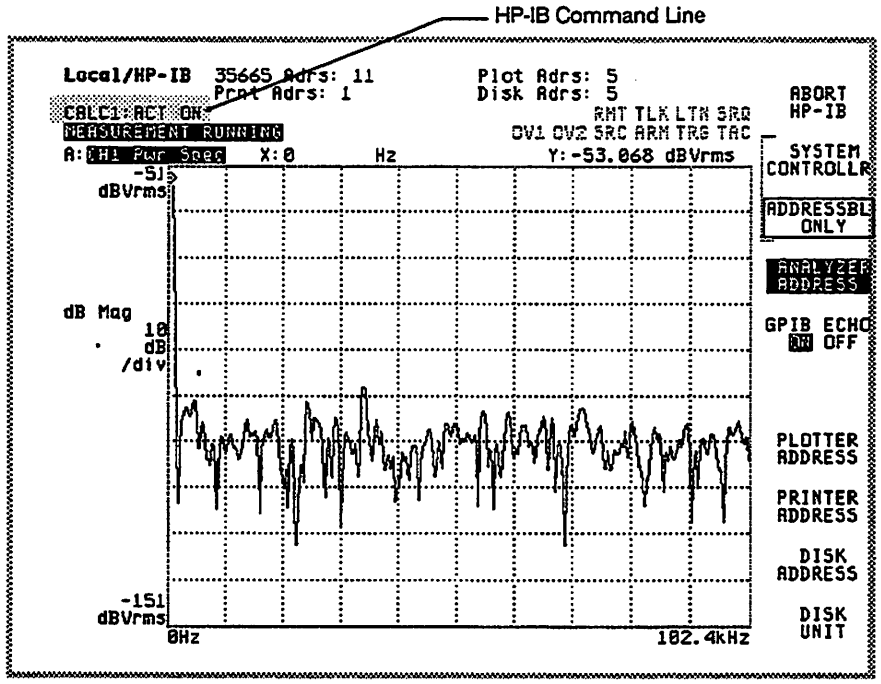

Figure 2-1. GPIB Echo

Program Buffers and the Active Program

You can record, load, or save your programs into any one of five available locations in memory called "program buffers." Only one of these buffers is active at any one time. This is called the active program. The active program defaults to Program 1 at power-up.

You can run any one of the five possible programs by pressing the softkey corresponding to that program in the [**BASIC**] menu. It becomes the active program. The name of the active program appears in the status line at the top of the screen. See figure 2-2.

You can also run the *currently active program* by pressing the [RUN PROGRAM] softkey in the [INSTRUMNT BASIC] menu.

Selecting the Active Program

Selecting any one of five possible programs to be the active program is simple. From the [**BASIC**] menu, press [INSTRUMNT BASIC] and then press [SELECT PROGRAM]. The five program softkeys are presented. Press the softkey of the program you want to be the active program. See figure 2-2.

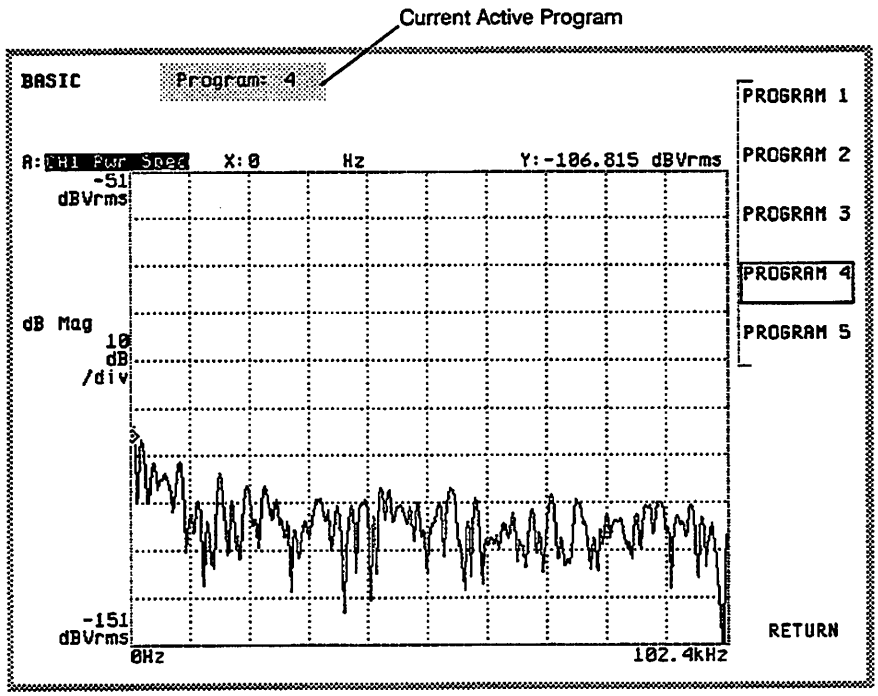

Figure 2-2. The [Select Program] Menu

To record a front-panel operation into a non-active program, select the program with the [SELECT PROGRAM] menu, then press the [ENABLE RECORDING] softkey.

For example, the following keystrokes, select Program 4 as the active program and record a single front-panel operation.

[BASIC]
[INSTRUMENT BASIC]
[SELECT PROGRAM]
[PROGRAM 4]
[ENABLE RECORDING]

[DISP FORMAT]
[UPPER LOWER]

[BASIC]

Program 4 is the active program and contains a three line program. Although it is very short, it is a representative recorded program.

Changing a Program Label

You can change the softkey label for the active program. Press the [LABEL PROGRAM] softkey in the [INSTRUMNT BASIC] menu. The alpha entry menu appears on the display. The current label appears in an entry window at the top of the screen. Use the menu and the alpha keys (or your keyboard) to change the name. When you finish editing the name, press [ENTER].

You can change the softkey label for each of the five program buffers by first selecting the appropriate program as the active program.

3

Controlling Programs

You can start, pause and stop an HP Instrument BASIC program from the HP 35665A front panel using various hardkeys and softkeys. This chapter describes how to control an HP Instrument BASIC program.

HP-IB commands can control HP Instrument BASIC programs over the external bus. You can use an external controller to run HP Instrument BASIC programs. For information on running, pausing and stopping programs from an external controller see chapter 8, "Interfacing with the HP-IB."

Running and Continuing a Program

The [**BASIC**] menu displays five program softkeys, corresponding to the five program buffers. See figure 3-1. The status line at the top of the screen indicates which program is currently the active program. These softkeys are initially labeled [RUN PROGRAM 1] through [RUN PROGRAM 5], but can be changed to display your own labels (see "Changing a Program Label" in chapter 2). Pressing one of these softkeys selects that program as the currently active program and runs the stored program. This menu gives you immediate access to running any one of five HP Instrument BASIC programs.

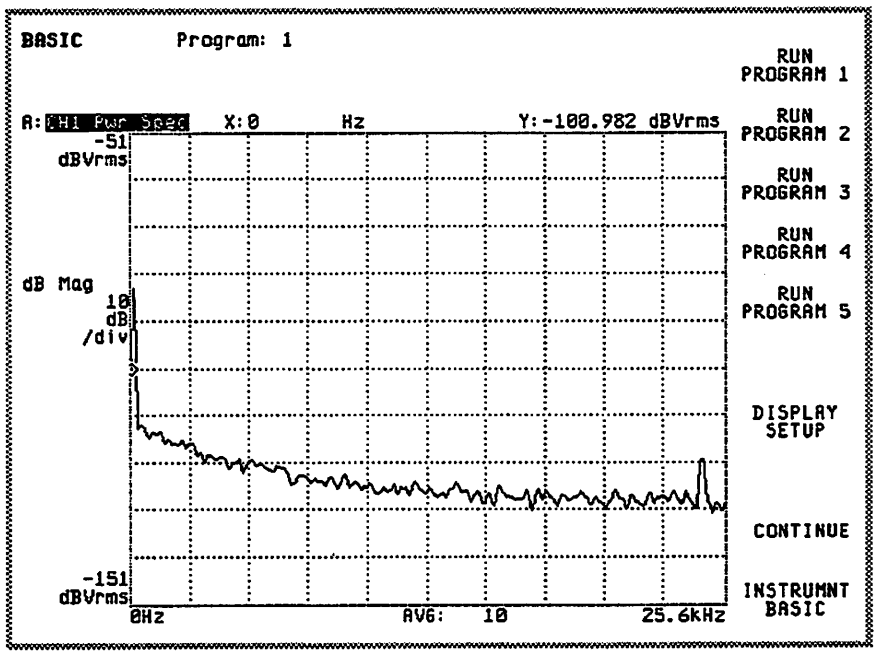

Figure 3-1. The [BASIC] Menu

In the [**BASIC**] menu you will also find the [INSTRUMNT BASIC] softkey. Pressing this key brings you into the HP Instrument BASIC operating environment (see figure 3-2). This menu only applies to the currently active program.

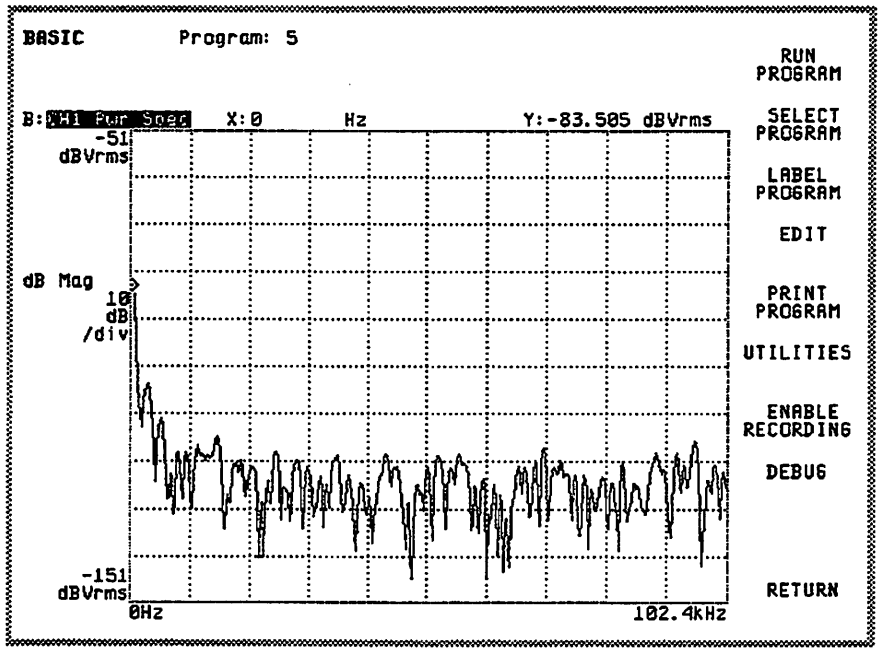

Figure 3-2. The [INSTRUMNT BASIC] Menu

To run the active program, press:

[**BASIC**]
[INSTRUMNT BASIC]
[RUN PROGRAM]

There is a [RUN] key in the [**BASIC**] [INSTRUMNT BASIC] [DEBUG] menu. This key allows you to run the currently active program during program debugging (see chapter 6, "Debugging Programs"). Both of these softkeys perform the same HP Instrument BASIC RUN command.

The RUN command is executed in two phases: prerun initialization and program execution.

The prerun phase consists of:

- Reserving memory space for variables specified in COM statements (both labeled and blank).
- Reserving memory space for variables specified by DIM, REAL, INTEGER, or implied in the main program segment. Numeric variables are initialized to 0; string variables are initialized to the null string.
- Checking for syntax errors that result from multiple program statements. Incorrect array references and mismatched parameters or COM lists are examples of these types of syntax errors.

After the prerun phase successfully completes, the program continues executing until one of the following events occurs:

- The program encounters an END or STOP statement.
- The program encounters a PAUSE statement.
- You press the [**Local/HP-IB**] hardkey to stop the program.
- You press the [**BASIC**] hardkey to pause the program.
- You press the [**Preset**] hardkey to stop the program.

Pausing a Program

You can pause a program by pressing the [**BASIC**] hardkey. Another way to pause a program is to insert a PAUSE statement into your program. (Refer to chapter 5, "Developing Programs," to learn how to insert statements into your recorded program.) In either case, the analyzer temporarily stops executing the program.

To continue a paused program, press the [CONTINUE] softkey in the [**BASIC**] menu or in the [INSTRUMNT BASIC] [DEBUG] menu. Continuing a paused program resumes the operation from where it was paused in the program. The program retains the values for any variables.

Pausing a program does not close any files that have been opened by the program. You will not be able to perform any of the following disk operations from the front panel after pausing a program that has left a file open on that medium:

- RENAME FILE
- DELETE FILE
- DELETE ALL FILES
- COPY FILE
- COPY ALL FILES
- FORMAT DISK

An HP Instrument BASIC "RESET" closes all open files. Press the [**Local/HP-IB**] hardkey while the program is running or press the [RESET] softkey in the [**BASIC**] [DEBUG] menu when the program is paused. There is one exception. An HP Instrument BASIC "RESET" does *not* close a file if it is the device for the PRINTER IS statement.

Keystroke recorded programs do not open files and therefore avoid this problem.

Stopping a Program

To completely stop a program, press the [**Local/HP-IB**] hardkey at any time while the program is running. This causes an HP Instrument BASIC "RESET." If the analyzer is under remote control, pressing the [**Local/HP-IB**] hardkey *twice* also resets an HP Instrument BASIC program. (The first press brings the instrument back to local and the second press resets the program.) A STOP statement in your program terminates the program but does *not* perform the reset operation.

Note

While the program is executing an INPUT statement, pressing the [**Local/HP-IB**] hardkey brings the program under local (front panel) control. This enables the front panel's alpha keys. Pressing [**Local/HP-IB**] again, enters a "U" on the input line. In this case, press the [**Preset**] hardkey to abort the program.

Variables retain their value after an HP Instrument BASIC RESET. Press [**BASIC**] [DEBUG] [EXAMINE VARIABLE] to examine variable values.

Pressing [**Preset**] stops a running program. It will also set the PRINTER IS device to the display (CRT).

For more information on the PAUSE and STOP statements see the "HP Instrument BASIC Language Reference" section of the *HP Instrument BASIC Users Handbook.*

4

Saving and Recalling Programs

HP Instrument BASIC programs can reside in memory, on disk, or in an external computer.

Transferring Programs

From the front panel you can transfer a program between memory and disk with the [**Save/Recall**] menus. Within a program, you can use the GET, SAVE and RE-SAVE statements to transfer program files to and from disk. The HP 35665A has an autoload feature which automatically recalls and runs a program from disk at power-up.

You can transfer a program file between the analyzer and an external controller. You can keystroke record a measurement sequence and then upload the program to the external controller for further editing. Programs developed on an external controller can be downloaded as well. Chapter 8, "Interfacing with the HP-IB," describes methods of transferring programs between the HP 35665A and an external controller.

This chapter describes transferring HP Instrument BASIC programs between program memory and the HP 35665A's volatile, non-volatile, internal, and external disk drives. The autoload feature is described at the end of this chapter.

Disk Formats and File Systems

To successfully transfer an HP Instrument BASIC program file, you must first understand the disk formats and file systems recognized by the HP 35665A.

HP Instrument BASIC in the HP 35665A recognizes two disk formats: LIF (Logical Interchange Format), and DOS (Disk Operating System). Formatting or initializing a disk determines the format of a disk or file system.

A LIF disk contains only one directory. This format should be used to exchange programs and data with other HP BASIC computers.

A DOS disk has a hierarchical structure of directories. The DOS format should be used to exchange data with DOS computers.

The *HP Instrument BASIC Users Handbook* refers to a third format, HFS (Hierarchical File System). The HP 35665A does not support HFS.

File Types

The HP 35665A supports three file types:

- ASCII
- BDAT
- untyped (referred to as DOS or HP-UX files)

"File type" is independent of disk format. ASCII, BDAT and untyped files exist on either LIF or DOS disks. Untyped files appear as HP-UX in the catalog of a LIF disk, or as DOS in the catalog of a DOS disk. To view the catalog of the analyzer's default disk, press:

```
[ Disk Utility ]
   [ CATALOG ON OFF ]
```

In HP Instrument BASIC, the "CREATE ASCII" command creates an ASCII file, the "CREATE BDAT" command creates a BDAT file, and the "CREATE" command creates an untyped file.

For more information, refer to "Disk I/O" in chapter 11 and the "Data Storage and Retrieval" chapter in the "HP Instrument BASIC Programming Techniques" section of the *HP Instrument BASIC Users Handbook.*

Note

The [COPY FILE] operation in the [Disk Utility] menu *does not* translate file types when you copy files across different file systems (DOS/LIF). Verify you are using the appropriate file type *before* copying a file.

DOS Conventions

On DOS disks, file names must conform to DOS conventions. File names are limited to 8 characters followed by a period and a three character extension. The period and extension are not required. File names are *not* case sensitive. For example, the following file names are equivalent:

```
PROG.ASC = Prog.ASC
```

The HP 35665A does not allow "wild card" characters in file names. You can use a wild card in disk operations.

The HP 35665A recognizes a directory path. For example:

```
ASSIGN @File to "\DATA\TEST1\BEFORE"
```

opens the file named "BEFORE" in the sub-directory "TEST1" under the directory "DATA." Use a "\" or a "/" to separate directory and file names. The file specifier can include a directory path. However, you cannot use the HP 35665A front panel keys to create a directory. You can create a directory with the CREATE DIR statement.

Using a DOS Disk to Transfer Data With a PC

You can transfer data from the HP 35665A to an IBM-compatible personal computer by writing an HP Instrument BASIC program that outputs the data to a DOS disk.

To ensure a successful transfer, remember:

- *Specify the correct disk format.* Either format the disk on the PC, or use the HP 35665A and the correct format option with the [FORMAT DISK] operation in the [**Disk Utility**] menu. You can determine the format of a disk by looking at its catalog. Press [**Disk Utility**] [CATALOG **ON** OFF].

- *Create untyped DOS files with the CREATE command.* Untyped files on DOS disks are extensible. They "grow" to the size needed. ASCII and BDAT files are not extensible. They usually cannot be read by other DOS applications.

- *Open files with the FORMAT ON option of the ASSIGN command.* FORMAT ON directs HP Instrument BASIC to store the data as ASCII characters.

You can also transfer HP Instrument BASIC programs to a personal computer using a DOS disk. The GET statement recalls HP Instrument BASIC programs from a DOS file into the analyzer's memory. The [RE-SAVE PROGRAM] softkey and the SAVE statement create untyped DOS files on a DOS disk.

For additional information about input and output operations, refer to "Front Panel Operations versus Keyword Statements" later in this chapter and "Disk I/O" in chapter 11, "Instrument Specific HP Instrument BASIC Features."

LIF Conventions

On LIF disks, file names must conform to LIF conventions. File names are limited to 10 characters which include all characters except ":", "<"and "|". Some LIF implementations do not allow lowercase letters.

LIF does not allow directories but you can label the disk with a volume name. The volume name is assigned at initialization.

Using a LIF Disk to Transfer Data with an HP BASIC computer

You can transfer data from the HP 35665A to an HP BASIC computer by writing an HP Instrument BASIC program that outputs the data to a LIF disk.

To ensure a successful transfer, remember:

- *Specify the correct disk format.* Use the [FORMAT DISK] operation in the [**Disk Utility**] menu. You can determine the format of a disk by looking at its catalog. Press [**Disk Utility**] [CATALOG **ON** OFF].

- *Use the CREATE ASCII command to create an ASCII file.* A LIF protect code is not allowed on an ASCII file.

- *Use the CREATE BDAT command to create a BDAT file.* HP Instrument BASIC allows and supports a LIF protect code on a BDAT file.

- *Open files with the FORMAT option of the ASSIGN command.* FORMAT ON directs HP Instrument BASIC to store the data as ASCII characters. FORMAT OFF, which is faster and takes less space, defaults to BDAT representation.

You can also transfer HP Instrument BASIC programs to an HP BASIC computer using a LIF disk. The GET statement recalls HP Instrument BASIC programs from a LIF file into the analyzer's memory. The [RE-SAVE PROGRAM] softkey and the SAVE statement create ASCII files on a LIF disk.

For additional information about input and output operations, refer to "Front Panel Operations versus Keyword Statements" later in this chapter and "Disk I/O" in chapter 11, "Instrument Specific HP Instrument BASIC Features."

Program Buffers

The HP 35665A has five program buffers in memory. The BASIC menu accesses each program buffer with a set of softkeys initially labeled [PROGRAM 1] through [PROGRAM 5]. Each buffer can hold a separate program.

You can transfer programs between any disk and any program buffer but *you cannot directly transfer programs between buffers.* A program that is recalled into a buffer overwrites the previous contents of that buffer.

You can, however, append program files to build one functional program. See "Appending Program Files From Disk," later in this chapter.

Memory

HP Instrument BASIC in the HP 35665A supports four mass storage devices:

- A volatile RAM disk (:MEMORY,0,0).
- A non-volatile RAM disk (:MEMORY,0,1).
- An internal disk drive (:INTERNAL).
- An external disk drive (Hewlett-Packard Subset/80) (:EXTERNAL,7XX, uu).

To specify the default storage device, use the MASS STORAGE IS command.

Front Panel Operation versus Keyword Statements

There are two ways to transfer program contents between disk and memory. You can transfer programs by either:

- Using the [**Save / Recall**] key in the SYSTEM group on the front panel

or

- Using the keyword statements SAVE, RE-SAVE and GET

The choice of which to use requires some knowledge of the advantages of each as well as your own particular requirements. Both methods are discussed in the following section.

The [Save / Recall] Menu

With the [**Save/Recall**] menu you can perform a variety of disk and file operations, as well as transfer complete programs between any of the five program buffers and any disk file. These menus have the following advantage:

- You can transfer programs directly between any file on disk and any of the five program buffers.
- The analyzer allocates memory automatically when you recall a program.
- The utilities are similar to the other save and recall operations in the analyzer.
- You can select a file in the catalog without typing in the name.

The Keyword Statements (SAVE, RE-SAVE and GET)

In an HP Instrument BASIC program, the keyword statements SAVE, RE-SAVE, and GET, save all or part of that program to disk. They also merge a program with a program from disk.

The SAVE, RESAVE, and GET keyword statements have the following advantages over the [**Save/Recall**] menu:

- You can store parts of a program to disk.
- You can recall programs and append them at any line in the currently active program.
- They are familiar to HP BASIC programmers.

Saving a Program to Disk

To save the current contents of the program buffer to a disk file, use the [**Save / Recall**] menus. This is the same system used for all disk access in the HP 35665A.

If you are saving a program to a new file name, press:

[**Save / Recall**]
[SAVE MORE]
[SAVE PROGRAM]

Type the name of the disk file in the entry window. You can use a keyboard or the front panel alpha keys. HP Instrument BASIC programs are stored as ASCII files on LIF disks and as untyped files on DOS disks.

Re-saving a program is similar to saving a file to a disk. In this case however, the disk already contains a file with the same file name. To overwrite the file, use [RE-SAVE PROGRAM] in the [SAVE MORE] menu. Enter the file name with your keyboard or the front panel alphanumeric keys.

To make the re-save process easier, use the disk catalog to select a file name. The catalog describes the contents of the default disk. To use the catalog, press the HP 35665A keys as follows:

1. [**Save / Recall**]
2. [CATALOG **ON** OFF]
3. Use the knob to highlight the desired file name. The name appears in the entry window.
4. [SAVE MORE]
5. [RE-SAVE PROGRAM]

The analyzer automatically re-saves the file with the file name you selected.

Note

The [RE-SAVE PROGRAM] softkey overwrites the contents of the file. No backup file is created; the contents of the existing file are lost.

Recalling a Program from Disk

When you recall a program file from the disk, it is loaded into the active program buffer. Any program recalled to the program buffer using the [**Save / Recall**] menus *overwrites* the current contents of the active program buffer.

To recall a program file from the disk to the active program buffer, press the HP 35665A keys as follows:

1. [**Save / Recall**]
2. [RECALL MORE]
3. [RECALL PROGRAM]
4. Enter the file name in the entry window.
5. Press [ENTER].

As with any recall operation, you can use the catalog. Press the HP 35665A keys as follows:

1. [**Save / Recall**]
2. [CATALOG **ON** OFF]
3. Move the knob to select the file name. The name appears in the entry window.
4. Press [RECALL MORE].
5. [RECALL PROGRAM]

The recalled program file is entered into the program buffer one line at a time and checked for syntax errors. Lines with syntax errors are commented out. The HP Instrument BASIC syntax error is displayed briefly in a pop-up message window. The error message is also written to the CRT. See chapter 5, Developing Programs, for information on allocating display partitions to view error messages.

Memory is allocated for the program's variables and working space (called the stack). When you use the [**Save / Recall**] menus to recall a program, memory is allocated automatically. For certain kinds of programs, the memory size may need to be increased.

See chapter 5 for more information on memory size.

Appending Program Files from Disk

To append program files from disk to the current program in memory, use the GET statement *within a program.* The GET statement recalls a specified file from the disk and appends it at a specified line in the current program (or at the beginning of the program if a line is not specified).

The following example program appends three program files to itself to build one functional program. It demonstrates how to merge files. It also provides a set of error-handing routines for your recorded programs.

The example program builds a shell composed of:

- an initialization program section.
- a typical keystroke recorded section.
- a cleanup section that contains error-traps and timeout-traps.

The core five line program (program lines 40 - 80) chains the other three programs segments to itself. *These five program statements must be deleted or commented out before you run the program.*

All of these files are on the HP 35665A Example Programs disk:

- SHELLBEG provides the setup and initialization.
- SHELLDEM is a typical keystroke recorded program.
- SHELLEND provides error-handling routines and cleanup.
- SHELLCHA pulls all files together using GET statements.

The file SHELLCHA contains the following program:

```
10 ! Program demonstrates chaining program segments
20 ! NOTE: Delete Lines 20 thru 80 immediately after
30 !       running SHELLCHA program
40 GET "SHELLBEG:,4",X,50
50 GET "SHELLDEM:,4",X,60
60 GET "SHELLEND:,4",X,70
70 DISP "Delete lines 20 - 80 before running program"
80 GOTO Endlabel
90 X:END
```

Line 40 performs a GET of the file "SHELLBEG" from the internal disk drive. It appends the file at line 90 (labeled "X:") and overwrites that line. It then instructs the program to continue at line 50.

The "SHELLBEG" file has a label "X:" as its last line. The program in memory also has that same label as its last line. Line 50 performs a GET of the file "SHELLDEM," appends it at the current label "X:" and then continues the program at line 60.

The SHELLDEM file also contains a label "X:" as its last line. Line 60 of the SHELLCHA program performs a GET, which appends the SHELLEND program file to the end of the program in memory.

Finally, line 80 of the SHELLCHA program skips to the label "Endlabel." This label, which was at the end of the SHELLEND file, is now at the end of the program in memory. The program would go on to execute the SHELLBEG program if the ENDLABEL statement had been omitted. That is, without the ENDLABEL statement, the program would run itself immediately after appending the three program files.

Note

Remember to comment out or delete program lines 10-80 before running the combined program.

To use the SHELLCHA program with your own recorded program do the following:

1. Insert the label "X:" in the line containing the END statement of your recorded program. Make sure you have not used the label "X:" elsewhere in your program.
2. Recall the SHELLCHA program and change the file name in line 50 from "SHELLDEM" to the name of your program file.

Autoloading a Program

HP Instrument BASIC allows you to automatically load one or more programs and run a designated program when you turn on the analyzer. To make an autoloading program, save it to the nonvolatile RAM disk or to a floppy disk in the internal drive with one of the following names:

- AUTO_BAS
- AUTO_BA1
- AUTO_BA2
- AUTO_BA3
- AUTO_BA4
- AUTO_BA5

At power-up, HP Instrument BASIC searches the internal disk drive and then the nonvolatile RAM disk for files with these special names. It searches for files in the order listed above, but it does not search for AUTO_BA1 if AUTO_BAS is found. If AUTO_BAS is found, it is loaded into the first program buffer and executed after all other programs have been loaded. If AUTO_BA1 through AUTO_BA5 are found, they are loaded into the first through fifth program buffers—but they are *not* executed.

Since the AUTO_BAS program is run after all programs are loaded, you may find it useful to have this program re-label the softkeys that run the loaded programs. The HP-IB command "PROG:EXPL:LAB" is used to re-label these softkeys. For example, the following statement changes the first program buffer's softkey label to "FINAL TEST":

```
OUTPUT 800; "PROG:EXPL:LAB PROG1,'  FINAL  TEST'"
```

To disable automatic loading of the AUTO_BA* files, press the [**Preset**] hardkey while you turn on the analyzer.

Developing Programs

Overview

For many applications, you can easily record and run programs without altering the program code that is generated with keystroke recording. However, with some knowledge of the HP Instrument BASIC language and the program development capabilities in the HP 35665A, you can add immeasurable power to your recorded programs. You can also create programs without using the keystroke recording feature.

This chapter describes the operation of the keys under the [INSTRUMNT BASIC] menu. See figure 5-1. At the end of the chapter the [DISPLAY SETUP] softkey (in the [**BASIC**] menu) is discussed. This softkey presents a menu that lets you manage a part of the screen display for output from the program.

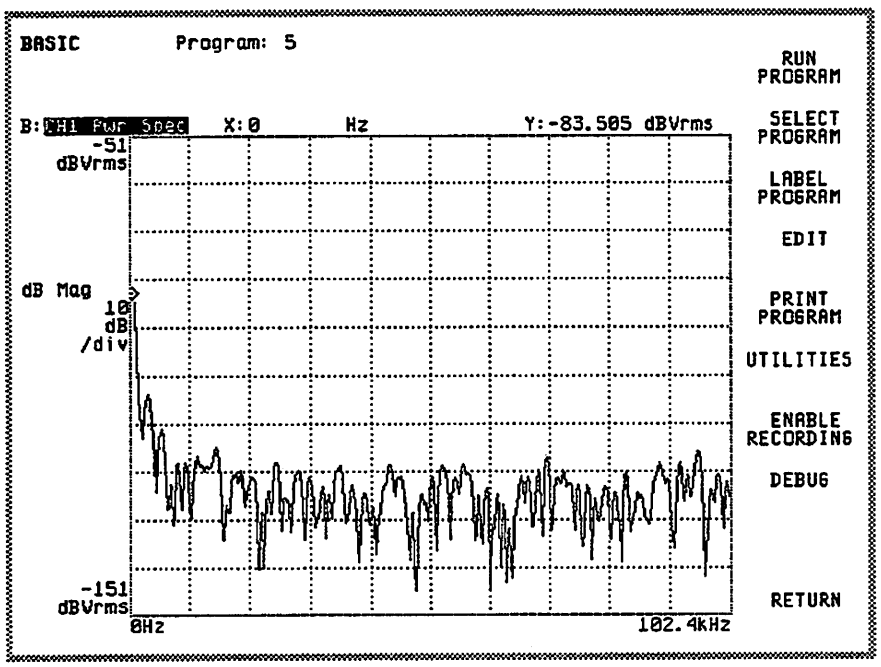

Figure 5-1. The [INSTRUMNT BASIC] Menu

The ability to change and enhance your program and its operating environment is found primarily under the [EDIT] and [UTILITIES] menus.

- Pressing [EDIT] places you in the HP Instrument BASIC editor. You can make changes to your program on a line-by-line basis using a keyboard or the front panel alpha-numeric keys.

- Pressing [UTILITIES] presents a menu of helpful utilities. You can make global changes to the program and its environment. You can renumber lines, allocate memory size, and remove the program.

- Pressing the [PRINT PROGRAM] softkey prints a hard copy program listing to an attached printer.

Using the HP Instrument BASIC editor

The HP Instrument BASIC editor allows you to create and alter program text. If you are familiar with the HP Series 200/300 BASIC editor, you will find it similar. If not, you should find the HP Instrument BASIC editor easy to learn and to use. This section tells you how to enter and edit an HP Instrument BASIC program.

To start the editor, press the [EDIT] softkey in the [INSTRUMNT BASIC] menu. The program, if one exists, usually appears on the display with the cursor on the first line of the program. If the program buffer is empty, the first line number (10) appears with the cursor positioned to enter text.

The current program line (the line containing the cursor) always appears as two lines on the screen, allowing you to enter up to 108 characters. The other lines display the first 51 characters (excluding line numbers).

The first 6 columns of each line are a numeric field specifying the program line number. Line numbers are right justified. Program lines are automatically numbered by the editor. You can manually edit the current line number to copy or move it a to different location in the program. Line numbers can also be renumbered in blocks with the [UTILITIES] [RENUMBER] softkey menu. Line number range from 1 to 32766.

Once in the [EDIT] menu you can use your keyboard or the front panel alpha-numeric keys.

Using the HP Instrument BASIC Editor With a Keyboard

Using a keyboard makes developing HP Instrument BASIC programs easy.

All of the "typewriter" keys are enabled. Letters can be entered in lower or upper case. All punctuation marks and special characters can be entered using the HP approved PC keyboard. See figure 5-2.

A Typewriter keys
B Cursor keys
C Numeric keys
D Function keys
E Computer control keys
F Indicator lights

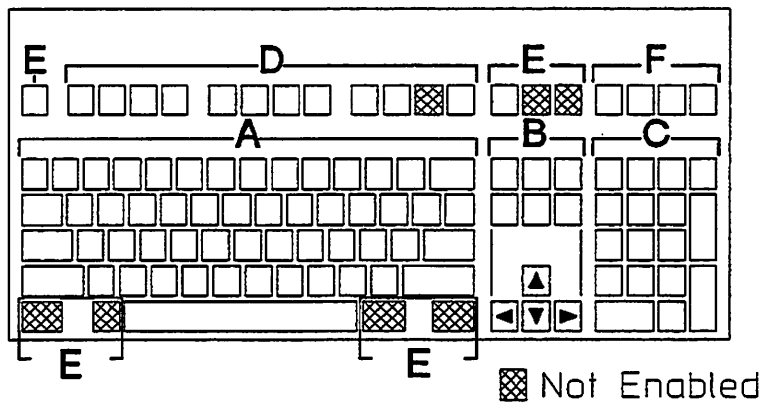

Figure 5-2. Using a keyboard with the HP 35665A.

The [*Enter*] key is used to store each line of program code and completes each alpha-numeric entry. The analyzer checks the line for syntax errors. If it detects an error, a pop-up message window displays the syntax error. If the analyzer does not detect an error, it stores the line.

Note

If you edit or enter text on the current program line and then move off the line without pressing the [*Enter*] key, all editing on the line is lost.

The [*Tab*] inserts two spaces. Pressing [*Shift*] [*Tab*] moves the cursor backwards two spaces.

The HP 35665A softkey menus load into the keyboard function keys, [*F1*] through [*F10*]. The [**Help**] hardkey loads into [*F12*]. [*F11*] is not enabled. See figure 5-3.

The "cursor" keys are enabled. The arrow keys indicate the direction in which they move the cursor.

The [*Home*] key moves the cursor to the beginning of the current line. The [*End*] key moves the cursor to the end of the current line.

The [*Page Up*] key moves the cursor a maximum of 15 lines upward. The [*Page Down*] key moves the cursor a maximum of 15 lines downward.

The [*Insert*] key inserts a new line of text. To get out of the insert mode, press the [*Insert*] key again or move the cursor off the current line. Remember, to save an edit you must press [*Enter*] while the cursor is on the current line.

The [*Delete*] key erases the character where the cursor is positioned. In addition, all characters to the right of the deleted character move one character to the left.

Pressing the [*Shift*] [*Delete*] keys deletes the current program line.

Pressing the [*Alt*] [*Delete*] keys ([*Alt Gr*] [*Delete*] keys on a non-U.S. English keyboard) deletes all characters from the current cursor position to the end of the line.

The [*Print Screen*] key is enabled. You can print the entire screen (excluding the softkey menu text) to an attached printer.

The [*Alt*] key is not enabled except when used to preset the analyzer.

Key presses made with the keyboard that have no meaning in a given operating context are ignored, just as they are when pressed from the front panel.

Caution

Pressing the [*Del*] key with the [*Alt*] key and the [*Ctrl*] key, presets the analyzer. (Just like a soft reboot in an IBM-compatible PC!)

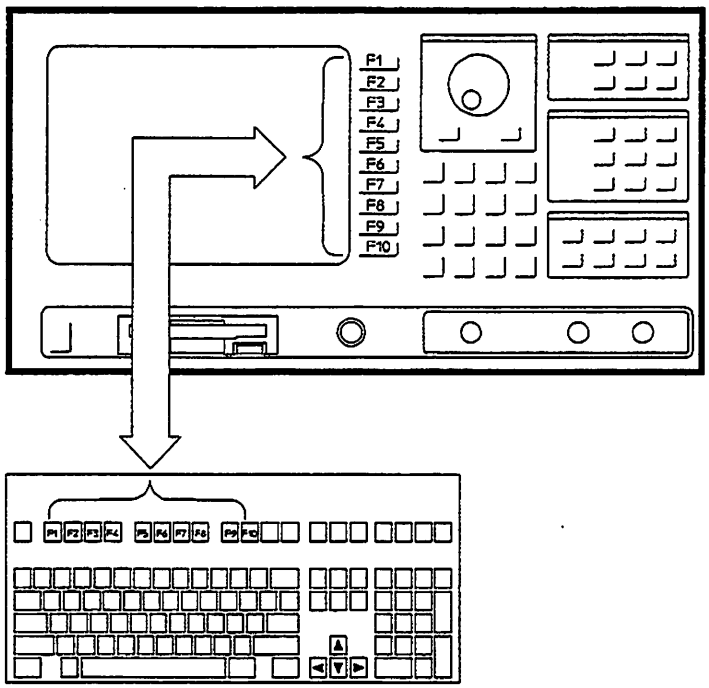

Figure 5-3. Mapping of HP 35665A Softkeys

To end an editing session, press the [*F10*] key, which corresponds to the [END EDIT] softkey. This returns you to the [INSTRUMNT BASIC] menu.

Connecting your keyboard

To connect the keyboard to the HP 35665A, plug the round connector into the front of the analyzer. See figure 5-4.

Caution Use only the HP approved keyboard on this product. HP does not warrant damage or performance loss caused by a non-HP approved keyboard.

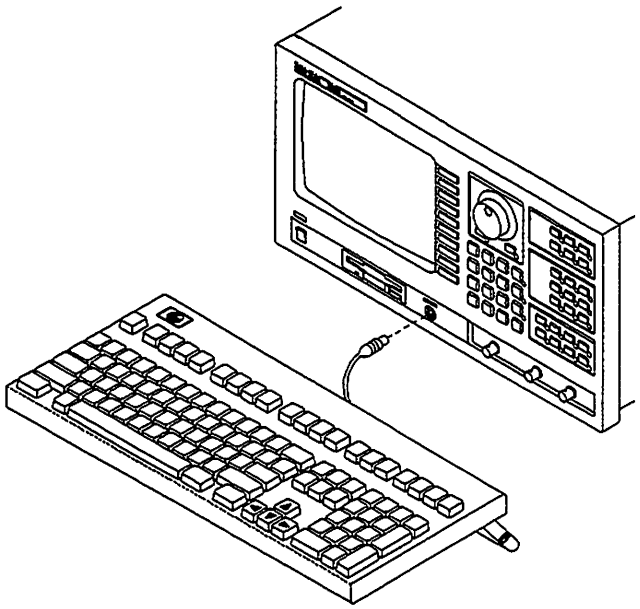

Figure 5-4. Connecting the keyboard.

Note

Some versions of the HP 35665A have the keyboard outlet on the analyzer's rear panel.

If you are using an international keyboard, specify the type of keyboard with the [Select Keyboard] softkey in the [**System Utility**] menu.

Using the [EDIT] Softkeys

The [EDIT] menu contains the softkeys shown in figure 5-5.

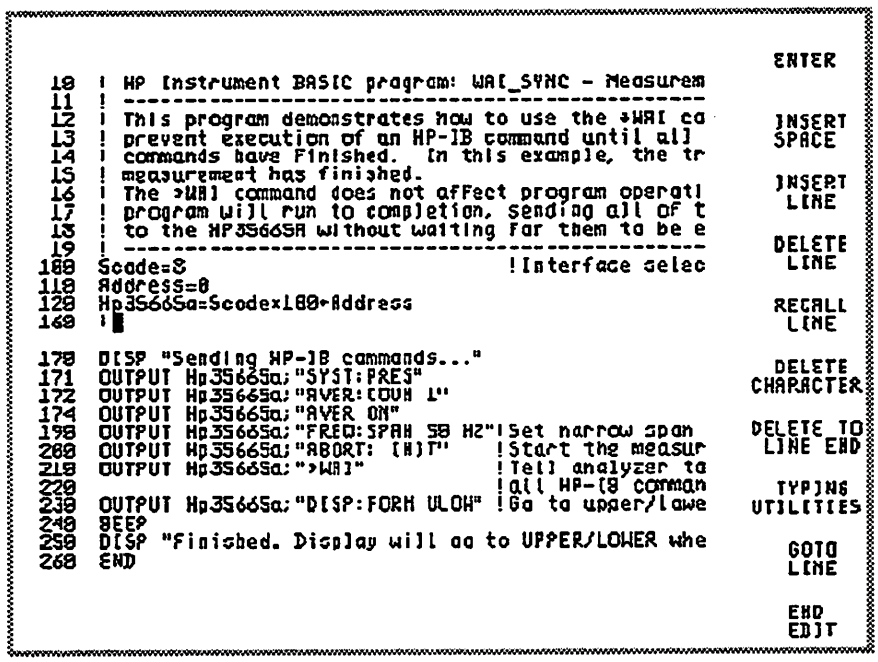

Figure 5-5. The [EDIT] Menu

You can move the cursor around in the program using the knob on the front panel or the cursor keys on the keyboard. When you get to a line you want to change, make the change and press the [ENTER] softkey or the [*Enter*] key on the keyboard. The analyzer checks the line for syntax and then stores it if the syntax is correct.

Getting Around in the Program

You can move the cursor from line to line within an existing program by:

- Using the knob
- Using the [GOTO LINE] softkey to jump directly to a specific line number or label
- Using the [ENTER] softkey (when not in insert mode) to step one line at a time
- Using the [*Enter*] key on the keyboard
- Using the cursor keys on the keyboard

Using the Knob

You can move the cursor in the EDIT mode with the knob on the front panel.

The line that the cursor is on is always the edited line. Rotating the knob clockwise on the currently edited line moves the cursor to the right. Rotating the knob counterclockwise moves the cursor to the left. When the cursor is at the end (far right) of the edited line, turning the knob clockwise moves the cursor down to the end of the next line. Conversely, when the cursor is at the beginning (far left) of the edited line, turning the knob counterclockwise moves the cursor to the beginning of the preceding line.

Using GOTO LINE

To jump immediately to any line or label in the program press the [GOTO LINE] softkey ([*F9*] on the keyboard). Enter the line number or the label of the line into the entry window and press [ENTER]. To specify a label, use the keyboard or use the front-panel keys and the optional [(_)] underscore softkey. You can enter the label in capital letters and it automatically converts to the proper case.

If the specified line exists, it appears in the middle of the display as the current program line. If you have specified a line number that doesn't exist, the cursor is placed on the line number closest to it. Specifying a non-existent line label generates an error message, "Line not found in this context."

A quick way to go to the last line of the program is to enter a number much larger than the largest possible program line number such as 99999 (or any number greater than 32766 or the last line number of your program).

Using the [ENTER] Softkey

You can use the [ENTER] softkey to move the cursor down one line at a time. All other softkeys that move the cursor alter program text.

Using the Keyboard

See the previous section, Using a Keyboard, for a description of getting around the program using the keyboard.

Entering Program Lines

When you finish entering or changing a program line, store it by pressing [ENTER]. The analyzer checks the line for syntax errors and converts letter case to the required form for names and keywords (HP Instrument BASIC commands). If it detects an error, a pop-up message window displays the syntax error. If no errors are detected, it stores the line.

Note

If you edit or enter text on the current program line and then move off the line without pressing [ENTER], all editing on the line is lost.

Renumbering, Copying and Moving Lines

If you want to change the line number of an edited program line, move the cursor to the line number field and enter a new line number. Changing the line number copies the line. The line does not move. To move the line, change the line number, press [ENTER] and then delete the original line.

If you want to revise and move the current line, edit the line, change the line number and then press [ENTER]. The revision only appears in the copied line.

If you change the line number and you are in insert mode, you remain in insert mode at the new line number.

When the cursor is in the line number field, entries operate in an overtype mode rather than in the insert mode as in the text portion of the program line. The [**Back Space**] hardkey in the numeric keypad moves the cursor over line numbers without deleting the number.

Inserting Spaces

Use the [INSERT SPACE] softkey to place a space at the position of the cursor. The text to the right of the cursor moves one place to the right. This softkey is located in more than one menu.

To insert a space with your keyboard, press the space bar.

Inserting Lines

You can easily insert one or more program lines above any existing line by placing the cursor on the existing line and pressing [INSERT LINE]. The [INSERT LINE] softkey toggles the insert mode on or off.

If you are using a keyboard, press the [*Insert*] key to insert a line. Pressing the function key [*F3*], also turns the insert mode on or off.

In the following example we use the [INSERT LINE] softkey to insert lines between two adjacent programs lines numbered 90 and 100.

Move the cursor to line 100 and press [INSERT LINE]. A new line, numbered 91, appears between line 90 and line 100. Press [Enter] to store the inserted line and another line appears numbered 92. If you continue to insert new lines and the inserted line number increments to 100, the current line 100 is renumbered to 101 to accommodate the inserted line.

To get out of insert mode, press [INSERT LINE] again or use the knob to move off of the current line. (Remember, any edits you make to the currently inserted line are lost if you leave insert mode without pressing [ENTER].) Make sure you have entered any changes to your final inserted line before exiting the insert mode.

Recalling Deleted Lines

If you used the [DELETE LINE] softkey or the [*Shift*] [*Delete*] keys to remove a line, the [RECALL LINE] softkey automatically recalls that line. This is useful for recovering lines deleted by mistake.

It is also useful for moving a line. Delete the line, move to the desired area of the program and press [INSERT LINE] ([*F3*] on the keyboard). Press [RECALL LINE] ([*F5*] on the keyboard) and then edit the recalled line to the current line number.

Note

Pressing [RECALL LINE] automatically aborts any changes made to the currently edited line.

Using the Front-Panel Alpha Keys

If you do not have a keyboard, you can use the alpha keys on the front panel of the HP 35665A. Nearly every hardkey is labeled with a corresponding letter of the alphabet. These may be familiar to you if you have performed any editing function on the HP 35665A, such as specifying a unique filename in a [**Save/Recall**] operation.

The alpha keys are arranged in alphabetical order from left to right, descending the front-panel hardkeys. See figure 5-6.

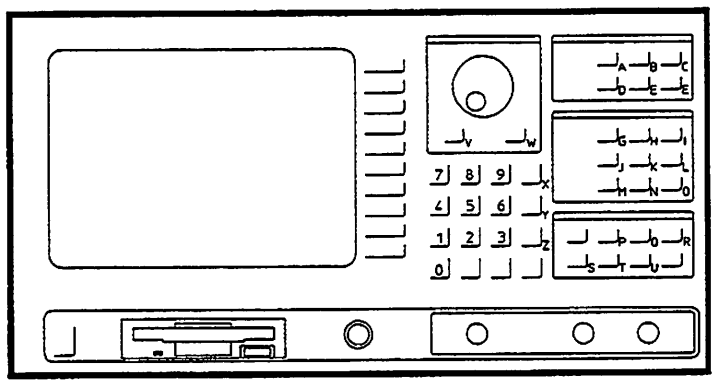

Figure 5-6. HP 35665A front panel alpha-numeric keys

You do not have to use the alpha keys to enter HP Instrument BASIC keywords. They can be entered via the [TYPING UTILITIES] menu. The front panel alpha keys are necessary to enter variable names, constants, labels and strings if you are not using a keyboard.

When in the HP Instrument BASIC editor, the front-panel keys are automatically in alpha mode. Pressing an alpha key enters the character at the cursor position in the current program line. When you exit the editor ([END EDIT]) the front-panel keys return to their labeled hardkey function.

Changing Case

The case of an alpha key is determined by the state of the [TYPING UTILITIES] [**UPPERCASE** lowercase] key. The default is uppercase. To enter lowercase letters, press the [UPPERCASE **lowercase**] key.

If you are using the keyboard, the case is determined by the [*Caps Lock*] key.

Using [TYPING UTILITIES]

The [TYPING UTILITIES] allows you to enter non-alphabetic symbols and insert HP Instrument BASIC keywords without a keyboard. See figure 5-7.

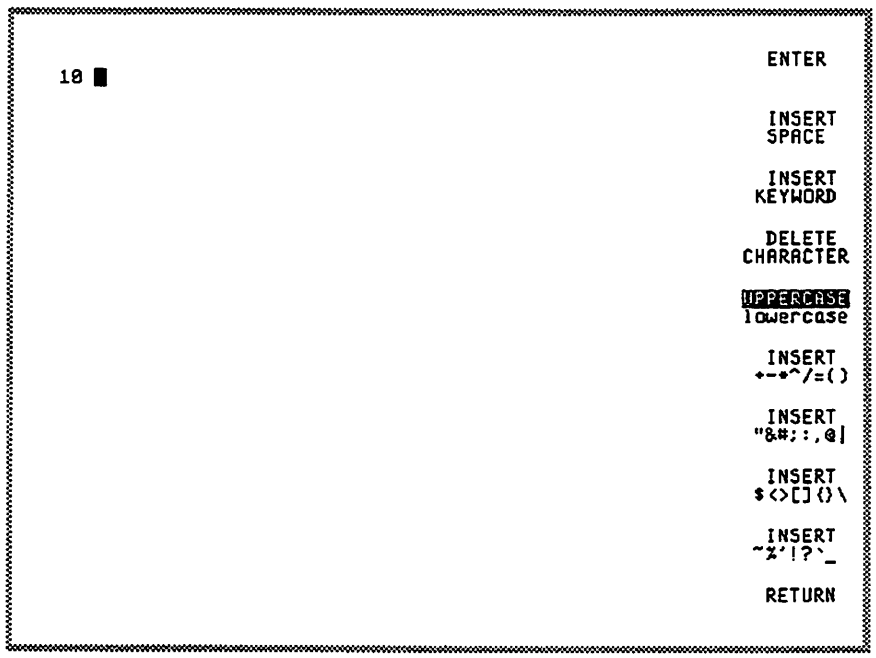

Figure 5-7. The [TYPING UTILITIES] Menu

The [ENTER], [INSERT SPACE], and [DELETE CHARACTER] keys are carried over from the [EDIT] menu. In addition, this menu contains the [**UPPERCASE** lowercase] key and the [INSERT KEYWORD] key.

Entering Symbols

Symbols are available in four menus. Each menu is labeled "INSERT" followed by a list of the available symbols in that menu. See figure 5-8.

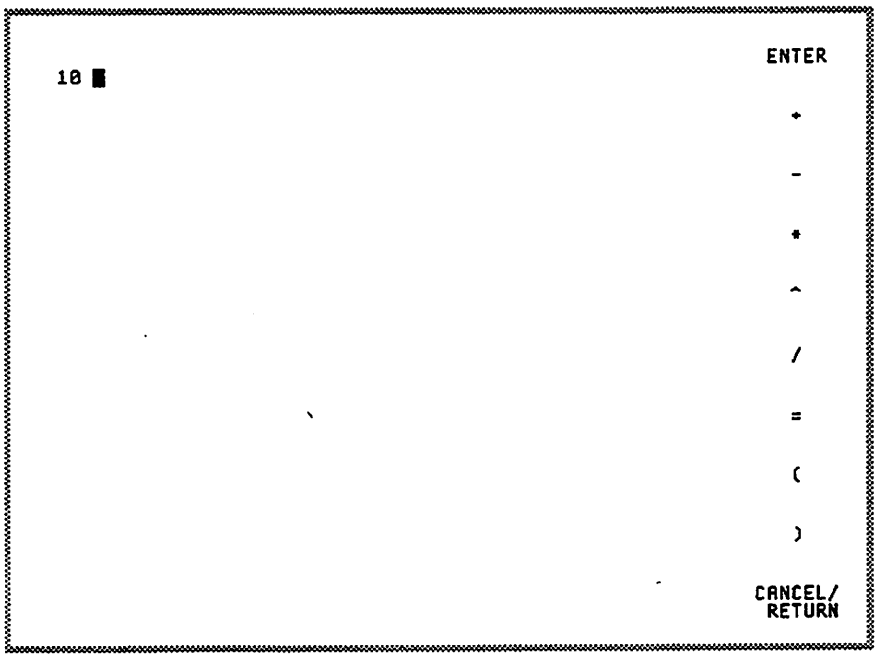

Figure 5-8. An [INSERT SYMBOL] Menu

For example, to enter an equal symbol (=), press [TYPING UTILITIES], then press [INSERT +-*^/=( )]. This brings up a softkey menu with each of the symbols listed in the label as a separate softkey. Press the softkey labeled [=].

Entering Keywords

You do not have to type an entire HP Instrument BASIC keyword if you use the [INSERT KEYWORD] softkey in the [TYPING UTILITIES] menu. When you are in the [INSERT KEYWORD] menu, pressing any alpha key presents a menu of keywords beginning with that letter.

For example, pressing [INSERT KEYWORD] and then the alpha key "A" (the [**Meas Data**] hardkey) presents a menu with the following softkeys:

- [ABORT]
- [ABS( ]
- [ACS( ]
- [AND]
- [ASCII]
- [ASN( ]
- [ASSIGN]
- [ATN( ]
- [CANCEL]

Pressing any one of these softkeys, other than [CANCEL] enters the corresponding text into the current program line. Keywords are always inserted in uppercase regardless of the current setting of the [**UPPERCASE** lowercase] softkey.

In cases where there are more than eight keywords starting with a particular letter, a softkey labeled [MORE] appears which allows you to access the rest of the keywords of that letter. When the last set of keywords is displayed, press [MORE] to get back to the first set. This allows you to cycle through all the keywords starting with a specific letter.

After pressing [INSERT KEYWORD] you can skip from one keyword menu to another simply by pressing another front-panel alpha entry key.

Notice that all keywords that require an argument are provided with the beginning parenthesis; for example, [ASN( ]. The parenthesis indicates the keyword as requiring an argument. When this keyword is selected, the keyword and both parentheses are inserted in the program line with the cursor placed automatically between the two. All keywords that require an argument are inserted this way.

To return to the previous menu without selecting a keyword, press [CANCEL].

Recording into an Existing Program

Another way to enter lines into your program is to use the keystroke recording capabilities of HP Instrument BASIC. To record measurement sequence operations into your program, move the cursor to the line where you want the recorded statements inserted. Then press [END EDIT], press [ENABLE RECORDING] and proceed with your recording as you normally would. Press [**BASIC**] to conclude the recording session as usual.

The inserted recording acts the same as if you had pressed [INSERT LINES] in the editor, and generates OUTPUT statements in insert mode.

The "ASSIGN @Hp35665a to 800" statement is *not* generated when you are recording into an existing program. The "ASSIGN @Hp35665a to 800" statement *must* be included in your program prior to any recorded OUTPUT commands. If you initially created the program using keystroke recording, this statement should already exist. If it does not exist, you will need to enter it.

Removing Program Text

The HP Instrument BASIC editor allows you to remove individual characters or entire lines. To learn how to remove the entire program see the description of the [UTILITY] [SCRATCH] softkey later in this chapter.

Deleting Characters using a keyboard

The [*Delete*] key on the keyboard erases the character where the cursor is positioned. In addition, all characters to the right of the deleted character move one character to the left.

The [*Backspace*] key also removes text. The cursor moves one space to the left and usually erases any characters in the cursor's path. It does not erase characters in the program line number field.

Pressing the [*Alt*] [*Delete*] keys ([Alt Gr] [*Delete*] keys on a non-U.S. English keyboard) deletes all characters from the current cursor position to the end of the line.

Deleting Characters using the [DELETE CHARACTER] softkey

The [DELETE CHARACTER] softkey, [*F6*] on the keyboard, removes the character under the cursor and moves all characters to the left one place. Repeatedly pressing [DELETE CHARACTER] causes text to the right of the cursor to be pulled in and deleted. The [DELETE CHARACTER] softkey functions the same in both the line number and program statement fields. However, in the line number field, only line numbers to the right of the cursor are pulled in and deleted. Program statement characters are not deleted when the cursor is in the line number field.

Another way to remove text on a line is with the [**Back Space**] key in the front panel's numeric key pad. Pressing [**Back Space**] removes the letter to the left of the cursor and moves the cursor (and all characters to the right of the cursor) one space to the left. When the cursor is on a line number, pressing the [**Back Space**] key simply moves the cursor back one position without deleting the number.

The [DELETE TO LINE END] softkey, [*F7*] on the keyboard, deletes all characters from the current cursor position to the end of the line.

Deleting Lines using a keyboard

Pressing the [*Shift*] [*Delete*] keys removes the current program line and places it in a buffer. When the current program line disappears, all subsequent lines in the display move up one line, but are not renumbered. The cursor maintains its column-relative position on the next highest numbered line.

If the [*Shift*] [*Delete*] keys are pressed when the cursor is on the last program line, the line text is removed but the line number remains with the cursor resting in the first column. This puts the editor in insert mode on the last line of the program (see "Inserting Lines"). To get out of insert mode, use the knob and move the cursor up one line.

Pressing the [*Shift*] [*Delete*] keys will *not* remove a subprogram line with a SUB keyword in it unless all program lines belonging to that subprogram are deleted first.

Deleting Lines using the [DELETE LINE] softkey

The [DELETE LINE] softkey, [*F4*] on the keyboard, removes the current program line in the same manner as pressing the [*Shift*] [*Delete*] keys on the keyboard.

To recall the last deleted line, press the [RECALL LINE] softkey, [*F5*] on the keyboard.

Using [UTILITIES]

There are some activities generally associated with editing that are located outside the [EDIT] menu, under the [**BASIC**] [UTILITIES] softkey. These editing utilities are more global in nature, rather than pertaining to single characters, words and lines as the editor does. See figure 5-9.

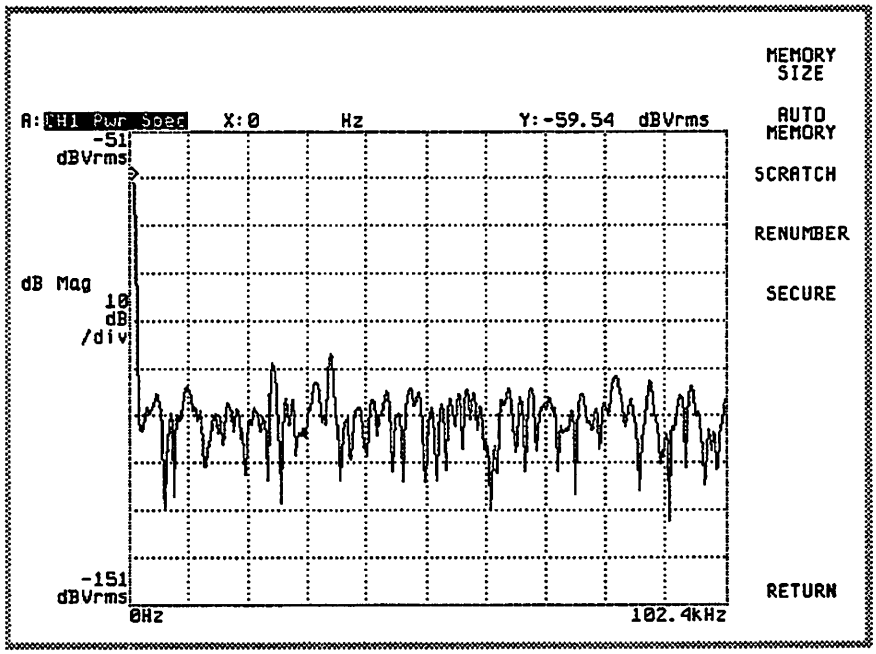

Figure 5-9. The [UTILITIES] Menu

If using a keyboard, the [UTILITIES] menu loads into the following function keys:

[MEMORY SIZE]	[*F1*]
[AUTO MEMORY]	[*F2*]
[SCRATCH]	[*F3*]
[RENUMBER]	[*F4*]
[SECURE]	[*F5*]
[RETURN]	[*F10*]

The [UTILITIES] menu is mostly composed of HP Instrument BASIC keywords that can be executed interactively, most of which directly effect the contents of the program ([SCRATCH], [RENUMBER] and [SECURE]).

Two other utility softkeys are not keywords: [MEMORY SIZE] and [AUTO MEMORY]. These utilities allow you to directly change the program's operating space.

MEMORY SIZE

Press [MEMORY SIZE] to display the amount of working space (commonly called the stack) currently allocated for the active program. The stack contains all variables not in COM as well as context information for functions and subprogram calls. The stack does not contain program code.

HP Instrument BASIC allocates the size of the stack for the most efficient use of memory resources. If you use recursive subprograms, HP Instrument BASIC may not allocate enough memory. If the analyzer runs out of stack space while the program is running it displays an error message, "Out of Memory" in a pop-up message window.

To increase the amount of memory allocated for the stack, press [MEMORY SIZE]. Enter the new amount using the numeric keys on the front panel or on the keyboard. The entry window is displayed when you press the first numeric key. Use the [EXP] softkey to enter the size using engineering notation. After entering the new memory size, press [ENTER]. HP Instrument BASIC may adjust your entry to the closest available increment of memory.

If you enter a number which exceeds the available memory, the memory size will be set to the largest available stack size. The minimum amount allocated by HP Instrument BASIC for the stack is 1122 bytes.

Memory available for HP Instrument BASIC programs is dependent upon the amount of memory space allocated for other uses. To display the usage of all of the analyzer's memory, press the [**System Utility**] hardkey then press [MEMORY USAGE]. A table displays the amount of memory allocated for all HP Instrument BASIC programs (code and program stacks) as well as memory allocated for other functions of the analyzer.

AUTO MEMORY

The [AUTO MEMORY] softkey resizes stack space automatically to fit the current active program. This is similar to the operation that occurs when a program is loaded with the [**Save / Recall**] menus. This is faster than using the [MEMORY SIZE] key and works well for most programs.

In some cases, [AUTO MEMORY] may allocate more memory than the HP Instrument BASIC program needs. Use the [MEMORY SIZE] softkey to reduce the amount of memory allocated for your program. If you receive an "Out of Memory" error when you try to run the program you can use the [MEMORY SIZE] softkey to increase the memory size. Programs that use recursive functions or subprograms may need to have memory increased manually with the [MEMORY SIZE] softkey.

SCRATCH

Pressing the [SCRATCH] softkey brings up a menu that allows you to clear the current program and/or variables. See figure 5-10.

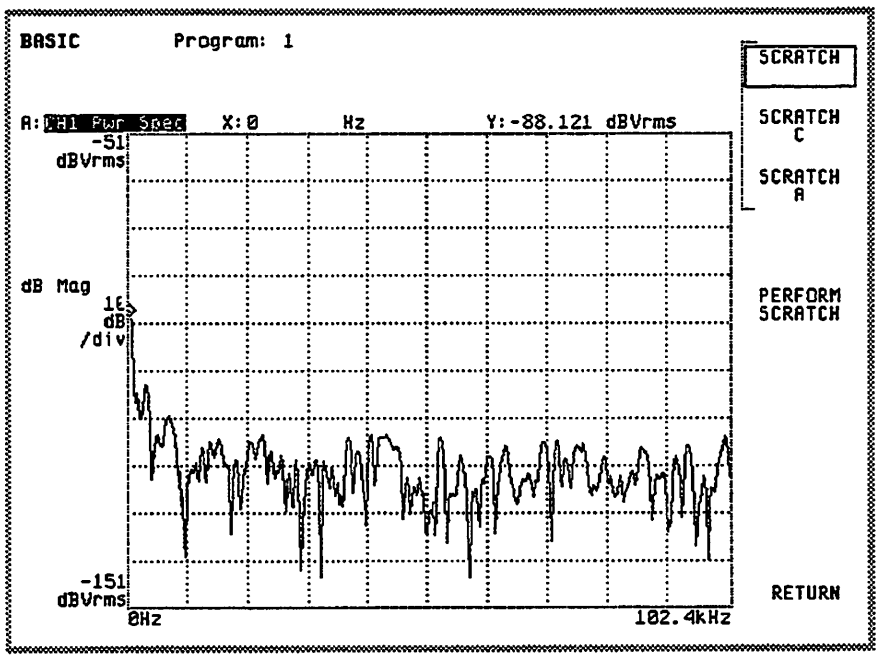

Figure 5-10. The [SCRATCH] Menu

The softkeys load into the keyboard function keys as follows:

[SCRATCH]	[*F1*]
[SCRATCH C]	[*F2*]
[SCRATCH A]	[*F3*]
[PERFORM SCRATCH]	[*F5*]
[RETURN]	[*F10*]

You must first select a combination of program and/or variables to clear by pressing [SCRATCH], [SCRATCH C], or [SCRATCH A]. The scratch operation is not executed, however, until you press the [PERFORM SCRATCH] softkey.

SCRATCH

This key selects the current active HP Instrument BASIC program and all variables not in COM.

SCRATCH C

This key selects all variables including those in COM, but does not clear the program.

SCRATCH A

This softkey selects the current active HP Instrument BASIC program and all variables, including those in COM.

The analyzer does not clear the memory until you press [PERFORM SCRATCH]. To cancel a SCRATCH operation, press [RETURN] at any time prior to pressing [PERFORM SCRATCH].

RENUMBER

Pressing [RENUMBER] displays a menu that allows you to change the line numbering for the entire active program. See Figure 5-11.

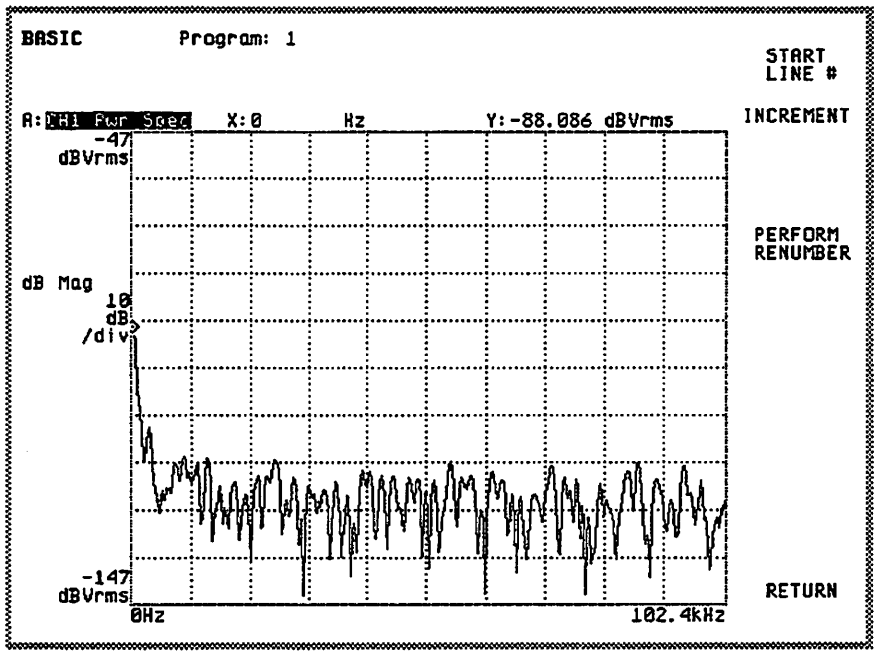

Figure 5-11. The [RENUMBER] Menu

The [RENUMBER] menu loads into the keyboard function keys as follows:

[START LINE #]	[*F1*]
[INCREMENT]	[*F2*]
[PERFORM RENUMBER]	[*F4*]
[CANCEL/RETURN]	[*F10*]

To select the number that is assigned to the first line in the program when renumbering lines in a program, press [START LINE #]. If you do not define the starting line number, the first line number defaults to 10.

Press [INCREMENT] to specify the increment between the renumbered line numbers. The default is 10. For example, if [START LINE #] is 10 and [INCREMENT] is 5, the line numbers will be 10 . . . 15 . . . 20 . . . 25 . . . and so on.

Once these parameters are defined, press [PERFORM RENUMBER] to execute the command. To cancel the renumbering operation, press [RETURN] at any time prior to pressing [RENUMBER PROGRAM].

SECURE

The [SECURE] menu allows you to "protect" program lines. "Protected" program lines cannot be listed to a printer or viewed in EDIT mode. See figure 5-12.

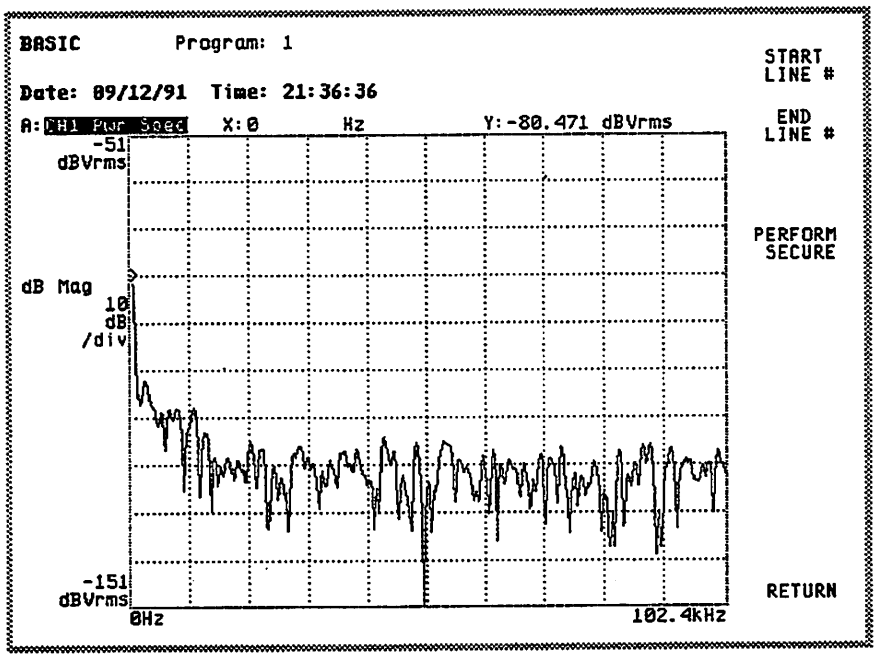

Figure 5-12. The [SECURE] Menu

The [SECURE] softkeys load into the keyboard function keys as follows:

[START LINE #]	[*F1*]
[END LINE #]	[*F2*]
[PERFORM SECURE]	[*F4*]
[RETURN]	[*F10*]

To secure a block of the active program:

1. Press [START LINE #].
2. Enter the beginning line number of the program block. (The value defaults to 1.)
3. Press [END LINE #].
4. Enter the ending line number of the program block. (The value defaults to 32766.)
5. Press [PERFORM SECURE].

Since [START LINE #] value defaults to 1 and the [END LINE #] value defaults to 32766, you can secure the entire program by pressing [PERFORM SECURE] without altering the start line and end line values.

Secured lines cannot be printed or viewed in the editor. They appear only as an asterisk following the line number (*). Secured lines can, however, be deleted from the program using the editor. You may leave this menu at any time by pressing [RETURN].

Caution

Secured program lines cannot be unsecured. Be sure to keep an unsecured version of the program for your own records.

Using [PRINT PROGRAM]

The [PRINT PROGRAM] softkey allows you to print the current contents of the active program buffer to a printer attached to the HP-IB interface. The printer address must be listed under [**Local/HP-IB**] [PERIPHERL ADDRESSES] [PRINTER ADDRESS]. To enable printing over the HP-IB, the analyzer must be set as the System Controller in the [**Local/HP-IB**] menu.

Note

If you press [PRINT PROGRAM] and do not have a printer connected or properly configured, HP Instrument BASIC continues attempting to print until you either press [**Local/HP-IB**] or [**Preset**].

Once the printing operation is completed, the PRINTER IS device is set to the display (CRT).

Using [DISPLAY SETUP]

Pressing the [**BASIC**] [DISPLAY SETUP] key allows you to allocate a partition of the analyzer's display to be used by your program. Alternately, HP Instrument BASIC can return any allocated partition of the display to the analyzer.

The HP 35665A display is divided into two small partition areas (UPPER, and LOWER) and one large area (FULL), which encompasses both the UPPER and LOWER partition areas.
See figure 5-13.

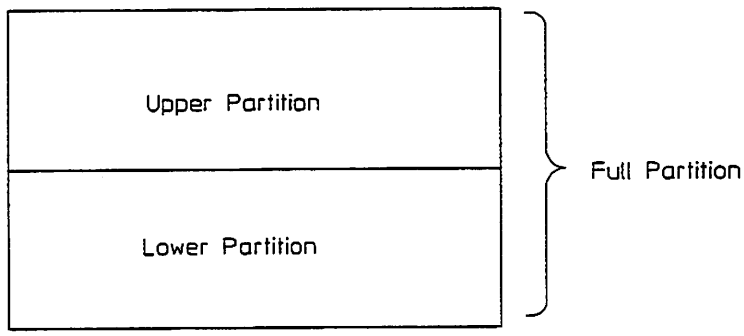

Figure 5-13. The Display Partitions

All screen output commands, such as PRINT and DRAW, require that you allocate a partition of the screen in order to view the results of the command. This can be performed in your program or interactively using the [DISPLAY SETUP] softkey. See figure 5-14.

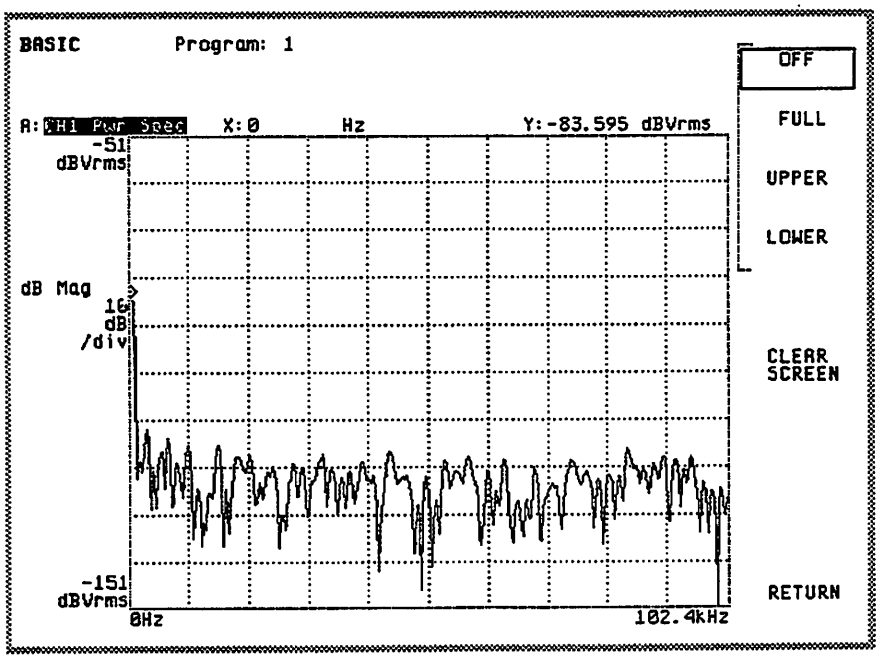

Figure 5-14. The [DISPLAY SETUP] Menu

The [DISPLAY SETUP] menu softkeys load into the following keyboard function keys:

[OFF]	[*f1*]
[FULL]	[*f2*]
[UPPER]	[*f3*]
[LOWER]	[*f4*]
[CLEAR SCREEN]	[*f6*]
[RETURN]	[*f10*]

You can allocate display partitions from within your program using the HP-IB command "DISP:PROG" and specifying the parameter UPPer, LOWer or FULL. For example the statement

```
OUTPUT 800;"DISP:PROG FULL"
```

allocates the single trace box of the display. This command corresponds to selecting [FULL] from the [DISPLAY SETUP] menu. Table 5-1 shows the relationship between the [DISPLAY SETUP] softkeys and the corresponding HP-IB commands required to program the same functions.

Table 5-1. The Display Partitions

MENU	ALLOCATES	HP-IB Command
OFF	NO DISPLAY	DISP:PROG OFF
FULL	SINGLE TRACE AREA	DISP:PROG FULL
UPPER	UPPER TRACE AREA	DISP:PROG UPP
LOWER	LOWER TRACE AREA	DISP:PROG LOW

Most display allocation should be handled by your program with the HP-IB commands. It is best to use these softkeys during program development.

The [CLEAR SCREEN] softkey clears all text and graphics from whichever partition is active.

For more information about controlling the display, refer to chapter 7, "Graphics and Display Techniques."

6

Debugging Programs

The process of creating programs usually involves correcting errors. You can minimize these errors by using keystroke recording for your measurement sequence program segments and by writing structured, well-designed programs.

Of course bugs can and do appear in even the best designed programs. HP Instrument BASIC contains some useful features to help you track them down.

Overview

The HP Instrument BASIC tools provided for program debugging are simple and, if used properly, can be very helpful. The [**BASIC**] menu contains the [DEBUG] softkey. See figure 6-1.

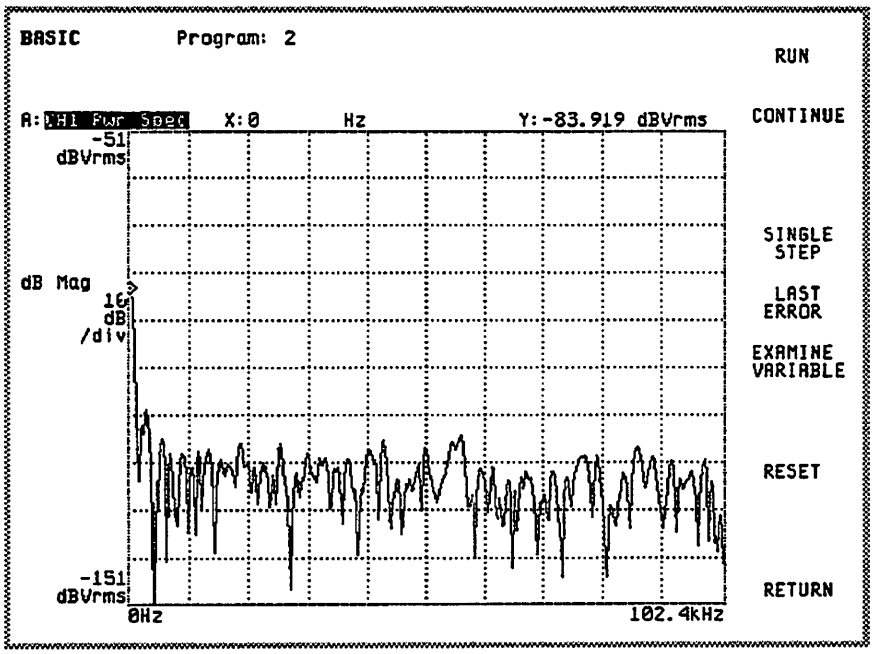

Figure 6-1. The [DEBUG] Menu

If using a keyboard, the [DEBUG] menu loads into the function keys as follows:

[RUN]	[*F1*]
[CONTINUE]	[*F2*]
[SINGLE STEP]	[*F4*]
[LAST ERROR]	[*F5*]
[EXAMINE VARIABLE]	[*F6*]
[RESET]	[*F8*]
[RETURN]	[*F10*]

The [DEBUG] menu provides several debugging facilities. For example, using the [DEBUG] menu you can:

- RUN or CONTINUE your program normally
- SINGLE STEP through your program one line at a time
- Display the last error encountered in your program
- Examine program variables

By examining the values assigned to variables at various places in the program, you can get a much better idea of what is really happening in your program.

Use the [SINGLE STEP] softkey to execute the program one line at a time. You can study the program's operation and examine variable values.

By inserting a PAUSE statement in your program you can pause the program at any line and then examine the values of variables at that point in the program. Press [CONTINUE] to resume operation to the next PAUSE statement or to the end of the program. Press [SINGLE STEP] to walk through program lines following the PAUSE statement.

By combining these different features you can examine the program's operation and solve your particular problem.

Using [EXAMINE VARIABLE]

Pressing [EXAMINE VARIABLE] displays an entry window that allows you to enter the name of the variable you want to examine. The default is the name of the last examined variable. It also brings up the editor (the alpha entry menu), so you can enter the variable name.

You must first perform a prerun operation to examine the value assigned to any variable in your program. A prerun is executed when you press either [RUN] or [SINGLE STEP]. After the prerun, press the [EXAMINE VARIABLE] key and enter the name of an existing variable in your program.

You can enter the variable as all uppercase letters. When you are finished entering a variable name, press [ENTER].

If you use [SINGLE STEP] and the program has not executed the line assigning that variable, the variable returns a value of zero.

Examining Strings

Enter string variables as you would any other variable. The entry window wraps to display a maximum of 10 lines of 42 characters each.

To select only a section of a string, use the HP Instrument BASIC substring syntax (see the "HP Instrument BASIC Programming Techniques" section in the *HP Instrument BASIC Users Handbook*). For example, to examine the 7 character substring starting at the second character of A$ enter:

```
A$[2;7]
```

Examining Arrays

You can examine an entire array or individual elements of the array. For example the entry:

```
I_array(1),I_array(2),I_array(3)
```

displays the elements 1 through 3 of the array *I_array*.

To select an entire array for examination enter the array variable name followed by an asterisk, (*); for example, I_array(*).

```
Example
```

I_array(20) is an integer array. The first and second elements are set to 100. After pressing [EXAMINE VARIABLES], enter "I_array(*)." The following is displayed:

```
I_array(*)  =  100  100  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
```

An individual array element (for example, I_array(17)), is specified the same as any single variable.

Setting Breakpoints

A common method of debugging a program is the use of breakpoints. A breakpoint causes the program to stop at a defined point so that you can examine the program state at that point. In HP Instrument BASIC this is accomplished by inserting PAUSE statements in the program code. When the program runs, you can use [EXAMINE VARIABLE] to check or change variable values. Press [CONTINUE] to continue the program until the next PAUSE, STOP or END statement is encountered.

You can enter PAUSE statements and otherwise alter the contents of the active program by using the [**BASIC**] [EDIT] softkey. See chapter 5, "Developing Programs," for a description of the HP Instrument BASIC editing capabilities in the HP 35665A.

Using [SINGLE STEP]

The [SINGLE STEP] softkey allows you to execute your program one line at a time. The line to be executed appears in the first line of the display. See figure 6-2.

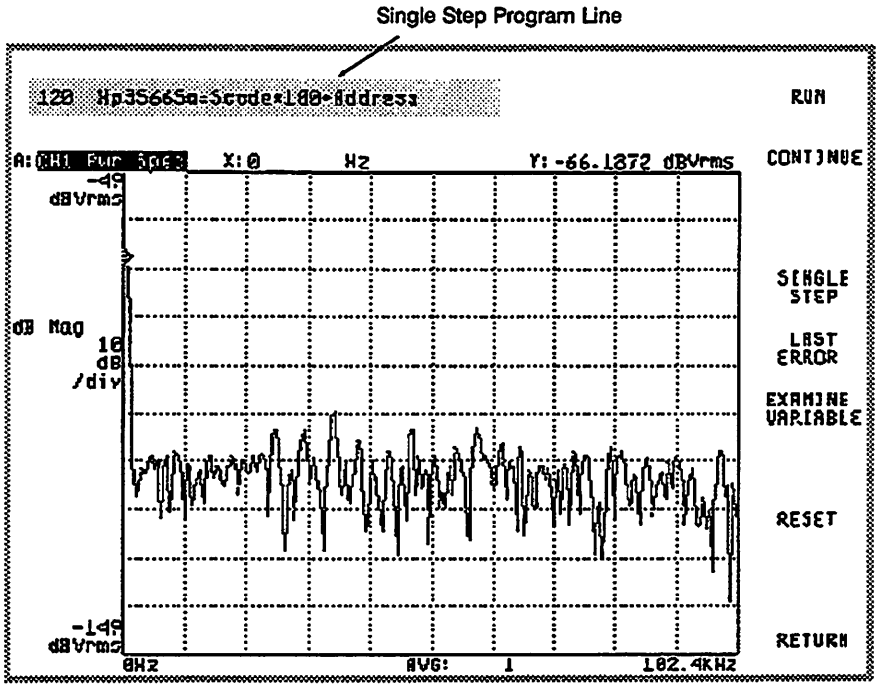

Figure 6-2. Single Step

You can use [SINGLE STEP] from the beginning of the program or from any point where it has been paused. To resume regular execution of a program after using [SINGLE STEP], press [CONTINUE].

[SINGLE STEP] can be very helpful when used in conjunction with the [EXAMINE VARIABLES] key and the PAUSE statement. By placing a PAUSE statement at a point of interest in your program, you can run the program until it pauses, then single step through the critical program lines, checking variables values or program operation. To resume program execution at any time, press [CONTINUE].

Using [RUN], [CONTINUE], and [LAST ERROR]

The [RUN] softkey operates the same as pressing [**BASIC**] [RUN]. It executes a prerun sequence and then begins executing the program at the first program line and continues until it reaches a PAUSE, STOP or END statement, or until the program is paused or stopped from the front panel.

The [CONTINUE] softkey allows you to resume regular program operation from a paused program or from a program in single step mode. This is identical to the [**BASIC**] [CONTINUE] softkey operation.

The [LAST ERROR] softkey displays the error number and message of the last error encountered by the program. This is the front panel equivalent to the HP BASIC command, ERRM$.

Using [RESET]

The [RESET] softkey allows you to bring the program environment back to its default state. This is especially useful when you are using single step mode and you want to restart the program. Pressing [RESET] sends an abort message to the HP-IB interface, resets the program counter to the first program line and closes all open files with one exception. Pressing [RESET] does *not* close a file if it is the device for a PRINTER IS statement.

7

Graphics and Display Techniques

HP Instrument BASIC programs have the ability to allocate portions of the instrument's display for text and graphics. This section provides a description of the various programming techniques used to do both.

Using the Partitions

There are several HP Instrument BASIC commands that require a display as an output device. These include commands such as PRINT, CLEAR SCREEN, MOVE, DRAW and GCLEAR. Since HP Instrument BASIC programs share all hardware resources with the instrument, the display must be shared for instrument and program use. All commands that output data to the screen write to a screen buffer and in order to view this output buffer, a portion of the display must be released from the instrument. You can do this manually when the program is not running by using the [**BASIC**] [DISPLAY SETUP] softkey menu. Performing equivalent actions from within a program that is running, requires sending an HP-IB message to the instrument; both to borrow a screen partition and again to give it back.

Allocating Partitions

The instrument's screen is divided into two trace boxes (upper and lower). The upper and lower trace boxes can be combined into one large trace box for single trace displays. Any of these three trace boxes, called display partitions, can be used by an HP Instrument BASIC program.

There are two other non-partition areas of the screen that can be accessed by HP Instrument BASIC programs. The area on the right of the screen is reserved for softkey labels and can be accessed using the ON KEY statement. Also, a line at the top of the screen can be accessed via the DISP and INPUT statements.

To request one of the partitions from the analyzer, send the instrument the corresponding HP-IB command. "DISP:PROG UPP" allocates the upper partition, "DISP:PROG LOW" allocates the lower partition, and "DISP:PROG FULL" allocates the full screen partition. See figure 7-1.

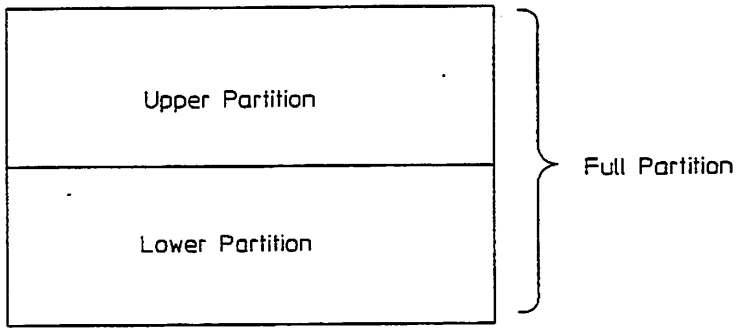

Figure 7-1. The Display Partitions

The following is an example of a program segment that prints a message to the upper trace box:

```
30  ASSIGN @Hp35665a TO 800
40  OUTPUT @Hp35665a;"DISP:PROG UPPER"
50  CLEAR SCREEN
60  PRINT "This is the upper partition"
```

To be sure that you are writing to a partition that has not been assigned, include a WAIT statement. Or, add an HP-IB query command followed by an ENTER statement to synchronize the program with the instrument. The previous example would look like this:

```
30  ASSIGN @Hp35665a TO 800
40  OUTPUT @Hp35665a;"DISP:PROG UPPER"
50  OUTPUT @Hp35665a;"DISP:PROG?"
60  ENTER  @Hp35665a;Part$
70  CLEAR SCREEN
80  PRINT "This is the upper partition"
```

The command DISP:PROG? (line 50 above) requests the instrument to send the current partition status. The ENTER statement on the next line reads that status and then continues.

De-Allocating Partitions

To return the display partition to the analyzer, use the "DISP:PROG OFF" command. This should be done before the termination of any program that has allocated a display partition. It may also be required within the program to allow someone to view instrument trace data. The following example demonstrates this command:

```
830  OUTPUT @Hp35665a;"DISP:PROG OFF"
```

Using Text

Most of the text capabilities of HP Instrument BASIC are covered in detail in the "HP Instrument BASIC Programming Techniques" section of the *HP Instrument BASIC Users Handbook.* The PRINT statement works the same in every partition. Information is printed starting at the top of the current partition and continues until the bottom of the partition is reached where the screen then scrolls up to allow additional lines to be printed. Causing the screen to scroll does *not* effect any graphics displayed on the screen, because text and graphics are written to different planes of the display.

All partitions have a width of 58 characters. The height varies according to partition. Both upper and lower partitions each contain 14 lines. The full partition contains 29 lines.

This information is useful if you are using the "PRINT TABXY" statement to position text. For example, the following program segment prints a message in the center of the full partition (assuming it has been allocated earlier in the program).

```
.
.
100  Maxlines=29
110  PRINT TABXY(25,Maxlines/2);"CENTER"
.
.
```

The following program segment demonstrates a technique to get text onto the screen quickly. Write your display message to a long string, using the OUTPUT statement, and then print the string to the screen. This speeds up screen display time considerably.

```
60   DIM Temp$[100],Big$[2000]
70   OUTPUT Temp$;"This is the first line of text"
80   Big$=Big$&Temp$
90   OUTPUT Temp$;"This is the second line of text"
100  Big$=Big$&Temp$
110  PRINTER IS CRT; WIDTH 2000
120  PRINT Big$
```

You can also print to the screen using the OUTPUT statement in conjunction with the display address (1). For example, the statement

```
OUTPUT CRT;" OUTPUT 1 WORKS WELL TOO"
```

writes the quoted text to the screen.

The display responds to several of the standard ASCII "control codes." These characters can be sent to the CRT by printing or sending the CHR$ function of the ASCII number. For example, CHR$(7) is the control code for the "bell" (CTRL-G) and has the effect of sounding the beeper. For more information on control codes recognized by the CRT, see the *HP Instrument BASIC Users Handbook.*

Note

It is sometimes a practice to embed these control codes in your PRINT statements when using external computers to develop programs. For example, the HP 9836 Series 200 Workstation allows you to enter control characters directly into the program using the "ANY CHAR" key. If you do this, do not attempt to use the HP Instrument BASIC editor on the program. This editor does not recognize embedded control codes and its actions may be unpredictable.

Using Graphics

The graphics commands of HP Instrument BASIC are easy to understand and use. Use the MOVE statement to move the "pen" to a specific pixel location (without drawing) and then draw a line from the current pen location to another pixel coordinate using the DRAW statement. The GCLEAR statement removes all graphics.

The PEN command provides an easy method of erasing lines drawn by the DRAW command. When PEN 1 is issued (the default state), all DRAW commands act normally. When PEN 0 is issued, all DRAW commands erase any pixels their path encounters. Where there are no lines in the path, none are drawn.

As an example of using the MOVE and DRAW commands, the following statement moves the logical pen to a point 100 pixels to the right of, and 150 pixels above, the lower left corner of the display:

```
100  MOVE 100,150
```

The following statement then draws a line to coordinates (200,10):

```
110  DRAW 200,10
```

Finally, these two statements erase the previously drawn line:

```
120  PEN 0
130  DRAW 100,150
```

As with text output, the program has to be assigned a partition before graphics can be viewed. Text and graphics output to a partition appear on separate planes. The pixel dimensions of each partition are shown in table 7-1.

Table 7-1. Maximum Pixel Coordinates

Partition	X Max	Y Max
FULL	474	345
UPPER	474	171
LOWER	474	171

In all partitions, pixel coordinate (0,0) is at the bottom-left corner and clipping occurs automatically if the X,Y coordinate exceeds the displayable range of the current partition.

The following program displays a "HELP" screen and demonstrates many of the techniques discussed so far.

```
10   DIM A$[58],String$[2000]
20   OUTPUT 800;"DISP:PROG FULL"
30   OUTPUT 800;"DISP:PROG?"
40   ENTER 800;P$
50   GCLEAR
60   CLEAR SCREEN
70   MOVE 0,0
80   DRAW 474,0
90   DRAW 474,345
100  DRAW 0,345
110  DRAW 0,0
120  MOVE 0,310
130  DRAW 474,310
140  PRINT TABXY(28,2);"HELP"
150  OUTPUT A$;" This program demonstrates how to print"
160  String$=String$&A$
170  OUTPUT A$;" several lines of text at one time.  This"
180  String$=String$&A$
190  OUTPUT A$;" method offers the fastest possible print speed."
200  String$=String$&A$
210  PRINTER IS CRT;WIDTH 2000      !prevent auto cr/lf
220  PRINT TABXY(1,4);String$
230  END
```

Running this program produces the screen display in figure 7-2.

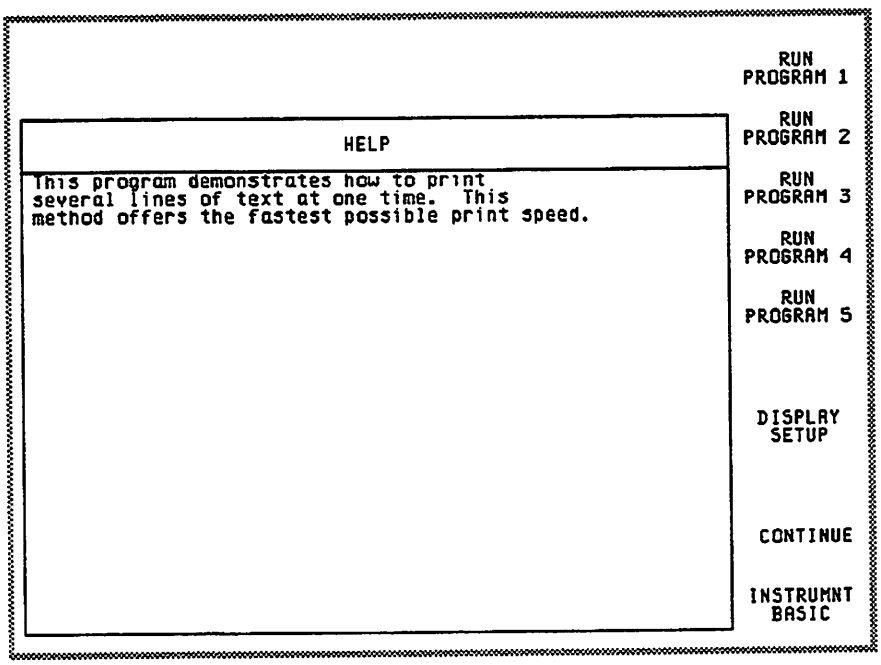

Figure 7-2. "HELP" Screen Output

8

Interfacing with the HP-IB

Introduction

This section describes the techniques necessary for programming the HP-IB interface. It also describes specific details of how this interface works and how to use it to control or interface with systems containing various HP-IB devices.

The HP-IB interface is Hewlett-Packard's implementation of the IEEE-488.1 Digital Interface for Programmable Instrumentation. The acronym HP-IB stands for "Hewlett-Packard Interface Bus," and is often referred to as the "bus."

The HP-IB Interface is both easy to use and allows great flexibility in communicating data and control information between the HP Instrument BASIC program and external devices.

HP Instrument BASIC is essentially an HP-IB instrument controller residing inside an instrument. It uses the host instrument's HP-IB interface for external communication and an internal HP-IB interface to communicate with the host instrument. This unique arrangement presents a few differences between HP Instrument BASIC's implementation of HP-IB control and the standard HP Series 200/300 BASIC Controller. A description of the interaction of HP Instrument BASIC with the host instrument and the external HP-IB interface is given in the section entitled "The HP Instrument BASIC HP-IB Model."

Communicating with Devices

This section describes programming techniques used to transfer data to and from HP-IB devices. General bus operation is described in a later section.

HP-IB Device Selectors

Since the HP-IB allows the interconnection of several devices, each device must have a means of being uniquely accessed. A device selector consists of two parts: the interface select code and the device's primary address. When a particular HP-IB device is to be accessed, it must be identified with both its interface select code and its bus address.

The interface select code is the first part of an HP-IB device selector. HP Instrument BASIC programs run inside a host instrument and communicate with it over the internal bus, which is addressed with select code 8. HP Instrument BASIC programs can also communicate with external devices via the host instrument's HP-IB interface. The external bus select code is 7.

The second part of an HP-IB device selector is the device's primary address. Each HP-IB device has a primary address which can be configured. The address can range from 0 to 30. For example, to specify the device on the interface at select code 7 (external bus) with a primary address of 22, use device selector = 722.

Each device's address must be unique. The procedure for setting the address of an HP-IB device is given in the installation manual for each device. Since the host instrument is the only device on the internal interface, its primary address on that interface is arbitrary and the instrument will respond to any primary address with a select code equal to 8XX (e.g., 800, 811, 822, etc.).

Secondary Addressing

Many devices have operating modes which are accessed through the extended addressing capabilities defined in the bus standard. Extended addressing provides for a second address parameter in addition to the primary address. Examples of statements that use extended addressing are as follows:

```
100  ASSIGN @Device TO 72205  !22=primary, 05=secondary
110  OUTPUT @Device;Message$

200  OUTPUT 72205;Message$

150  ASSIGN @Device TO 7220529       !Additional secondary
160                                  !address of 29
170  OUTPUT @Device;Message$

120  OUTPUT 7220529;Message$
```

The range of secondary addresses is 00-31. Up to six secondary addresses may be specified—a total of 15 digits including interface select code and primary address. Refer to the device's operating manual for programming information associated with the extended addressing capability.

Moving Data Through the HP-IB

Data is sent from the program through the HP-IB with the OUTPUT statement. Data is entered into the program with the ENTER statement.

The following examples illustrate the use of HP-IB device selectors with OUTPUT and ENTER statements.

Examples

```
100   Hpib=7
110   Device_addr=22
120   Device_selector=Hpib * 100 + Device_addr
130   !
140   OUTPUT Device_selector;"SYST:ERR?"
150   ENTER Device_selector;Reading

320   ASSIGN @Hpib_device TO 702
330   OUTPUT @Hpib_device;"Data message"
340   ENTER @Hpib_device;Number

440   OUTPUT 800;"SOUR:FREQ 1 KHZ"

380   ENTER 724;Readings(*)
```

General Structure of the HP-IB

Communications through the HP-IB are made according to a precisely defined set of rules. These rules ensure that only orderly communication takes place on the bus.

For conceptual purposes, the organization of the HP-IB can be compared to that of a committee. A committee uses "rules of order" to govern the manner in which they conduct their business. For example, a committee may conduct their meetings using "Robert's Rules of Order." For the HP-IB, the rules of order are the IEEE 488.1 standard.

The HP-IB System Controller is analogous to the chairman of a committee. Only one device can be designated System Controller and it is designated *before* running a program. The System Controller cannot be changed while under the control of a HP Instrument BASIC program. However, as it is possible for a chairman to designate an "acting chairman" for the committee, so can control be passed to another device on the HP-IB. This device is called the Active Controller. It can be any device capable of directing HP-IB activities, such as an instrument (using printing and plotting functions) or a desktop computer.

When the System Controller is first turned on or reset, it assumes the role of Active Controller. These responsibilities may be subsequently passed to another device while the System Controller tends to other business. This ability to pass control allows more than one computer to be connected to the HP-IB at the same time.

In an effective committee, only one person may speak at a time. It is the responsibility of the chairman to "recognize" which member is to speak. Usually, all committee members present are expected to listen at all times; however, this is not always the case on the HP-IB. One of the most powerful features of the bus is the ability to selectively send data to individual (or groups of) devices. This allows fast talkers to communicate with fast listeners without having to wait for slower listeners on the bus.

During a committee meeting, the current chairman is responsible for telling the committee which member is to be the "talker" and which members are to be the "listeners." Before these assignments are given, she gets the attention of the members. The talker and listeners are designated and then the talker presents the data. The designation process may be repeated after the talker has completed his message.

On the HP-IB, the Active Controller takes similar action when a talker and the listener(s) are designated. The attention signal line (ATN) is asserted while the talker and listener(s) are being addressed. ATN is then cleared, signaling that those devices not addressed to listen may ignore all subsequent data messages. Thus, the ATN line separates data from commands. Commands are accompanied with the ATN line being true, while data messages are sent with the ATN line being false.

On the HP-IB, devices are addressed to talk and addressed to listen in an orderly manner. The Active Controller first sends a single command that causes all devices to stop listening. The talker's address is then sent, followed by the address(es) of the listener(s). After all listeners have been addressed, the data is sent from the talker to the listener(s). Only device(s) addressed to listen accept any data that is sent through the bus (until the bus is reconfigured by subsequent addressing commands).

The transfer of data, called a data message, exchanges information between devices on the HP-IB. A committee conducts business by exchanging ideas and information between the speaker and those listening to his presentation. On the HP-IB, data is transferred from the active talker to the active listener(s) *at a rate determined by the slowest active listener on the bus.* This restriction on the transfer rate is necessary to ensure that no data is lost by any device addressed to listen. The handshake used to transfer each data byte ensures that all data output by the talker is received by all active listeners.

Examples of Bus Sequences

With HP Instrument BASIC, all data transfers through the HP-IB involve a talker and only one listener.

The following example illustrates the sequence of commands which are generated by the Active Controller to send data to an HP-IB device through the bus with a simple OUTPUT statement.

```
OUTPUT 701;"DATA"
```

1. The unlisten command is sent.
2. The talker's address, which is also a command, is sent. In this case, the address of the active controller.
3. The listener's address (01), which is also a command, is sent.
4. The data bytes "D", "A", "T", "A", CR, and LF are sent; all bytes are sent using the HP-IB's interlocking handshake to ensure that the listener has received each byte.

Similarly, all ENTER statements involve transferring data from a talker to only one listener. For instance, the following ENTER statement invokes the following sequence of commands and data-transfer operations.

```
ENTER 722;Voltage
```

1. The unlisten command is sent.
2. The talker's address (22), which is a command, is sent.
3. The listener's address, also a command, is sent. In this case, the listener's address is the active controller's address.
4. The data is sent by device 22 to the controller using the HP-IB handshake.

General Bus Management

The HP-IB standard provides several mechanisms that allow managing the bus and the devices on the bus. The following is a summary of the statements that invoke these control mechanisms.

ABORT is used to abruptly terminate all bus activity and reset all devices to their power-on states.

CLEAR is used to set all (or only selected) devices to a pre-defined, device-dependent state.

LOCAL is used to return all (or selected) devices to local (front panel) control.

LOCAL LOCKOUT is used to disable all devices' front panel controls.

REMOTE is used to put all (or selected) devices into their device-dependent, remote modes.

SPOLL is used to perform a serial poll of the specified device which must be capable of responding.

TRIGGER is used to send the trigger message to a device (or selected group of devices).

These statements (and functions) are described in the following sections. However, the actions that a device takes upon receiving each of the above statements are generally different for each device. *For external devices, refer to the particular device's documentation to determine how it responds.*

All of the bus management statements, with the exception of ABORT, require that the HP Instrument BASIC program be the Active Controller on the interface. A running program is always the Active Controller on the internal interface (select code 8). For the program to be the active controller on the external interface (select code 7), the host instrument must either be set as the System Controller or have control passed to it from the external controller. The program automatically assumes the controller status of the host instrument. For additional information refer to "The HP Instrument BASIC HP-IB Model" section later in this chapter.

REMOTE

External Devices

Most HP-IB devices can be controlled either from the front panel or from the bus. The device is in the "Local" state if the front panel controls are currently functional. If the device is controlled through the HP-IB, it is in the Remote state. Pressing the [**Local/HP-IB**] key returns the device to Local (front panel) control; unless the device is in the "Local Lockout" state, or the device is the host instrument.

The Remote message is automatically sent to all devices whenever the System Controller is powered on, is reset, or sends the Abort message. A device enters the Remote state automatically whenever it is addressed. The REMOTE statement also sends the Remote message. This causes all (or specified) devices on the bus to change from local control to remote control. The host instrument must be set to System Controller before an HP Instrument BASIC program can execute the REMOTE statement on select code 7 (the external bus).

Examples

```
REMOTE 7

ASSIGN @Device TO 700
REMOTE @Device

REMOTE 700
```

Host Instrument

The REMOTE statement has no effect on the host instrument because it is *always* in remote control whenever an HP Instrument BASIC program is running. Specifying the internal interface in a REMOTE statement has no effect and does not generate an error.

LOCAL LOCKOUT

External Devices

The Local Lockout message effectively locks out the "local" switch present on most HP-IB device front panels. This prevents anyone from interfering with the device's system operations by pressing buttons. Local lockout maintains system integrity. As long as Local Lockout is in effect, no bus device can be returned to local control from its front panel.

The Local Lockout message is sent by executing the LOCAL LOCKOUT statement. This message is sent to all devices on the external interface.

Examples

```
ASSIGN @Hpib TO 7
LOCAL LOCKOUT @Hpib

LOCAL LOCKOUT 7
```

The Local Lockout message is cleared when the Local message is sent by executing the LOCAL statement. Executing the ABORT statement does *not* cancel the Local Lockout message.

Host Instrument

The Local Lockout message is not supported for the host instrument because some front panel functionality is always necessary in order to pause or to abort the program. Specifying the internal interface in a LOCAL LOCKOUT statement does not generate an error and has no effect.

LOCAL

External Devices

It is good systems practice to return all devices to local control upon conclusion of remote-control operations. For example, an operator might need to troubleshoot or to work from the front panel to make special tests. Executing the LOCAL statement returns the specified devices to local (front panel) control.

If primary addressing is specified, the Go-to-Local message is sent only to the specified device. However, if the interface select code alone is specified (LOCAL 7), the Local message is sent to *all* devices on the external interface. Any previous Local Lockout message which is still in effect is automatically cleared.

Examples

```
ASSIGN @Hpib TO 7
LOCAL @Hpib

ASSIGN @Device TO 700
LOCAL @Device
```

Host Instrument

The LOCAL statement has no effect on the host instrument because it is always in remote control whenever an HP Instrument BASIC program is running. Specifying the internal interface in a LOCAL statement does not generate an error.

TRIGGER

External HP-IB Devices

The TRIGGER statement sends a Trigger message to a selected device or group of devices. The purpose of the Trigger message is to initiate some device-dependent action; for example, it can be used to trigger a digital voltmeter to perform its measurement cycle. The response of a device to a Trigger Message is strictly device-dependent. Neither the Trigger message nor the interface indicates what action is initiated by the device.

Examples

```
ASSIGN @Hpib TO 7
TRIGGER @Hpib

ASSIGN @Device TO 707
TRIGGER @Device
```

Specifying only the interface select code sends a Trigger message to all devices currently addressed to listen on the bus. Specifying a device's primary address in the statement triggers only the device addressed by the statement.

Host Instrument

The TRIGGER statement is fully compatible on the internal HP-IB interface. The HP 35665A must be set to trigger on the HP-IB for this statement to be effective.

```
OUTPUT @HP35665a;"TRIG:SOUR BUS"
TRIGGER @hp35665A
```

CLEAR

External HP-IB Devices

The CLEAR statement provides a means of "initializing" a device to its predefined, device-dependent state. When the CLEAR statement is executed, the Clear message is sent either to all devices or to the specified device, depending on the information contained within the device selector. If only the interface select code is specified, all devices on the specified HP-IB interface are cleared. If primary-address information is specified, the Clear message is sent only to the specified device. Only the Active Controller can send the Clear message.

Examples

```
ASSIGN @Hpib TO 7
CLEAR @Hpib

ASSIGN @Device TO 700
CLEAR @Device
```

Host Instrument

The CLEAR statement is fully compatible on the internal interface.

ABORT

External Devices

This statement terminates all activity on the external bus and returns all of the devices on the HP-IB to a reset (or power-on) condition. Whether this affects other modes of the device depends on the device itself. The HP Instrument BASIC program must be the Active Controller or the System Controller to perform this function. If it is the System Controller and has passed active control to another device, executing this statement returns active control to the program. *Only the interface select code is specified*; primary-addressing information (such as 724) is not included.

Examples

```
ASSIGN @Hpib TO 7
ABORT @Hpib

ABORT 7
```

Aborting the Internal Bus

ABORT is not supported for the internal bus, select code 8. Executing ABORT 8 does not generate an error.

HP-IB Service Requests

Most HP-IB devices, such as voltmeters, frequency counters, and spectrum analyzers, are capable of generating a "service request" when they require the Active Controller to take action. Service requests are generally made after the device has completed a task (such as making a measurement) or when an error condition exists (such as a printer being out of paper). The documentation, operating or programming manuals, for each device describes the device's capability to request service and the conditions in which the device requests service.

To request service, the device sends a Service Request message (SRQ) to the Active Controller. The mechanism by which the Active Controller detects these requests is the SRQ interrupt. Interrupts allow an efficient use of system resources, because the system executes a program until interrupted by an event's occurrence. If enabled, the external event initiates a program branch to a routine which "services" the event and executes remedial action.

Setting Up and Enabling SRQ Interrupts

In order for an HP-IB device to initiate a service routine in the Active Controller, two prerequisites must be met:

1. The SRQ interrupt event must have a defined service routine.
2. The SRQ interrupt must be enabled to initiate the branch to the service routine.

The following program segment shows an example of setting up and enabling an SRQ interrupt.

```
100  Hpib=7
110  ON INTR Hpib GOSUB Service_routine
120  !
130  Mask=2
140  ENABLE INTR Hpib;Mask
```

Since HP Instrument BASIC recognizes only SRQ interrupts, the value assigned to the mask is meaningless. However, a mask value may be present as a placeholder for compatibility with HP Series 200/300 BASIC programs.

When an SRQ interrupt is generated by any device on the bus, the program branches to the service routine when it exits the current line — either when the execution of the line is completed or when the line calls a user-defined function. The service routine must perform the following operations:

1. Determine which device is requesting service (parallel poll).
2. Determine what action is requested (serial poll).
3. Clear the SRQ line.
4. Perform the requested action.
5. Re-enable interrupts.
6. Return to the former task (if applicable).

Note

The ON INTR statement must always precede the ENABLE INTR statement when the two are used in the same program.

Servicing SRQ Interrupts

The SRQ is a level-sensitive interrupt; in other words, the interrupt may not be immediately detected when the SRQ line goes low. This implies that an interrupt may not be generated if the SRQ is present momentarily but does not remain long enough to be sensed by the controller. The level-sensitive nature of the SRQ line also has implications, which are described in the following example.

Example of a SRQ Interrupt

Assume only one device is currently on the bus. The following service routine first serially polls the device requesting service, thereby clearing the interrupt request. In this case, the controller did not have to determine which device was requesting service because only one device is on the bus. Also, the type of interrupt is not determined because only service request interrupts are enabled in HP Instrument BASIC. The service is then performed, and the SRQ event is re-enabled to generate subsequent interrupts.

```
500 Serv_rtn:  Ser_poll=SPOLL(@Device)
510    ENTER @Device;Value
520    PRINT Value
530    ENABLE INTR 7
540  RETURN
```

The IEEE standard specifies that when an interrupting device is serially polled, it is to stop interrupting until a new condition arises (or the same condition arises again). In order to "clear" the SRQ line, *it is necessary to perform a serial poll on the device*. This poll is an acknowledgement from the controller to the device that it has seen the request for service and is responding. The device then removes its request for service by releasing the SRQ line. When the SRQ line is released, the line goes high.

If the SRQ line had not been released, the controller would have immediately branched to the service routine after enabling interrupts on the external interface (line 530). This is another implication of the level-sensitive nature of the SRQ interrupt.

Once an interrupt is sensed and logged, the interface cannot generate another interrupt until after the initial interrupt is serviced. The controller disables all subsequent interrupts from an interface until a pending interrupt is serviced. For this reason, it is necessary to allow for subsequent branching.

Conducting a Serial Poll

A sequential poll of individual devices on the bus in known as a Serial Poll. The status of a specific device is returned in response to a Serial Poll. One entire byte is used. This is called the "Status Byte" message. Depending on the device, the Status Byte may indicate an overload condition, a request for service, or a printer which is out of paper. The particular response of each device depends on the device.

The SPOLL function performs a Serial Poll of the specified device. The HP Instrument BASIC program *must be* the Active Controller in order to execute it.

Examples

```
ASSIGN @Device TO 700
Status_byte=SPOLL(@Device)

Spoll_724=SPOLL(724)
```

The Serial Poll is meaningless for the external bus since it must poll the individual devices on the bus. Therefore, primary addressing must be used with the SPOLL function.

Passing and Regaining Control

Passing control can be accomplished in one of two ways: it can be handled by the system, or it can be handled by the program. To handle it programmatically, use the PASS CONTROL statement. HP Instrument BASIC or the analyzer can control the external bus (select code 7). The following statements first define the HP-IB's select code, specify the new Active Controller's primary address and then pass control to that controller.

```
100  Hp_ib=7
110  New_ac_addr=20
120  PASS CONTROL 100*Hp_ib+New_ac_addr
```

Once the new Active Controller has accepted active control, the controller which passed control assumes the role of a non-Active Controller on the HP-IB. *HP Instrument BASIC programs cannot act as a device when in a role of non-Active controller.*

Active control of the internal HP-IB bus (select code 8) cannot be passed. The statement "PASS CONTROL 800" passes control of the external bus to the instrument. This is required whenever the analyzer performs a plot operation to a peripheral on the bus or the analyzer accesses an external disk drive. These concepts are discussed next in "The HP Instrument BASIC HP-IB Model."

The HP Instrument BASIC HP-IB Model

The fact that HP Instrument BASIC resides in, and co-exists with an instrument creates a large set of possible interactions, both internally within the instrument as well as externally with other controllers and instruments. This section defines the principal players and rules of order when HP Instrument BASIC executes within the host instrument.

External and Internal Busses

There is physically only one HP-IB port and one HP-IB address for the HP 35665A. HP Instrument BASIC has access to two HP-IB ports: the "real" external port (select code 7) and a "virtual" internal port (select code 8), through which it communicates with the HP 35665A. See figure 8-1.

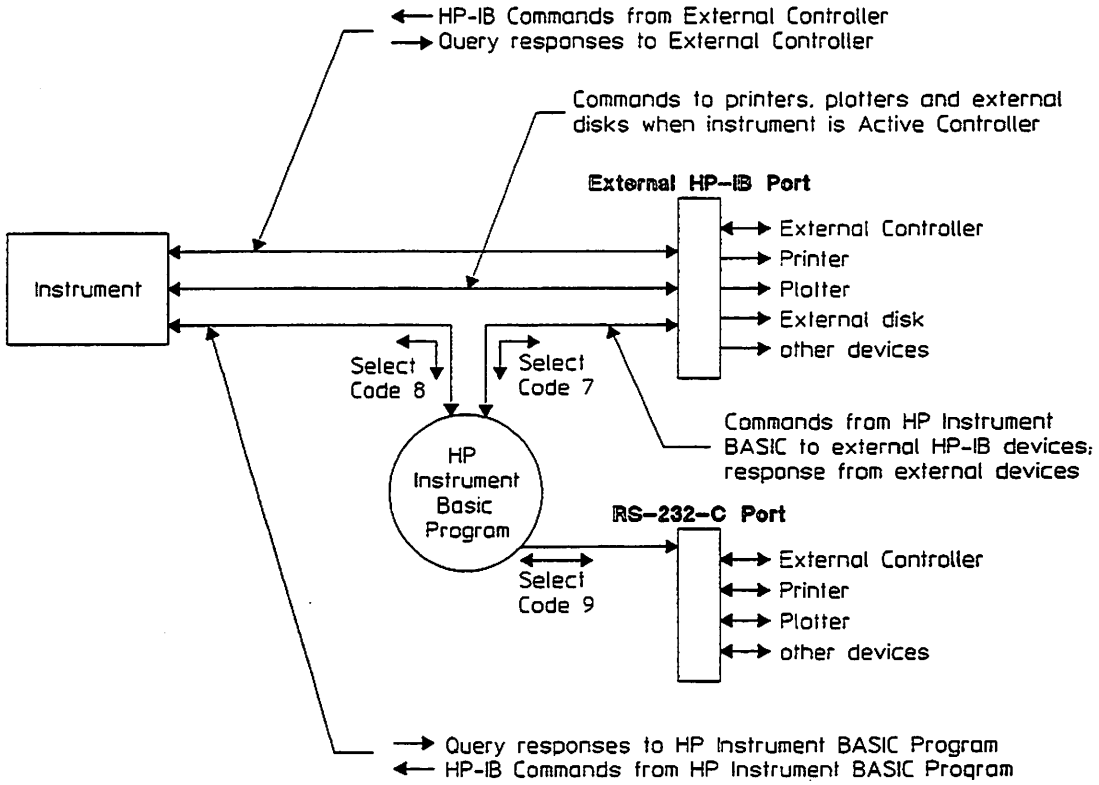

Figure 8-1. HP 35665A External and Internal Port

Port 1 on the rear panel is the RS-232-C serial port and is discussed in the next chapter.

Service Request Indicators

An external controller may perform a serial poll (SPOLL) at any time without affecting a running HP Instrument BASIC program. There are two Service Request Indicators (SRI) – one for the external port and one for the internal port. The internal SRI can only be cleared by an HP Instrument BASIC program performing an SPOLL on device 800. The external SRI can only be cleared by an SPOLL from an external controller and can only be set when there is no active HP Instrument BASIC program.

The two SRI's are set to their OR'd value when a program starts, and again when it finishes. This assures that any pending SRQ's can be serviced by the instrument's new controller.

The pausing or termination of a program causes the PROGRAM_RUNNING bit in the Operation Status register to go low. This can be used to generate an external SRQ. (For an example, see the example program, TWO_CTLR, in chapter 10.)

Status Registers

The HP 35665A's status registers contain information about various analyzer conditions. There are eight register sets. Their reporting structure is summarized in figure 8-2.

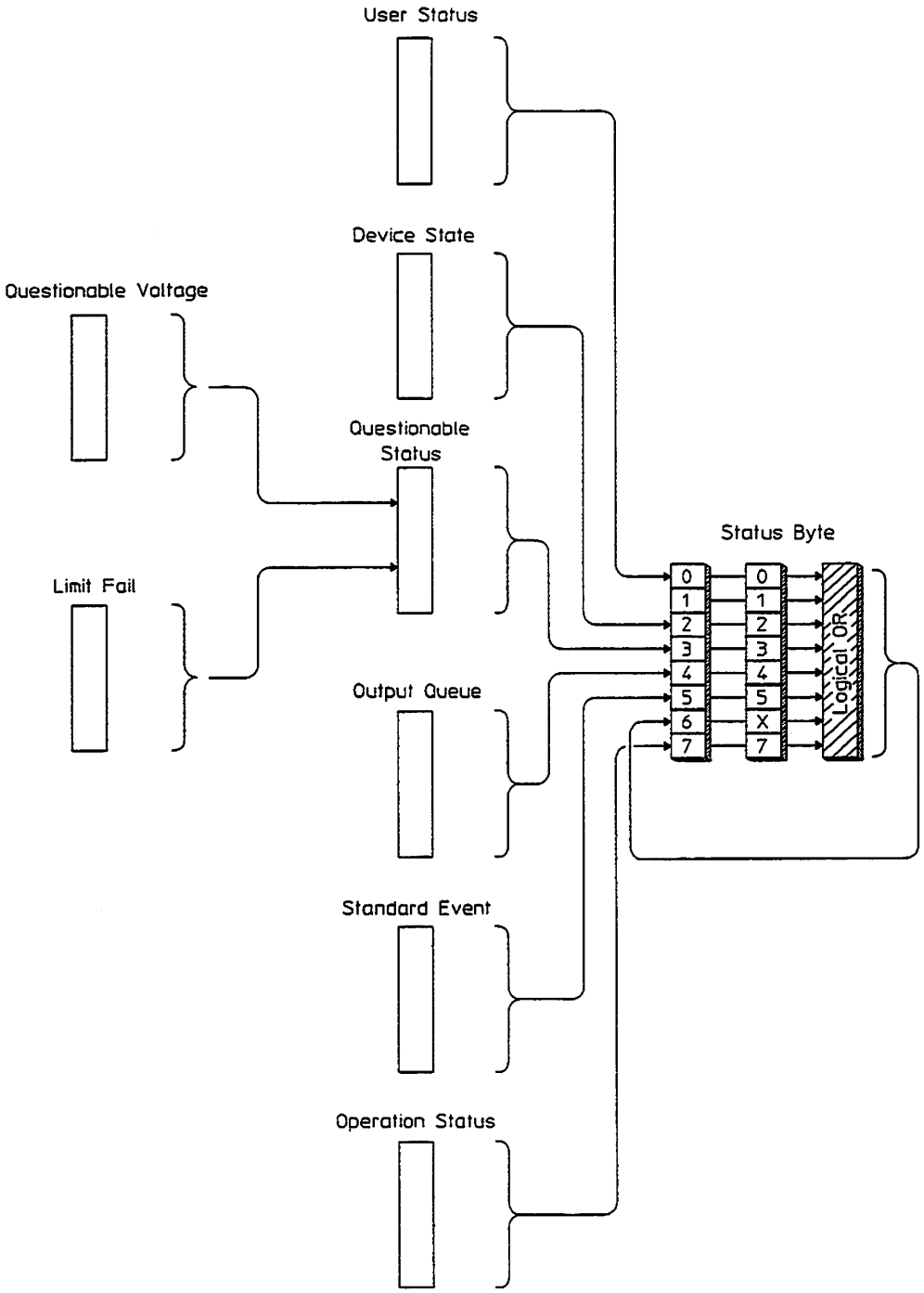

Figure 8-2. HP 35665A Status Registers

For more detailed information about the analyzer's register sets, refer to *HP-IB Programming with the HP 35665A*.

HP Instrument BASIC as the Active Controller

The HP Instrument BASIC program is *always* the Active Controller on the internal bus (select code 8). When a program starts running, the HP-IB controller status of the instrument is automatically passed to the program. See figure 8-3. For example, if the instrument is set as System Controller, a program running in the instrument automatically becomes the System Controller and the Active Controller on the external bus and the instrument relinquishes active control. When the program stops, the instrument regains active control.

Similarly, if an instrument set as Addressable Only is passed control from an external controller, any HP Instrument BASIC program running in the instrument becomes active controller on the external interface.

There are two cases when a program running in an instrument can become the Active Controller on the external interface:

- When the host instrument is set as System Controller and the program has *not* passed control.
- When the host instrument is set as Addressable Only and the instrument has been passed control from an external controller.

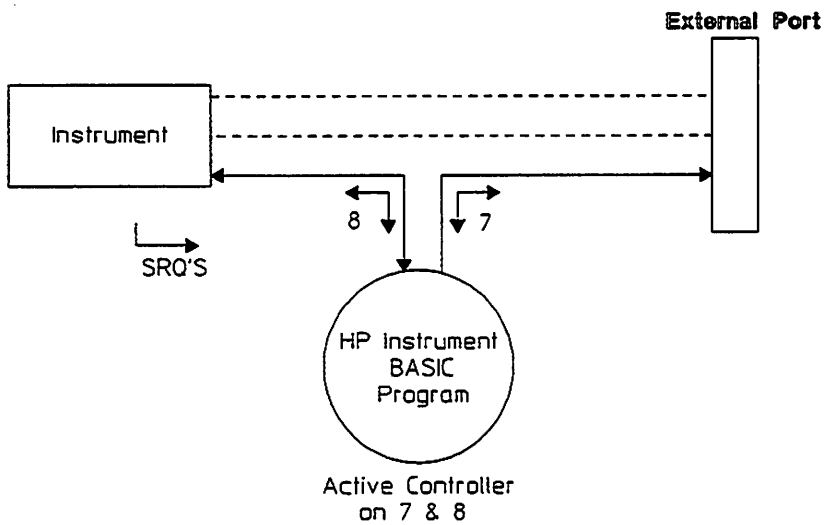

Figure 8-3. The Program as Active Controller on the External Interface

Passing Active Control to the Instrument

The only way that the HP 35665A can gain active control of the external interface while an HP Instrument BASIC program is running is if the program is currently the Active Controller on select code 7 and passes control to the instrument. Normally, the active controller on the external bus can pass control to any device on the interface by using the statement

```
PASS CONTROL 7xx
```

where "xx" represents the primary address of the device on the bus. However, since an HP Instrument BASIC program does not interface with the host instrument via select code 7, a different method must be used to pass control. To pass active control of the external interface from an HP Instrument BASIC program to the host instrument, use the statement:

```
PASS CONTROL 8xx
```

where "xx" represents any two digit number from 00 to 99. This allows the instrument to control external plotters, printers and disk drives. See figure 8-4. When the instrument is finished with its HP-IB control activity, it automatically passes control back to the program. If the instrument is waiting for control and the HP Instrument BASIC program terminates, control is implicitly passed back to the instrument. See figure 8-5.

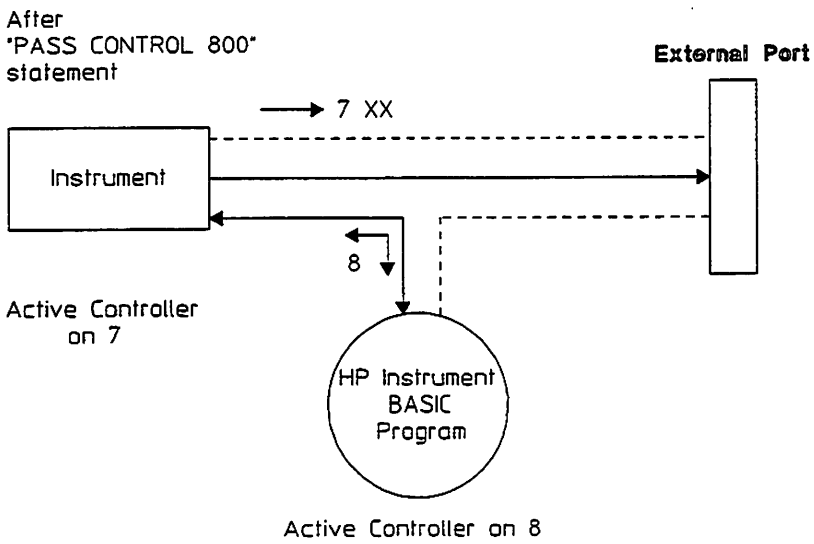

Figure 8-4. Passing Control of the External Interface to the HP 35665A

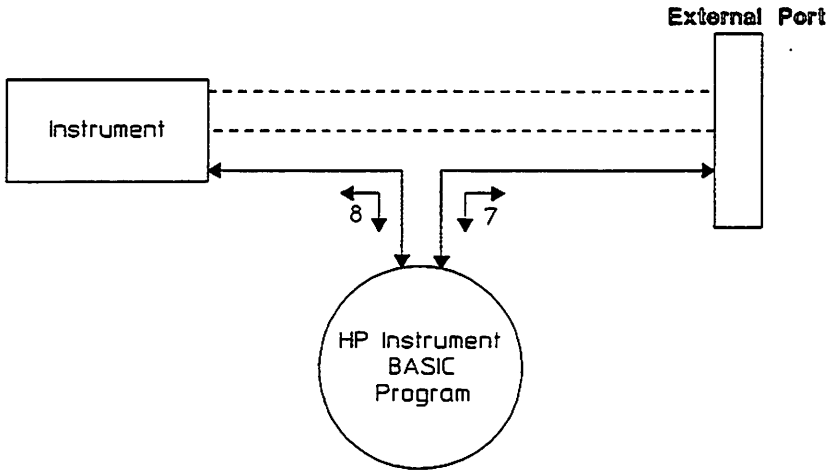

Figure 8-5. Control Passed Back to Program When Instrument Is Done

Note

Control of the internal bus is used to govern access to the external bus. When the instrument is given control of the internal bus, it actually gains access to the external HP-IB hardware.

HP Instrument BASIC as a Non-Active Controller

HP Instrument BASIC programs are always the Active Controller on the internal interface. There are two cases when an HP Instrument BASIC program does not have control of the external HP-IB interface:

- When the host instrument is set as Addressable Only and active control has *not* been passed from an external device.
- When the host instrument is set as System Controller and the program has passed control to either the host instrument or to another device on the external interface.

In both of these cases, the HP Instrument BASIC program cannot perform activities of any kind on the external bus. See figure 8-6.

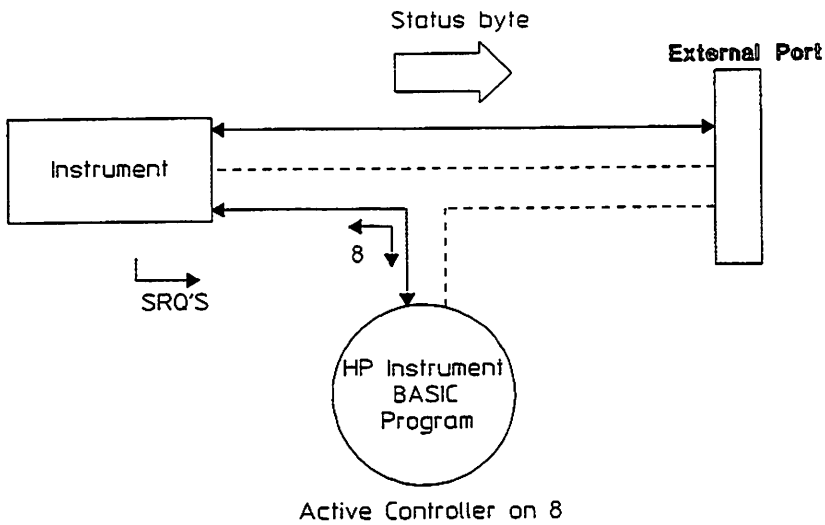

Figure 8-6. The Program as Non-Active Controller

Note

An HP Instrument BASIC program cannot act as a device on the external bus. To communicate with an external controller, the HP Instrument BASIC program must be Active Controller and the external controller must act as the device (see "Interfacing with an External Controller").

Interfacing with an External Controller

So far, we have limited our discussion to the ability to interface HP Instrument BASIC programs via HP-IB with a network of external devices. It is possible to include a computer in the network, and to interface an HP Instrument BASIC program with another program running in that computer.

External controller programs can interface with HP Instrument BASIC programs (hereafter referred to as "internal programs") over HP-IB in two ways:

The two programs can pass data back and forth using simple OUTPUT and ENTER statements. This requires coordination of both the internal and external programs and also requires that the internal program be the Active Controller during the interaction. To get an internal program and an external program to work together successfully, you should have a good understanding of the HP-IB model, as presented earlier in this chapter.

The external program can make use of the extensive set of HP 35665A HP-IB commands that interface with HP Instrument BASIC programs. These commands fall under the subsystems PROGram and MMEMory, and allow the external controller to remotely perform many of the HP Instrument BASIC front panel activities. This includes the ability to run, stop, pause, continue, get, save or delete an internal program. Commands in the SYSTem:COMMunicate:SERial subsystem configure the RS-232-C port. You can also remotely set a program's memory size and query or set the values of numeric and string variables.

Commands that allow you to transfer programs and program data to and from the instrument are included in the HP 35665A HP-IB command set. Programs can be transferred (uploaded and downloaded) between an external controller and the program buffer in the instrument. Data can be transferred between an external program and a non-running internal program by setting and querying internal program variables. These commands are described in detail in *HP-IB Programming with the HP 35665A*.

Transferring Data Between Programs

Using OUTPUT and ENTER Statements

All data sent from an external controller to the instrument's external port is received by the instrument — not by any program running in it. Therefore, an HP Instrument BASIC program that is not the Active Controller cannot enter or output data via the external interface bus. In order to pass data between an external controller and an internal program using OUTPUT and ENTER statements, *the internal program must be given active control* and the external controller must become the non-Active Controller. All HP Series 200/300 BASIC controllers have the ability to enter and output data via HP-IB while acting as a non-Active Controller.

Note

Moving data through the HP-IB and running a measurement in the host instrument at the same time can slow both operations significantly. It is recommended that you do not perform these operations concurrently.

One method of passing data between the two controllers is to first set the instrument as Addressable Only. Next, run an HP Series 200/300 BASIC program that starts the HP Instrument BASIC program and then passes control to it. Thereafter, the HP Instrument BASIC program can output data to, and enter data from, the external controller. The following two programs, found on the HP 35665A Example Programs disk, demonstrate how to transfer data between an internal program and an external controller program.

The first program, DTXFRB, runs on an HP Series 200/300 workstation. It assumes that a disk containing the corresponding HP Instrument BASIC program DTXFRA is in the HP 35665A disk drive. It remotely loads the HP Instrument BASIC program, starts it and then transfers active control to it. The HP Instrument BASIC program DTXFRA, with active control of the interface, queries the external program for the name of the drive to catalog, and then sends the cataloged string to the external program and passes back active control. After receiving the catalog data, the external program goes into a loop (line 460). This command continues to generate an error until control is passed back to the host computer, which again becomes the active controller.

```
10    !BASIC program:  DTXFRB -- Data transfer BASIC to BASIC
20    !-----------------------------------------------------------------
30    ! This program demonstrates how to transfer data from an HP Instrument
40    ! BASIC program.  This program, which runs on the computer, loads a
50    ! a program into the HP35665A, runs it, and then gives it control of
60    ! the bus. This program then acts as a device on the bus; sending and
70    ! receiving data. Before running this program, a disk with the program
80    ! 'DTXFRA' should be in the HP35665A's internal drive.
90    !-----------------------------------------------------------------
100   Scode=7                  !Select code for interface
110   Address=11               !Address for HP35665A
120   Hp35665a=Scode*100+Address
130   CLEAR Hp35665a
140   OUTPUT Hp35665a;"*CLS"   !Clear the EVENT registers
150   CLEAR SCREEN             !Clear the display
160   !
170   DIM Directory$(1:100)[85] !Array to hold catalog listing
180   !
190   INPUT "Put disk with program 'DTXFRA' into the HP35665A. Press
      ",A$
200   DISP "Loading program on HP35665A..."
210   OUTPUT Hp35665a;"MMEM:LOAD:PROG 'INT:DTXFRA'" !Load BASIC program from
      disk
220   OUTPUT Hp35665a;"*OPC?"
230   ENTER Hp35665a;Opc       !Wait here until program loaded
240   OUTPUT Hp35665a;"*ESR?" !Read the EVENT STATUS reg
250   ENTER Hp35665a;Esr
260   IF Esr0 THEN            !Have any errors occurred
270     BEEP
280     DISP "Error occurred while loading 'DTXFRA'...Cannot continue
        program."
290     STOP
300   END IF
310   !
320   OUTPUT Hp35665a;"*PCB 21" !Set pass control back address
330   !                                    to HP-IB address for controller
340   DISP "Running the program..."
350   OUTPUT Hp35665a;"PROG:STAT RUN"!Start the program
360   PASS CONTROL Hp35665a          !Give program control of bus
370   !
380   OUTPUT Scode;":INTERNAL"       !Wait until addressed to talk
390   DISP "Reading data..."
400   ENTER Scode;Directory$(*)      !Wait until addressed to listen
410   !
420   FOR I=1 TO 100                 !Print the catalog
430     IF LEN(Directory$(I))0 THEN PRINT Directory$(I)
440   NEXT I
450   !
460   ON ERROR GOTO 470 !Loop until control passed back
470   LOCAL Hp35665a
480   DISP ""
490   END
```

```
10   !BASIC program:  DTXFRA -- Data transfer BASIC to BASIC
15 !-----------------------------------------------------------------
20 ! This program demonstrates how to transfer data to and from an
30 ! external controller.  In this example a catalog listing is transferred
40 ! from the HP35665A to the external controller.  For more information
50 ! look at the program listing for 'DTXFRB'
60 !
70 ! This program is intended to be executed with HP Instrument BASIC.
80 !-----------------------------------------------------------------
90   DIM Directory$(1:100)[85] !Create string array for catalog
100 !
110   Host=721                  !Address for external controller
120 !
130   ON ERROR GOTO 140         !Loop until control is passed to the HP35665A
140   ENTER Host;Stor_dev$      !Address Host to talk, read device to catalog
150   OFF ERROR
160 !
170   DISP "Reading catalog..."
180   CAT Stor_dev$ TO Directory$(*)!Catalog into the string array
190 !
200   DISP "Transferring data..."
210   OUTPUT Host;Directory$(*)     !Address Host to listen, write array
220 !
230   PASS CONTROL Host             !Pass control back to host
240   DISP "DONE"
250   END
```

Setting and Querying Variables

Another means of transferring data between an internal and an external program involves the ability to set and query internal program variables from an external program. The "PROGram:NUMBer" and "PROGram:STRing" statements (and their query counterparts) are part of the HP 35665A HP-IB commands. *The internal program must not be running when these commands are executed.*

The command

```
PROG:NUMBer <"label">,<numeric value>
```

sets the value of a numeric variable in the program. The command

```
PROG:STRing <"label$">,<"string value">
```

sets the value of a string variable in the program. In both the PROG:NUMB and PROG:STR commands and queries, the *label* must be a string in quotes. In the PROG:STRing command, the *string variable data* must also be in quotes.

Numeric and string parameters can also be queried. The query

```
PROG:NUMBer? <"label">
```

returns the value of the specified INTEGER or REAL variable. If you precede this HP-IB command with the FORMat ASCII command (for example, OUTPUT 719;"FORM ASCii,5") the number returns as a readable ASCII number.

The query

```
PROG:STRing? <"label$">
```

returns the value of the specified string variable.

Arrays of REAL or INTEGER types may be sent or queried, but arrays of strings are not allowed. Array elements are separated by commas.

Examples

```
OUTPUT 711;"PROG:NUMB 'Test',99"

OUTPUT @Ibasic;"PROG:STRING 'A$','String Data'"

OUTPUT 711;"PROG:NUMB? 'Iarray(*)'"
```

The following program segment sends both numeric and a string variable queries and enters the resulting data:

```
10  ASSIGN @Prog TO 711
20  OUTPUT @Prog;"FORM ASCII,3"
30  OUTPUT @Prog;"PROG:NUMB? 'Test'"
40  ENTER @Prog; Testval
50  PRINT "The value of the variable Test =  ";Testval
60  OUTPUT @Prog;"PROG:STR? 'A$'"
70  ENTER @Prog; Str$
80  PRINT "A$ = ";Str$
90  END
```

Downloading and Uploading Programs

Programs can be transferred between an external controller and program memory using the HP-IB download command "PROG:DEFine" and its converse upload query "PROG:DEFine?." Programs that use these commands are executed in the external controller.

Downloading

Program data transferred (downloaded) from the external controller to the instrument is always transferred as an "arbitrary block." The arbitrary block may be a definite length block or an indefinite length block. The indefinite length block is by far the easiest to transfer. It is simply a block of data that begins with the characters "#0" preceding the first line and ends with a line-feed character accompanied by an EOI signal on the HP-IB interface.

When using the HP-IB command "PROG:DEF" to download program lines, the "#0" *should not be followed by a line-feed.* Each program line then requires a line number at its beginning and a line-feed at its end. To end the arbitrary block of program lines, a single line-feed must be output with the OUTPUT END parameter, which sends the EOI (End or Identify) signal on the HP-IB control lines.

The following program runs on an external HP Series 200/300 workstation. It demonstrates downloading a short program into the program buffer of the instrument. It is included in the HP 35665A Example Programs disk. The file must be an ASCII file.

```
10  ! ------------------------------------------------------------
20  ! HP BASIC Program: DOWNLOAD
30  ! This program downloads a file into an HP 35665A Instrument
40  ! BASIC program from an external controller.
50  ! The downloaded program must be an HP BASIC ASCII type file.
60  ! It will NOT work with DOS or HP-UX (untyped) files.
70  ! ------------------------------------------------------------
80  !
90   DIM Load_file$[20],Prog_line$[256],Command$[80],Name$[10]
100  DIM Diskname$[20],Answer$[2]
110  ASSIGN @Hp35665a TO 711
120 !
130 ! The file of program to download must be an ASCII file.
140 !
150  INPUT "ENTER NAME OF FILE TO DOWNLOAD ",Load_file$
160  ASSIGN @File TO Load_file$
170  INPUT "WHAT PROGRAM [1..5] DO YOU WANT TO DOWNLOAD IT TO?",Prognumber
180  OUTPUT @Hp35665a;"PROG:NAME PROG"&VAL$(Prognumber)
190  OUTPUT @Hp35665a;"PROG:DEL:ALL"
200  ON ERROR GOTO End_load
210  OUTPUT @Hp35665a;"PROG:DEF #0";
220  LOOP
230    ENTER @File;Prog_line$
240    PRINT Prog_line$
250    OUTPUT @Hp35665a;Prog_line$
260  END LOOP
270 !
280 End_load: !
290  OUTPUT @Hp35665a;CHR$(10) END
300  INPUT "SAVE PROGRAM TO INSTRUMENT'S DEFAULT DRIVE? [Y/N]",Answer$
310  IF UPC$(Answer$)="Y" THEN
320    INPUT "ENTER NAME FOR DISK FILE",Diskname$
330    OUTPUT @Hp35665a;"MMEM:STORE:PROGRAM '"&Diskname$&"'"
340  END IF
350  END
```

The OUTPUT statement on line 210 is terminated with a semicolon to suppress the line-feed that would otherwise occur.

As each line of the program is downloaded it is checked for syntax. If an error is found, the error message is displayed in a pop-up message window and the line is commented and checked for syntax again. If it still causes an error (for example the line may be too long) the line is discarded.

Any lines that currently exist in the memory buffer remain unless they are overwritten by downloaded program lines. This makes it easy to edit lines in an external controller and then download only the edited lines into an existing program. If you want to completely overwrite the current program in memory, you must delete the program first. This can be done remotely using the "PROG:DEL" command.

Uploading

The command "PROG:DEF?" is used to upload a program from the program buffer. The entire program is then returned as an definite length arbitrary block. A definite length block starts with the " #" character followed by a single digit defining the number of following digits to read as the block length. The following program demonstrates an uploading routine executed on an external controller. It is included in the HP 35665A Example Programs disk.

```
10 ! HP BASIC example program : UPLOAD
20 ! ------------------------------------------------------
30 ! This program runs on an HP BASIC workstation connected to
40 ! the HP 35665A with HP Instrument BASIC installed.  The 35665A
50 ! must have its address set to 711 and must be set up as
60 ! ADDRESSABLE ONLY on the HP-IB. This program uploads the
70 ! current program in the HP 35665A's memory to an ASCII file
80 ! on the workstation's current MSI disk.
90 ! ------------------------------------------------------
100   ASSIGN @Hp35665a TO 711
110   DIM Prog_line$[256]
120   INPUT "ENTER NAME OF FILE INTO WHICH TO UPLOAD PROGRAM ",Filename$
130   PRINT Filename$
140   CLEAR @Hp35665a
150   OUTPUT @Hp35665a;"PROG:DEF?"
160   ENTER @Hp35665a USING "#,A,D";Prog_line$,Ndigits
170   ENTER @Hp35665a USING "#,"&VAL$(Ndigits)&"D";Nbytes
180   PRINT Nbytes
190   Openfile(@File,Filename$,Nbytes)
200   ASSIGN @File TO Filename$
210   LOOP
220     ENTER @Hp35665a;Prog_line$
230   EXIT IF LEN(Prog_line$)=0
240     PRINT Prog_line$
250     OUTPUT @File;Prog_line$
260   END LOOP
270   ASSIGN @File TO *
290   END
300   SUB Openfile(@File,Filename$,Fisize)
310     ON ERROR GOTO Openerr
320     IF Fisize MOD 256 > 0 THEN Fisize=Fisize+256
330     CREATE ASCII Filename$,Fisize DIV 256
340 Openerr:  !
350     IF ERRN  <> 54 THEN
360       PRINT ERRM$
370     END IF
380   SUBEND
```

The subroutine, Openfile, (lines 300 through 330) creates a LIF file in which to save the uploaded program. The number of 256 byte records declared in the CREATE ASCII statement (line 330) is simply the file size (declared in the definite block header) divided by 256. Line 320 accommodates any remainder in this calculation by increasing the file size number by one record if any remainder exists.

Although this simple method works for many uploaded programs, there may still be a problem with the file size caused by the OUTPUT statement in line 250. This is because every ASCII line in a LIF file contains a two byte length header and possibly one additional pad byte to make the length an even number of bytes. These extra bytes are not included in the definite length block header information. You can account for this extra overhead by allocating an extra 10 to 15 percent of space when you create the ASCII file. For example, the Openfile subroutine could be rewritten as:

```
300  SUB Openfile(@File,Filename$,Fisize)
310      ON ERROR GOTO Openerr
315      Fisize = Fisize + (Fisize * .15)
320      IF Fisize MOD 256 > 0 THEN Fisize=Fisize+256
330      CREATE ASCII Filename$,Fisize DIV 256
```

9

Interfacing with the RS-232-C Serial Port

Introduction

This chapter describes the RS-232-C serial port. It explains how to configure the port. The connector, labeled Port 1, is located on the rear panel. Examples which illustrate the use of this interface in communicating with simple devices are presented throughout the chapter.

RS-232-C Serial Interface

The RS-232-C interface is used for simple asynchronous I/O applications such as driving printers, plotters, terminals and other peripherals or computers.

The serial interface changes 8-bit parallel data into bit-serial information and transmits the data. Data is received in this serial format and converted back to parallel data.

You should first determine that the peripheral device and the analyzer are compatible. The HP 35665A allows you to specify speed, handshaking (flow control), parity, character length and the number of stop bits.

Access to the RS-232-C serial port is not available from the front panel. The port is available only through the HP Instrument BASIC controller's select code 9. There are no select code switches. The select code is hardwired to select code 9. See figure 9-1.

Caution

If a real-time measurement is running, characters as input to the analyzer from the RS-232-C port may be lost. It is recommended that you pause a measurement during a RS-232-C input operation.

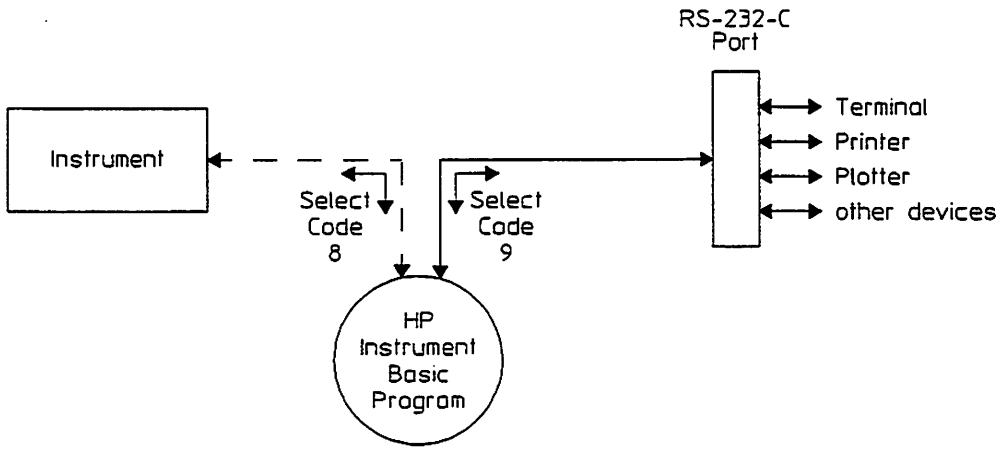

Figure 9-1. HP 35665A External RS-232-C Port

Asynchronous Data Communication

The terms Asynchronous Data Communication and Serial I/O refer to a technique of transferring information between two communicating devices by means of bit-serial data transmission. The data is sent, one bit at a time, and the characters are not synchronized with preceding or subsequent data characters. Each character is sent as a complete entity. Characters may be sent in close succession, or they may be sent sporadically as data becomes available. Start and stop bits are used to identify the beginning and end of each character, with the character data placed between them.

What HP Instrument BASIC Supports

The HP 35665A supports the minimum number of signals used in an RS-232-C system. The three key signals are connected to pins 2, 3, and 7 on the RS-232-C connector. Pin 2 carries "transmitted" data from the HP 35665A to the peripheral device. Pin 3 carries "received data" from the peripheral device to the HP 35665A. Pin 7 is a ground for the other two lines. See figure 9-2.

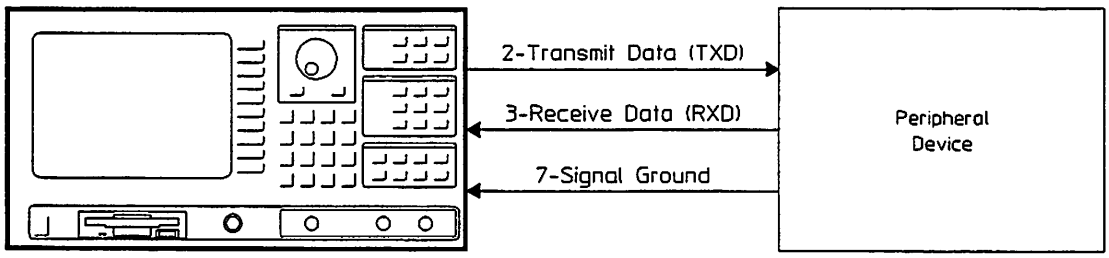

Figure 9-2. RS-232-C Lines

In addition, Pin 1 (DTR) and Pin 4 (RTS) are fixed high.

Asynchronous Transmission

The HP 35665A supports the asynchronous protocol. Asynchronous data communication is used in applications where high data integrity is not mandatory. Data is transmitted one character at a time. A start bit and one or more stop bits enclose the data character. Each character is individually synchronized or timed. The start and stop bits provide the timing.

Asynchronous Character Format

Each character consists of a start bit, 5 to 8 data bits, an optional parity bit and 1 or 2 stop bits. The HP 35665A does not support a 1.5 stop bit. The total time from the beginning of one start bit to the end of the last stop bit is called a character frame.

Figure 9-3 shows a structure of an asynchronous character and its relationship to previous and succeeding characters.

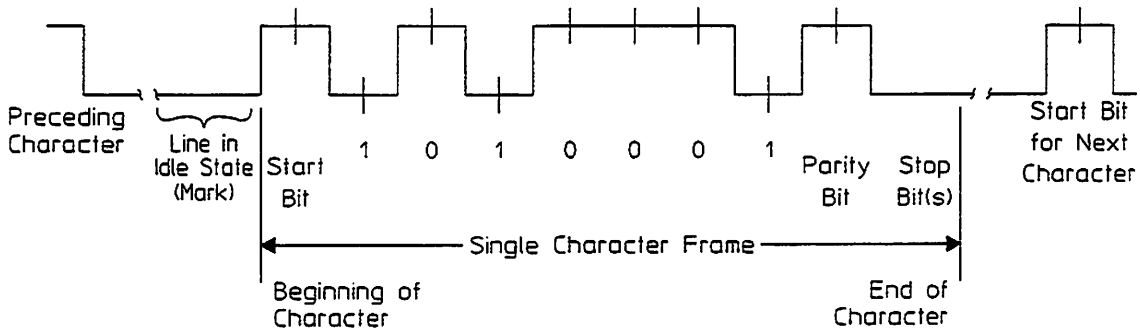

Figure 9-3. Asynchronous Format of a Single Character

Transferring Data

Two statements are used to transfer data between the HP 35665A and the peripheral device.

The OUTPUT statement sends data to the interface which, in turn, sends the information to the peripheral device.

The ENTER statement inputs data into the HP Instrument BASIC controller after the interface has received it from the peripheral device. The size of the buffer is 256 characters.

The OUTPUT and ENTER statements cause the HP 35665A to wait until the OUTPUT or ENTER operation is complete before continuing to the next line of the HP Instrument BASIC program. For OUTPUT statements, this means that the HP Instrument BASIC controller waits until the last character from the OUTPUT statement has been placed in the output buffer.

Establishing the Connection

Determine Link Operating Parameters

Before information can be successfully transferred between two devices, a communication link must be established. You must include the necessary protocol parameters to ensure compatibility between the communicating machines. To determine the proper parameters for your application, answer the following questions:

- What line speed (baud rate) is being used?
- How many bits (excluding start, stop and parity bits) are included in each character?
- What parity is being used: none, odd, or even?
- How many stop bits are required on each character you transmit?
- What line terminator should you use on each outgoing line?
- What line terminator, if any is sent at the end of each incoming line?
- What form of handshaking (pacing, flow control) is used?

All these parameters are configured under program control by use of HP-IB commands. The HP-IB commands can be sent within your HP Instrument BASIC program or they can be sent from an external controller. Refer to chapter 8, "Interfacing with the HP-IB," for additional information about configuring the RS-232-C interface from an external controller or refer to *HP-IB Programming with the HP 35665A*.

Parameters are set at the factory to accommodate common configurations. You can use the defaults, or you can override them with the appropriate HP-IB commands. The HP-IB commands are presented at the end of this chapter.

The configuration of the interface is stored in non-volatile RAM and is saved between sessions. The parameters do not change unless you explicitly change them with an HP-IB command. The following program queries and displays the configuration of the RS-232-C interface . It is a sub-routine of the example program RS232CNF which is listed in chapter 10 and is included in the HP 35665A Example Programs disk.

```
10 ! RS232CNF,  $Revision: 1.1 $
20 Parity_on$="NO"
30 OUTPUT 800;"SYST:COMM:SER:BAUD?"
40 ENTER 800;Baud
50 OUTPUT 800;"SYST:COMM:SER:BITS?"
60 ENTER 800;Bits
70 OUTPUT 800;"SYST:COMM:SER:PACE?"
80 ENTER 800;In_pace$
90 OUTPUT 800;"SYST:COMM:SER:PAR:CHEC?"
100 ENTER 800;Parity_on
110 IF Parity_on THEN Parity_on$="YES"
120 OUTPUT 800;"SYST:COMM:SER:PAR?"
130 ENTER 800;Parity$
140 OUTPUT 800;"SYST:COMM:SER:SBIT?"
150 ENTER 800;Sbits
160 OUTPUT 800;"SYST:COMM:SER:TRAN:PACE?"
170 ENTER 800;Out_pace$
180 !
190 CLEAR SCREEN
200 PRINT "RS-232 Interface Configuration"
210 PRINT
220 PRINT "baud rate:           ";Baud
230 PRINT "data bits:           ";Bits
240 PRINT "stop bits:           ";Sbits
250 PRINT "parity check enabled: ";Parity_on$
260 PRINT "parity type:          ";Parity$
270 PRINT "transmitter pacing:   ";Out_pace$
280 PRINT "receiver pacing:      ";In_pace$
290 END
```

Hardware Requirements

The RS-232-C connector, Port 1, is located on the analyzer's rear panel. The connector is set up for DTE (Data Terminal Equipment) applications. It has a 9-pin female, D-series connector. Table 9-1 lists the pin designators for the connector.

Table 9-1. Port 1 Pin Designators

Pin	Signal Description
1	Data Terminal Ready (fixed high)
2	Transmitted Data
3	Received Data
4	Request To Send (fixed high)
5	not used
6	not used
7	Signal Ground
8	not used
9	not used

Cables

You can use standard RS-232-C compatible cables as long as the signal lines are connected properly. See figure 9-4. The RS-232-C standard limits the cabling distance to 50 feet. Cables are available through your local Hewlett-Packard Sales and Service Office.

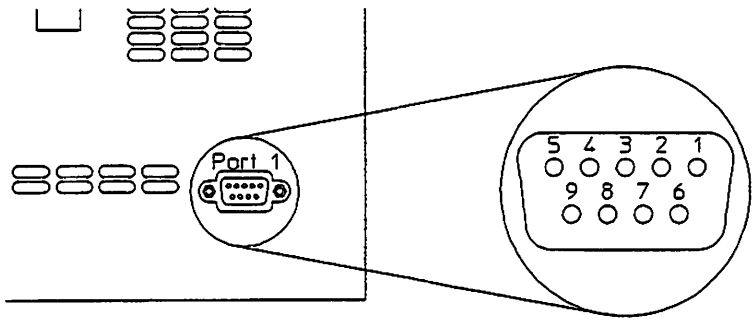

Figure 9-4. RS-232-C (Port 1) Pin Assignments.

Configuring the RS-232-C Port

Use the following HP-IB commands to configure the RS-232-C interface. The commands can be sent within you HP Instrument BASIC program or they can be sent from an external controller. For addition information about running an HP Instrument BASIC program from an external controller see "Interfacing with an External Controller" in chapter 8.

Refer to the command reference section at the end of this chapter for detailed information about the command syntax.

Character Length

The length of a character is set with the SYSTem:COMMunicate:SERial[:RECeive]:BITS command. Legal values are 5, 6, 7 or 8.

Example Command Statements

```
OUTPUT 711;"SYSTEM:COMMUNICATE:SERIAL:RECEIVE:BITS 7"

OUTPUT 711;"SYST:COMM:SER:BITS 5"
```

Example Query

```
OUTPUT 711;"SYST:COMM:SER:BITS?"
```

Number of Stop Bits

The number of stop bits is specified with the SYSTem:COMMunicate:SERial[:RECeive]:SBITs command. Legal values are 1 and 2.

Example Command Statements

```
OUTPUT 711;"SYSTEM:COMMUNICATE:SERIAL:RECEIVE:SBITS 2"

OUTPUT 711;"SYST:COMM:SER:SBIT 1"
```

Example Query

```
OUTPUT 711;"SYST:COMM:SER:SBIT?"
```

Speed (Baud Rate)

The rate at which data bits are transferred between the analyzer and the peripheral device is specified with the SYSTem:COMMunicate:SERial[:RECeive]:BAUD command. Legal values are 300, 1200, 2400, 4800, and 9600.

Example Command Statements

```
OUTPUT 711;"SYSTEM:COMMUNICATE:SERIAL:RECEIVE:BAUD 300"

OUTPUT 711;"SYST:COMM:SER:BAUD 2400"
```

Example Query

```
OUTPUT 711;"SYST:COMM:SER:BAUD?"
```

Parity

Parity is a technique used to detect transmission error by counting the number of bits in a character. Parity options include:

- NONE - Parity bit is not included.
- ODD - Parity is odd; there is an even number of "1"s in character bits.
- EVEN Parity is even; there is an odd number of "1"s in character bits.

Two separate commands are used. One command sets parity and another command enables the verification of parity.

Setting Parity Type

Parity is set with the SYSTem:COMMunicate:SERial[:RECeive]:PARity[:TYPE] command. Values can be set to NONE (a parity bit is not included), ODD or EVEN.

Example Command Statements

```
OUTPUT 711;"SYSTEM:COMMUNICATE:SERIAL:RECEIVE:PARITY:TYPE NONE"

OUTPUT 711;"SYST:COMM:SER:PAR ONE"
```

Example Query

```
OUTPUT 711;"SYST:COMM:SER:PAR?"
```

Parity Verification

To turn on parity verification send the SYSTem:COMMunicate:SERial[:RECeive]:PARity:CHECk command. Legal values are 0 or OFF and 1 or ON.

Example Command Statements

```
OUTPUT 711;"SYSTEM:COMMUNICATE:SERIAL:RECEIVE:PARITY:CHECK 0"

OUTPUT 711;"SYST:COMM:SER:PAR:CHEC 1"
```

Example Query

```
OUTPUT 711;"SYST:COMM:SER:PAR:CHEC?"
```

Handshaking

Controlling the flow of data is important during the data transfer operation to avoid losing data. The input buffer of a printer may become full as the speed of character transmission exceeds the printer's ability to print. Data might be lost if the printer runs out of paper during the transmission. To ensure that the transmitter does not send characters faster than the receiver can process them, a pacing mechanism is used to control the flow of data. This pacing mechanism is called a "handshake." Handshaking is performed automatically as part of the OUTPUT or ENTER operation.

Controlling the Flow of Data

The HP 35665A supports one type of handshake protocol to control the flow of data; the XON/XOFF protocol. Check your peripheral device's documentation to verify it supports the protocol you want to use.

In the XON/XOFF protocol, the *receiving device* controls the flow of data. The receiving device sends XOFF (ASCII DC3, decimal value 19) to halt the flow of data. When the receiving device is able to receive additional characters, it sends XON (ASCII DC1, decimal value 17) to signal the transmitter to resume transmission.

NONE specifies that a protocol is not used during transmission.

The SYSTem:COMMunicate:SERial[:RECeive]:PACE command sets the type of handshake when the analyzer is receiving data from a peripheral device (for example, from another computer).

Example Command Statements

```
OUTPUT 711;"SYSTEM:COMMUNICATE:SERIAL:RECEIVE:PACE NONE"

OUTPUT 711;"syst:comm:ser:pace XON"
```

Example Query

```
SYSTEM:COMMUNICATE:SERIAL:PACE?
```

The SYSTem:COMMunicate:SERIAL:TRANsmit:PACE command sets the type of handshake used when the analyzer is sending data. It defines the protocol the analyzer expects from the device.

Example Command Statements

```
OUTPUT 711;"SYSTEM:COMMUNICATE:SERIAL:TRANSMIT:PACE NONE"

OUTPUT 711;"syst:comm::ser:tran:pace XON"
```

Example Query

```
SYST:COMM:SER:TRAN:PACE?
```

Transferring Data

The serial RS-232-C interface is designed for relatively simple serial I/O operations.

Entering and Outputting Data

When the RS-232-C interface is properly configured by sending the appropriate HP-IB commands, you are ready to begin data transfers. Outbound data messages are created by OUTPUT statements. Inbound data messages are created by the interface as messages are received from the peripheral device. They are transferred to HP Instrument BASIC by ENTER statements.

Any valid OUTPUT or ENTER statement and variable(s) list may be used, but you must be sure that the data format is compatible with the peripheral device. For example, non-ASCII data sent to an ASCII line printer may result in unexpected behavior.

Outbound Data Messages

Outbound data messages are created when an OUTPUT statement is executed. Data is transmitted directly from the outbound buffer.

The output operation completes after the last byte of the character has been sent. The HP Instrument BASIC controller waits until the last byte in the statement variable list is transmitted by the interface.

An end-of-line (EOL) sequence is automatically sent at the end of data unless a semicolon or END appears at the end of a OUTPUT statement. The semicolon delimiter overrides EOL sequence output.

The Output Statement

Data items are transferred one byte at a time, beginning with the left-most item in the source list and continuing until all of the source items have been sent. Items in the list must be separated by either a comma or a semicolon. Depending on the use of item separators in the source lists, data items in the output may or may not be separated by item terminators. The end-of-line (EOL) sequence is described above.

Example Program Statements

```
OUTPUT 9;"Hello World"

PRINTER IS 9
PRINT "Hello World"
```

Inbound Data Messages

Inbound data messages are created by the RS-232-C interface as information is received from the peripheral device. ENTER statements are terminated when a new-line character (ASCII LF, decimal value 10) is encountered. A carriage-return (ASCII CR, decimal value 13) does *not* terminate a data string.

Caution 

Running a real-time measurement at the same time while receiving data at a high baud rate, may result in an overrun condition.

The Enter Statement

Items in ENTER statements can be separated by either a comma or a semicolon. Trailing punctuation is *not* allowed. A data item is terminated with a new-line character (ASCII 10).

Example Program Statement

```
ENTER 9;A$
```

Clearing the Input Buffer

The input buffer holds 256 characters. Sometimes the input buffer contains characters from previous transmissions. The statement, CLEAR 9, clears the input buffer.

Error Detection

Two types of incoming data errors can be detected by the HP 35665A.

- Parity Errors are signaled when the parity bit does not match–even or odd–the number of "ones" (including the parity bit) as defined by the interface configuration. When parity is disabled, a parity check is not made.

- Framing errors are signaled when start and stop bits are not properly received during the expected time frame. They can be caused by a missing start bit, noise errors near the end of the character, or by improperly specifying character length at the HP 35665A or the peripheral device.

Two additional types of errors can be detected by analyzer.

- Overrun errors result when the HP 35665A does not consume characters as fast as they arrive.

  Overrun conditions can occur during real-time measurements. If a real-time measurement is running at the same time the RS-232-C port is transferring data at a high baud rate, you may encounter an overrun error. This happens as a result of sending a second character to the analyzer before it can read the first character.

  Another type of overrun occurs when characters are sent to the analyzer but an HP Instrument BASIC program is *not* reading the characters. There is a 256 byte input buffer which is filled as characters are received. If this input buffer is filled, an overrun error is generated.

- Received BREAKs are detected as a special type of framing error. They generate the same type of HP Instrument BASIC error as framing errors.

The Device State Register

The HP 35665A's status registers contain information about various analyzer conditions including the detection of errors. The Device State Register Set contains this information. See figure 9-5.

Bit 9 is set to 1 when a framing error, overrun error, parity error or break is detected.

Bits 6, 7 and 8 are used to monitor the condition of data transfers.

- Bit 6 is set to 1 when a character is in the input buffer.
- Bit 7 is set to 1 when input is held off due to handshake protocol conditions.
- Bit 8 is set to 1 when output is held off due to handshake protocol conditions.

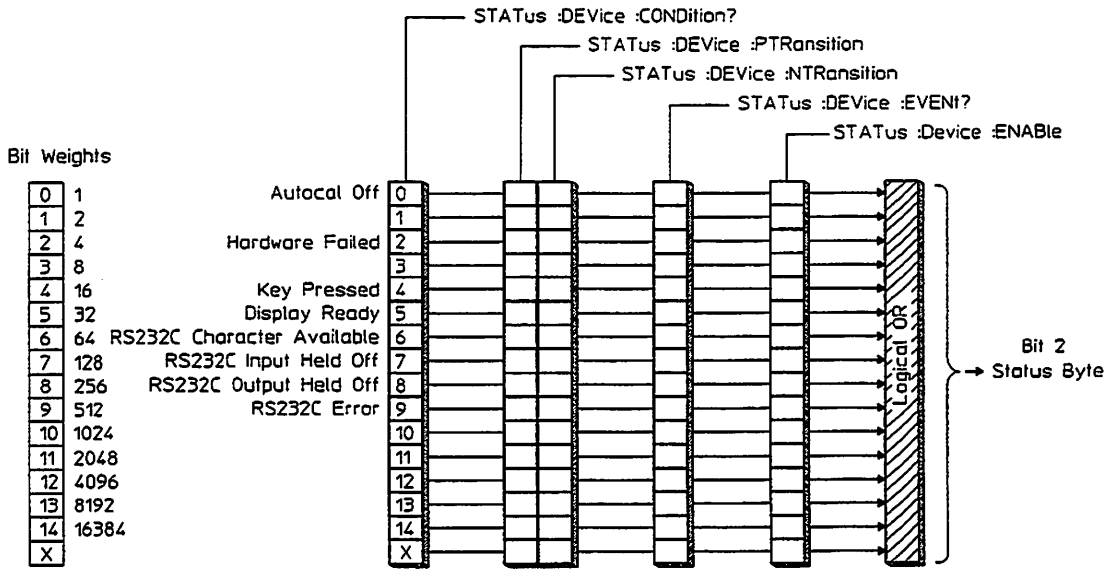

Figure 9-5. The Device State Register Set

Event-Initiated Branching

HP Instrument BASIC allows event-initiated branching, which uses interrupts to redirect program flow. Each time the program finishes a line, the analyzer executes an "event-checking" routine. This "event-checking" routine causes the program to branch to a specified statement if an enabled event has occurred.

The HP 35665A supports one type of event-initiated branching. ON TIMEOUT generates an interrupt when an interface or device takes longer than a specified time to respond to a data-transfer handshake.

All "ON-event" statements have a corresponding "OFF-event" statement.

It is possible to temporarily disable an event-initiated branch. A special section of code can be "protected," that is, not be interrupted, with a DISABLE statement.

See "Program Structure and Flow" in "HP Instrument BASIC Programming Techniques" (*HP Instrument BASIC Users Handbook*) for additional information about event-initiated branching.

Timeouts

ON TIMEOUT defines and enables an event-initiated branch to be taken when an I/O timeout occurs on the specified interface. The timeout is specified in seconds. For the RS-232-C interface the maximum timeout is 25.5 seconds.

Timeouts apply to ENTER and OUTPUT statements, and operations involving the PRINTER IS, PRINTALL IS and PLOTTER IS external devices.

OFF TIMEOUT deactivates ON TIMEOUT. DISABLE does *not* affect ON TIMEOUT.

Examples

```
ON TIMEOUT 9, 1.5 GOTO 1200

ON TIMEOUT 9, .5 GOSUB Service_routine
```

Command Reference

The command reference describes the HP-IB commands which configure the RS-232-C port. Each command description has the following:

- A brief description of the command.
- A syntax description.
- Example statements.
- A returned format description.
- An attribute summary. This field defines the command's preset state and specifies compliance with SCPI. A "confirmed" command complies with SCPI 1991. An "approved" command complies with SCPI 1992. An "instrument-specific" command does not conform to the SCPI standard.
- A detailed description.

SYSTem:COMMunicate:SERial[:RECeive]:BAUD command/query

Specifies the data-transfer rate between the analyzer and peripheral devices over the RS-232-C port.

Command Syntax:

```
SYSTem:COMMunicate:SERial[:RECeive]:BAUD
<number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  300:9600
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":syst:communicate:ser:baud 300"
OUTPUT 711;"System:Comm:Serial:Rec:Baud 9600"
```

Query Syntax:

```
SYSTem:COMMunicate:SERial[:RECeive]:BAUD?
```

Return Format:

```
Integer
```

Attribute Summary:

```
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This command specifies the baud rate between the analyzer and any peripheral devices.

Legal values are 300, 1200, 2400, 4800, and 9600.

SYSTem:COMMunicate:SERial[:RECeive]:BITS command/query

Specifies the number of bits in a character for the RS-232-C interface.

Command Syntax:

```
SYSTem:COMMunicate:SERial[:RECeive]:BITS
<number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  5:8
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"SYST:COMM:SER:REC:BITS 7"
OUTPUT 711;"SYST:COMM:SER:BITS 8"
```

Query Syntax:

```
SYSTem:COMMunicate:SERial[:RECeive]:BITS?
```

Return Format:

```
Integer
```

Attribute Summary:

```
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Legal values are 5, 6, 7 or 8.

SYSTem:COMMunicate:SERial[:RECeive]:PACE command/query

Sets the RS-232-C receiver handshake pacing type.

Command Syntax: `SYSTem:COMMunicate:SERial[:RECeive]:PACE {NONE|XON}`

Example Statements:
```
OUTPUT 711;"SYST:COMM:SER:REC:PACE XON"
OUTPUT 711;"system:comm:serial:pace none"
```

Query Syntax: `SYSTem:COMMunicate:SERial[:RECeive]:PACE?`

Return Format:
```
XON
OFF
```

Attribute Summary:
```
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This command specifies the handshake type when the analyzer is receiving data.

XON specifies the XON/XOFF protocol. The analyzer sends XOFF (ASCII DC3, decimal code 19) to the peripheral to stop transmitting data. When it is able to receive additional data, the analyzer sends XON (ASCII DC1, decimal code 17) to the peripheral which then resumes transmitting data.

NONE specifies the absence of a protocol for pacing.

Check the documentation for your peripheral device to verify it supports the specified protocol.

SYSTem:COMMunicate:SERial[:RECeive]:PARity:CHECk command/query

Enables RS-232-C parity verification capability.

Command Syntax:
```
SYSTem:COMMunicate:SERial[:RECeive]:PARity:CHECk
{OFF|0|ON|1}
```

Example Statements:
```
OUTPUT 711;"system:communicate:serial:parity:check ON"
OUTPUT 711;"SYST:COMM:SER:REC:PAR:CHEC  1"
```

Query Syntax:
```
SYSTem:COMMunicate:SERial[:RECeive]:PARity:CHECk?
```

Return Format:
```
Integer
```

Attribute Summary:
```
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This command turns on parity verification for the RS-232-C interface. To set the type of parity, send the SYSTem:COMMunicate:SERial[:RECeive]:PARity[:TYPE] command.

SYSTem:COMMunicate:SERial[:RECeive]:PARity[:TYPE] command/query

Sets the parity generated for characters transmitted over the RS-232-C interface.

Command Syntax:
```
SYSTem:COMMunicate:SERial[:RECeive]:PARity[:TYPE]
{NONE|EVEN|ODD}
```

Example Statements:
```
OUTPUT 711;"SYST:COMM:SER:PAR ODD"
OUTPUT 711;"syst:comm:ser:rec:parity:type none"
```

Query Syntax:
```
SYSTem:COMMunicate:SERial[:RECeive]:PARity?
```

Return Format:
```
EVEN
ODD
NONE
```

Attribute Summary:
```
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

If you want to specify odd parity, send SYST:COMM:SERIAL:PARITY ODD.

If you want to specify even parity, send SYST:COMM:SERIAL:PARITY EVEN.

If a parity bit is *not* to be included, send SYST:COMM:SERIAL:PARITY NONE.

To turn on parity verification, send the SYSTem:COMMunicate:SERial:RECeive:PARity:CHECk ON command.

SYSTem:COMMunicate:SERial[:RECeive]:SBITs command/query

Specifies the number of "stop bits" sent with each character over the RS-232-C interface.

Command Syntax:

```
SYSTem:COMMunicate:SERial[:RECeive]:SBITs
<number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  1:2
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"SYST:COMM:SER:REC:SBIT 1"
OUTPUT 711;"system:communicate:serial:sbits 2"
```

Query Syntax:

```
SYSTem:COMMunicate:SERial[:RECeive]:SBITs?
```

Return Format:

```
Integer
```

Attribute Summary:

```
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Legal values are 1 and 2.

SYSTem:COMMunicate:SERial:TRANsmit:PACE command/query

Sets the RS-232 transmitter handshake pacing type.

Command Syntax: `SYSTem:COMMunicate:SERial:TRANsmit:PACE {NONE|XON|ACK}`

Example Statements:
```
OUTPUT 711;"SYST:COMM:SER:TRAN:PACE XON"
OUTPUT 711;"system:comm:serial:transmit:pace none"
```

Query Syntax: `SYSTem:COMMunicate:SERial:TRANsmit:PACE?`

Return Format:
```
XON
OFF
```

Attribute Summary:
```
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This command specifies the handshake type when the analyzer is sending data. It defines the protocol the analyzer expects from the device.

XON specifies the XON/XOFF protocol. The peripheral device must send XOFF (ASCII DC3, decimal code 19) to the analyzer to stop the transmission of data. When the peripheral device is ready to accept more data, it must send XON (ASCII DC1, decimal code 17) to the analyzer which then resumes transmission.

NONE specifies the absence of a protocol for pacing.

Check the documentation for your peripheral device to verify it supports the specified protocol.

Example Programs

This chapter contains listings of some of the programs on the HP 35665A Example Programs disk.

ARBSOURC Demonstrates programming the arbitrary source and transferring trace data to the data registers. The arbitrary source uses one of three waveforms, which have been downloaded to the data register. An existing trace is copied to the data register, uploaded, shifted left and right and then reloaded into the arbitrary source data register.

GRPDELAY Demonstrates programming simple marker functions and plotting group delay on the display.

MANARM Demonstrates using the Standard Event Register to detect a WAITING_FOR_ARM event and generates an SRQ. Program handles SRQ interrupt and allows you to arm the measurement.

OPC_SYNC Demonstrates synchronizing the program and the analyzer using the *OPC statement to set the OPERATION_COMPLETE bit in the Standard Event Status Register. The Status Register is masked to generate an SRQ when all pending operations have completed. The program handles the service request interrupt.

OPCQSYNC Demonstrates synchronizing the program and the analyzer using the *OPC? query. The programs pauses on an ENTER statement while it waits for the pending HP-IB operations to complete and for a "1" to be returned in response to the *OPC? query.

RPNCALC Demonstrates waveform math capabilities of the analyzer by creating a immediate mode waveform calculator. Uses data registers to implement Reverse Polish Notation calculations. Allows most waveform math functions to be performed using trace data, register data, and constants.

RS232CNF Displays the current configuration of the RS-232-C interface. Provides softkeys to change the configuration.

TWO_CTLR Demonstrates using an external controller to download an HP Instrument BASIC program, run it, and query variables.

WAI_SYNC Demonstrates synchronizing the program and the analyzer using the *WAI statement.

ARBSOURC

```
10 !-----------------------------------------------
20 ! HP BASIC Program: ARBSOURCE
30 ! This program is used with the HP 35665a
40 ! Dynamic Signal Analyzer.  It allows a user
50 ! to create ramp, triangle, and square waves
60 ! and download them to an analyzer data register
70 ! to be used by the arbitrary source.
80 ! It also allows captured traces to be uploaded
90 ! to the program, shifted left or right, and then
100 ! downloaded into a data register.
110 !-----------------------------------------------
120  COM /Traces/ Trace_in(1:1024),Trace_out(1:1024)
130  COM /Assigns/ @Format_off,@Format_on,@Hp35665a
140  ASSIGN @Format_on TO 800;FORMAT ON
150  ASSIGN @Format_off TO 800;FORMAT OFF
160  ASSIGN @Hp35665a TO 800
170  INTEGER Byte_count,Block_count
180  ! Set data xfer to binary
190  OUTPUT @Format_on;"FORM:DATA REAL, 64"
200  Frequency=1024
210  High=2.5          ! high limit for arbitrary source
220  Low=-2.5          ! low limit for arbitrary source
230  Running=False
240 !
250 !-----------------------------------------------
260 !  Setup display, and arbitrary source
270 !-----------------------------------------------
280  OUTPUT @Hp35665a;"SYST:PRES"
290  OUTPUT @Hp35665a;"DISP:FORM ULOW"
300  OUTPUT @Hp35665a;"CALC1:FEED 'XTIM:VOLT 1'; *WAI"
310  OUTPUT @Hp35665a;"CALC2:FEED 'D1'; *WAI"
320  OUTPUT @Hp35665a;"ABOR;:INIT; *WAI"
330  OUTPUT @Hp35665a;"INIT:CONT OFF"
340  OUTPUT @Hp35665a;"TRAC D1,TRAC1; *WAI"
350  OUTPUT @Hp35665a;"DISP:WIND2:TRAC:Y:AUTO ONCE"
360  OUTPUT @Hp35665a;"SOUR:USER D1"
370  OUTPUT @Hp35665a;"SOUR:FUNC USER"
380  OUTPUT @Hp35665a;"SOUR:VOLT 0.99975 VPK"
390  OUTPUT @Hp35665a;"OUTP ON"
400 !
410 !-----------------------------------------------
420 !  Setup menu
430 !-----------------------------------------------
440  ON KEY 0 LABEL "RAMP" GOSUB Ramp
450  ON KEY 1 LABEL "TRIANGLE" GOSUB Triangle
460  ON KEY 2 LABEL "SQUARE" GOSUB Square
470  ON KEY 3 LABEL "COPY     A ->D1" GOSUB Storetrace
480  ON KEY 4 LABEL "LOAD     D1 " GOSUB Data_in
490  ON KEY 5 LABEL "SHIFT    TRACE" CALL Shift_trace
500  ON KEY 6 LABEL "" GOTO Waiting
```

```
510  ON KEY 7 LABEL "" GOTO Waiting
520  ON KEY 8 LABEL "PAUSE/   CONTINUE" GOSUB Pause_cont
530  ON KEY 9 LABEL "EXIT" GOTO Endit
540 !
550  Waiting:!GOTO Waiting
560  GOTO Waiting
570 !
580 Ramp:! create ramp wave in D1
590  GOSUB Enterdata
600  N=1
610  FOR I=1 TO 1024/Period
620  Current=Low
630   FOR J=1 TO Period
640     Trace_out(N)=Current
650     N=N+1
660     Current=Current+(High-Low)/Period
670   NEXT J
680  NEXT I
690  !
700  CALL Data_out ! send Trace_out to D1
710  RETURN
720 !
730 Triangle:! create triangle wave in D1
740  GOSUB Enterdata
750  N=1
760  FOR I=1 TO 1024/Period
770  Current=Low
780   FOR J=1 TO Period/2
790     Trace_out(N)=Current
800     N=N+1
810     Current=Current+(High-Low)/Period
820   NEXT J
830    Current=Current-(High-Low)/Period
840   FOR J=1 TO Period/2
850     Current=Current-(High-Low)/Period
860     Trace_out(N)=Current
870     N=N+1
880   NEXT J
890  NEXT I
900  CALL Data_out ! send Trace_out to D1
910  RETURN
920 !
930 Square:! create square wave in Trace_out array
940  GOSUB Enterdata
950  Toggle=1
960  N=1
970  Current=Low
980  FOR I=1 TO 1024/Period
990    FOR J=1 TO Period
1000     Trace_out(N)=Current
1010     N=N+1
1020    NEXT J
1030    IF Toggle THEN Current=High
```

```
1040     IF NOT Toggle THEN Current=Low
1050     Toggle=NOT Toggle
1060   NEXT I
1070   CALL Data_out ! send Trace_out to D1
1080   RETURN
1090 !
1100 Storetrace:! Copy Trace 1 to D1
1110   OUTPUT @Hp35665a;"TRAC D1,TRAC1"
1120   OUTPUT @Hp35665a;"DISP:WIND2:TRAC:Y:AUTO ONCE"
1130   RETURN
1140 !
1150 ! Enter frequency, which must be power of two so
1160 ! that waveform is symmetrical in 1024 point block
1170 Enterdata:!
1180   INPUT "Enter frequency (power of 2, from 128 up)",Frequency
1190   Period=128/Frequency*1024
1200   RETURN
1210 !
1220 Data_in:! Read D1 trace data into Trace_in array
1230   OUTPUT @Format_on;"TRAC:DATA? D1"
1240   ENTER @Format_on USING "%,A,D";A$,Byte_count
1250   ENTER @Format_on USING "%,4D";Block_count
1260   ENTER @Format_off;Trace_in(*)
1270   ENTER @Format_on;A$
1280   RETURN
1290 !
1300 Pause_cont:! Pause or continues measurement
1310   IF Running THEN
1320     OUTPUT @Hp35665a;"INIT:CONT OFF"
1330   ELSE
1340     OUTPUT @Hp35665a;"INIT:CONT ON"
1350   END IF
1360   Running=NOT Running
1370   RETURN
1380 !
1390 Endit:END
1400 !
1410 SUB Data_out
1420 !-----------------------------------------------
1430 ! Sends trace data from Trace_out array to D1
1440 ! data register in HP 35665a and sets source
1450 ! output as arbitrary source
1460 !-----------------------------------------------
1470 COM /Traces/ Trace_in(1:1024),Trace_out(1:1024)
1480 COM /Assigns/ @Format_off,@Format_on,@Hp35665a
1490     OUTPUT @Format_on;"TRAC:DATA D1,";
1500     OUTPUT @Format_on;"#48192";
1510     OUTPUT @Format_off;Trace_out(*);
1520     OUTPUT @Format_on;CHR$(10)
1530     OUTPUT @Hp35665a;"DISP:WIND2:TRAC:Y:AUTO ONCE"
1540     FOR I=1 TO 1024
1550       Trace_in(I)=Trace_out(I)
1560     NEXT I
```

```
1570 SUBEND
1580 !
1590 SUB Shift_trace
1600 !-----------------------------------------------
1610 ! Shifts trace left or right by defined stepsize
1620 ! and loads shifted trace into D1
1630 !-----------------------------------------------
1640 COM /Traces/ Trace_in(1:1024),Trace_out(1:1024)
1650 Stepsize=10
1660 ON KEY 0 LABEL "",2 GOTO Waiting
1670 ON KEY 1 LABEL "SHIFT    LEFT",2 GOSUB Shift_left
1680 ON KEY 2 LABEL "SHIFT    RIGHT",2 GOSUB Shift_right
1690 ON KEY 3 LABEL "",2 GOTO Waiting
1700 ON KEY 4 LABEL "",2 GOTO Waiting
1710 ON KEY 5 LABEL "DEFINE   STEPSIZE",2 GOSUB Step_size
1720 ON KEY 6 LABEL "",2 GOTO Waiting
1730 ON KEY 7 LABEL "",2 GOTO Waiting
1740 ON KEY 8 LABEL "",2 GOTO Waiting
1750 ON KEY 9 LABEL "RETURN",2 GOTO Sub_end
1760 !
1770 Waiting:GOTO Waiting
1780 !
1790 Shift_left:! shift the trace left <Stepsize> bins
1800 I=1
1810 FOR J=Stepsize TO 1024
1820  Trace_out(I)=Trace_in(J)
1830  I=I+1
1840 NEXT J
1850 FOR J=1 TO Stepsize-1
1860   Trace_out(I)=Trace_in(J)
1870   I=I+1
1880 NEXT J
1890 CALL Data_out
1900 RETURN
1910 !
1920 Shift_right:! shift the trace right <Stepsize> bins
1930 I=1
1940 FOR J=Stepsize TO 1024
1950  Trace_out(J)=Trace_in(I)
1960  I=I+1
1970 NEXT J
1980 FOR J=1 TO Stepsize-1
1990   Trace_out(J)=Trace_in(I)
2000   I=I+1
2010 NEXT J
2020 CALL Data_out ! send trace to D1
2030 RETURN
2040 Step_size:!
2050 INPUT "Enter shift stepsize (1..1024)",Stepsize
2060 RETURN
2070 Copytrace:!
2080 FOR I=1 TO 1024 ! update Trace_in with new data
2090   Trace_in(I)=Trace_out(I)
```

```
2100 NEXT I
2110 RETURN
2120 !
2130 Sub_end:SUBEND
```

GRPDELAY

```
10  !-----------------------------------------
20  ! HP Instrument BASIC example program:
30  !
40  ! GRPDELAY
50  !
60  ! This program plots group delay on the display
70  ! and provides simple marker functions.
80  ! For use with the HP 35665A Dynamic Signal
90  ! Analyzer using HP Instrument BASIC.
100  !-----------------------------------------
110  ! Initialize variables
120  Blocksize=401
130  DIM Scaledtrc(1:401)
140  COM Trace_in(1:401),Grpdelay(1:401)
150  W=402                          ! width of window
160  H=150                          ! height of window
170  Xoff=72                        ! offset from left
180  Yoff=19                        ! offset from bottom
190  Apert=.005                     ! default aperture
200  Currentbin=200                 ! marker origin
210  Hp35665=800                    ! internal bus addr
220  ASSIGN @Format_on TO Hp35665;FORMAT ON
230  ASSIGN @Format_off TO Hp35665;FORMAT OFF
240  ASSIGN @Hp35665a TO Hp35665
250  INTEGER Byte_count,Block_count
260  ! Set up display
270  CLEAR SCREEN
280  GCLEAR
290  OUTPUT @Hp35665a;"INIT:CONT OFF"
300  OUTPUT @Hp35665a;"DISP:PROG LOW"
310  ! Print instructions and see if demo setup wanted
320  Instructions(Demo)
330  IF Demo=0 THEN GOTO Skipdemo
340  ! This is a demonstration measurement setup.
350  ! To use it connect the source output directly
360  ! to channels A and B.
370  ! It performs an FRF of the a weight filter in
380  ! Channel 2 input.
390  OUTPUT @Hp35665a;"SYST:PRES"
400  OUTPUT @Hp35665a;"CAL:AUTO OFF"
410  OUTPUT @Hp35665a;"SOUR:FUNC RAND"
420  OUTPUT @Hp35665a;"SOUR:VOLT 1 Vrms"
430  OUTPUT @Hp35665a;"OUTP ON"
440  OUTPUT @Hp35665a;"INIT:CONT OFF"
450  OUTPUT @Hp35665a;"INP2 ON"
460  OUTPUT @Hp35665a;"INP2:FILT:AWE ON"
470  OUTPUT @Hp35665a;"CALC1:FEED 'XFR:POW:RAT 2,1'; *WAI"
480  OUTPUT @Hp35665a;"FREQ:SPAN 13000 Hz"
490  OUTPUT @Hp35665a;"FREQ:STAR 200 Hz"
```

```
500  OUTPUT @Hp35665a;"AVER ON;*WAI"
510  OUTPUT @Hp35665a;"ABOR;:INIT; *WAI"
520  OUTPUT @Hp35665a;"DISP:PROG LOW"
530 Skipdemo:!
540  ! Set trace coordinate to unwrapped phase
550  OUTPUT @Hp35665a;"CALC1:FORM UPH; *WAI"
560  ! Autoscale it
570  OUTPUT @Hp35665a;"DISP:WIND1:TRAC:Y:AUTO ONCE"
580  ! Set data xfer to binary
590  OUTPUT @Format_on;"FORM:DATA REAL, 64"
600  ! Get block of trace data
610  OUTPUT @Format_on;"CALC:DATA?"
620  ENTER @Format_on USING "%,A,D";A$,Byte_count
630  ENTER @Format_on USING "%,"&VAL$(Byte_count)&"D";Block_count
640  ENTER @Format_off;Trace_in(*)
650  ENTER @Format_on;A$
660  ! Get current span
670  OUTPUT @Hp35665a;"FREQ:SPAN?"
680  ENTER @Format_on;Span$
690  Span=VAL(Span$)
700  ! Differentiate phase of frequency data
710  Diff(Apert,Span,Ymax,Ymin,Dx) ! returns Ymax,Ymin,Dx
720  Lowerbounds=(Dx/2)+1          ! where dy/dx data starts
730  Upperbounds=Blocksize-(Dx/2)  ! and stops
740  ! Scale group delay data between Ymin and Ymax
750  FOR I=Lowerbounds TO Upperbounds
760    Scaledtrc(I)=FNScaley(Grpdelay(I),Ymin,Ymax,H)
770  NEXT I
780  ! Plot group delay on CRT
790  GCLEAR
800  Drawbox(H,W,Xoff,Yoff)
810  Drawtrace(Scaledtrc(*),Xoff,Yoff,Dx)
820  Labelplot(Ymax,Ymin,@Hp35665a)
830  ! Draw cursor mark
840  Drawmark(Xoff+Currentbin,Yoff+Scaledtrc(Currentbin))
850  !
860  DISP "Aperture = .5% of span"
870  ! Set up softkeys
880  ON KEY 0 LABEL "APERT .5%OF SPAN" GOSUB Halfpct
890  ON KEY 1 LABEL "APERT  1%OF SPAN" GOSUB Onepct
900  ON KEY 2 LABEL "APERT  2%OF SPAN" GOSUB Twopct
910  ON KEY 3 LABEL "APERT  4%OF SPAN" GOSUB Fourpct
920  ON KEY 4 LABEL "APERT  8%OF SPAN" GOSUB Eightpct
930  ON KEY 5 LABEL "APERT 16%OF SPAN" GOSUB Sixteenpct
940  ON KEY 6 LABEL "MARKER   LEFT" GOSUB Left
950  ON KEY 7 LABEL "MARKER   RIGHT" GOSUB Right
960  ON KEY 8 LABEL "MOVE     MARKER TO" GOSUB Jumpmrk
970  ON KEY 9 LABEL "EXIT" GOTO Ex
980 Waiting:GOTO Waiting
990 Halfpct:!
1000  Apert=.005
1010  DISP "Aperture = .5% of span"
1020  GOSUB Grpdisp
```

```
1030  RETURN
1040 Onepct:!
1050  Apert=.01
1060  DISP "Aperture = 1% of span"
1070  GOSUB Grpdisp
1080  RETURN
1090 Twopct:!
1100  Apert=.02
1110  DISP "Aperture = 2% of span"
1120  GOSUB Grpdisp
1130  RETURN
1140 Fourpct:!
1150  DISP "Aperture = 4% of span"
1160  Apert=.04
1170  GOSUB Grpdisp
1180  RETURN
1190 Eightpct:!
1200  Apert=.08
1210  DISP "Aperture = 8% of span"
1220  GOSUB Grpdisp
1230  RETURN
1240 Sixteenpct:!
1250  DISP "Aperture = 16% of span"
1260  Apert=.16
1270  GOSUB Grpdisp
1280  RETURN
1290 Grpdisp:! Recalculate group delay and replot
1300  GCLEAR
1310  Drawbox(H,W,Xoff,Yoff)
1320  Diff(Apert,Span,Ymax,Ymin,Dx)
1330  Lowerbounds=Dx/2+1
1340  Upperbounds=Blocksize-Dx/2
1350  ! scale group delay
1360  FOR I=Lowerbounds TO Upperbounds
1370     Scaledtrc(I)=FNScaley(Grpdelay(I),Ymin,Ymax,H)
1380  NEXT I
1390  Drawtrace(Scaledtrc(*),Xoff,Yoff,Dx) ! draw group delay function
1400  Labelplot(Ymax,Ymin,@Hp35665a)
1410  ! draw cursor mark
1420  IF Currentbin<Lowerbounds THEN Currentbin=Lowerbounds
1430  IF Currentbin>Upperbounds THEN Currentbin=Upperbounds
1440  Drawmark(Xoff+Currentbin,Yoff+Scaledtrc(Currentbin))
1450  RETURN
1460 Left: !Move marker left one bin
1470  Nextbin=-1
1480  OFF KEY 6
1490  GOSUB Movemark
1500  ON KEY 6 LABEL "MARKER   LEFT" GOSUB Left
1510  RETURN
1520 Right:!Move marker right one bin
1530  Nextbin=1
1540  OFF KEY 7
1550  GOSUB Movemark
```

```
1560  ON KEY 7 LABEL "MARKER   RIGHT" GOSUB Right
1570  RETURN
1580 Jumpmrk:! Move marker to desired bin
1590  DISP "Enter bin # between ";Lowerbounds;"and";Upperbounds;" "
1600  ENTER 2;Bin
1610  IF Bin<Lowerbounds OR Bin>Upperbounds THEN
1620    DISP "Must be between ";Lowerbounds;"and";Upperbounds
1630    GOTO Jumpmrk
1640  END IF
1650  Nextbin=Bin-Currentbin
1660 Movemark:!
1670  PEN 0    ! set pen to erase previous mark
1680  Drawmark(Xoff+Currentbin,Yoff+Scaledtrc(Currentbin))
1690  PEN 1    ! set pen to draw new mark
1700  ! redraw local area of trace first
1710  Lowmark=Currentbin-6
1720  Himark=Currentbin+6
1730  IF Lowmark<Lowerbounds THEN Lowmark=Lowerbounds
1740  IF Himark>Upperbounds THEN Himark=Upperbounds
1750  MOVE Xoff+Lowmark,Yoff+Scaledtrc(Lowmark)
1760  FOR I=Lowmark TO Himark
1770    DRAW I+Xoff,Yoff+Scaledtrc(I)
1780  NEXT I
1790  Test=Nextbin+Currentbin
1800  IF (Test>Upperbounds) OR (Test<Lowerbounds) THEN GOTO Skip
1810  Currentbin=Currentbin+Nextbin
1820 Skip:!
1830  ! Draw new mark
1840  Drawmark(Xoff+Currentbin,Yoff+Scaledtrc(Currentbin))
1850  ! Display marker data
1860  Binwidth=Span/400
1870  DISP "X: ";(Currentbin-1)*Binwidth;"Hz    ";
1880  DISP "Y: ";
1890  DISP USING "MDDD.DDDE,#";Grpdelay(Currentbin)
1900  DISP " Sec  Bin #:  ";Currentbin
1910  RETURN
1920  Ex:END
1930 !
1940 SUB Drawbox(H,W,Xoff,Yoff)
1950  ! Draws a box around trace area
1960  ! H = height of box
1970  ! W = width of box
1980  ! Xoff = offset from bottom of partition
1990  ! Yoff = offset from left of partition
2000  MOVE Xoff,Yoff
2010  DRAW Xoff,Yoff+H
2020  DRAW Xoff+W,Yoff+H
2030  DRAW Xoff+W,Yoff
2040  DRAW Xoff,Yoff
2050  SUBEND
2060 !
2070 DEF FNScaley(Y,Ymin,Ymax,H)
2080  ! Returns value of Y scaled between Ymin and Ymax
```

```
2090  ! values on scale of H y-axis pixels
2100  REAL Scaled_y,Temp
2110  IF Ymax-Ymin<>0 THEN
2120    Temp=(Y-Ymin)/(Ymax-Ymin)
2130    Scaled_y=Temp*H
2140    RETURN Scaled_y
2150  ELSE
2160    RETURN 0
2170  END IF
2180 FNEND
2190 !
2200 SUB Drawmark(X,Y)
2210  ! Draws a cross marker at pixel location X,Y
2220  MOVE X,Y+5
2230  DRAW X,Y-5
2240  MOVE X-5,Y
2250  DRAW X+5,Y
2260 SUBEND
2270 !
2280 SUB Labelplot(Ymax,Ymin,@Dev)
2290  ! Labels X axis and Y axis
2300  Yunit$="Sec"
2310  Xunit$="Hz"
2320  PRINT TABXY(1,1);
2330  PRINT USING "MD.DDE,#";Ymax
2340  PRINT TABXY(3,2),Yunit$;
2350  PRINT TABXY(1,12);
2360  PRINT USING "MD.DDE,#";Ymin
2370  PRINT TABXY(3,13),Yunit$;
2380  OUTPUT @Dev;"DISP:WIND1:TRAC:X:LEFT?"
2390  ENTER @Dev;Xmin
2400  OUTPUT @Dev;"DISP:WIND1:TRAC:X:RIGHT?"
2410  ENTER @Dev;Xmax
2420  PRINT TABXY(9,14);Xmin;Xunit$;
2430  PRINT TABXY(51,14);
2440  PRINT USING "DDDDDD,#,2A";Xmax;Xunit$
2450 SUBEND
2460 !
2470 SUB Drawtrace(Trace(*),Xoff,Yoff,Dx)
2480  ! Plots the group delay trace on the screen
2490  INTEGER Start
2500  Start=(Dx/2)+1
2510  MOVE Start+Xoff,Yoff+Trace(Start)
2520  FOR I=Start+1 TO 401-Dx/2
2530    DRAW I+Xoff,Yoff+Trace(I)
2540  NEXT I
2550 SUBEND
2560 !
2570 SUB Diff(Apert,Sp,Ymax,Ymin,Dx)
2580  ! Differentiates trace_in by % of span (aperture)
2590  ! and returns the max and min values, and delta x
2600  ! (in bins) of the differentiation.
2610  COM Trace_in(1:401),Grpdelay(1:401)
```

```
2620  Binwidth=Sp/400
2630  Dx=(Sp*Apert)/Binwidth
2640  Ymax=0
2650  Ymin=0
2660  FOR I=Dx+1 TO 401
2670    J=(I)-(Dx/2)
2680    Grpdelay(J)=(Trace_in(I)-Trace_in(I-Dx))/(Dx*Binwidth*360)
2690    IF Grpdelay(J)>Ymax THEN Ymax=Grpdelay(J)
2700    IF Grpdelay(J)<Ymin THEN Ymin=Grpdelay(J)
2710  NEXT I
2720  ! Set Ymin and Ymax smaller to allow for overhead
2730  Ovr=.05*(Ymax-Ymin)   ! amount of trace overhead
2740  IF Ymin>0 THEN
2750    Ymax=Ymax-Ovr
2760  ELSE
2770    Ymax=Ymax+Ovr
2780  END IF
2790  IF Ymin>0 THEN
2800    Ymin=Ymin+Ovr
2810  ELSE
2820    Ymin=Ymin-Ovr
2830  END IF
2840 SUBEND
2850 !
2860 SUB Instructions(Demo)
2870  ! Provides instructions for using program
2880  ! and allows user to continue or stop
2890  ! or use default demo measurement setup
2900  PRINT "         This program generates a group delay function"
2910  PRINT "         from frequency response data in Trace A."
2920  PRINT
2930  PRINT "         Press CONTINUE if the analyzer is already in "
2940  PRINT "         FFT Analysis mode and Trace A contains a"
2950  PRINT "         frequency domain trace. (The trace coordinates"
2960  PRINT "         will be changed to unwrapped phase.)"
2970  PRINT
2980  PRINT "         Press USE DEMO SETUP to  run the program "
2990  PRINT "         without a DUT."
3000  PRINT
3010  PRINT "         After running the program you may use the "
3020  PRINT "         marker softkeys (NOT the hardkeys) to obtain"
3030  PRINT "         group delay data for individual frequencies.";
3040  ON KEY 0 LABEL "CONTINUE" GOTO Continue
3050  ON KEY 1 LABEL "USE DEMO SETUP" GOTO Demo
3060  ON KEY 2 LABEL "QUIT" GOTO Quit
3070  FOR I=3 TO 9
3080   ON KEY I LABEL "" GOTO Waiting
3090  NEXT I
3100  Waiting:GOTO Waiting
3110  !
3120  Quit:!
3130    DISP "Halting program"
3140    OUTPUT 800;"DISP:PROG OFF"
```

```
3150     STOP
3160   Demo:!
3170     Demo=1
3180     CLEAR SCREEN
3190     PRINT
3200     PRINT
3210     PRINT "           This demonstration uses the internal A Weight"
3220     PRINT "           filter of Channel 2 as the device under "
3230     PRINT "           test of a frequency response measurement."
3240     PRINT "           Connect the output of the source to both"
3250     PRINT "           channels and press ENTER"
3260     INPUT "Press ENTER when ready...",A$
3270     GOTO Subend
3280   Continue:!
3290     Demo=0
3300 Subend:CLEAR SCREEN
3310 SUBEND
```

MANARM

```
10  ! HP Instrument BASIC example program: MANARM
20  ! ----------------------------------------------------------------
30   !
40  ! This program demonstrates using the instrument's
50  ! status registers to enable SRQs for event-
60  ! initiated program interrupts. In this case the
70  ! waiting_for_arm bit is detected.
80   !
90  ! ----------------------------------------------------------------
100  Sc=8
110  Addr=0
120  Device=(Sc*100)+Addr
130  ASSIGN @Hp35665a TO Device
140  CLEAR SCREEN
150  OUTPUT @Hp35665a;"SYST:PRES"
160  IF Sc=8 THEN
170    OUTPUT @Hp35665a;"DISP:FORM ULOW"
180    OUTPUT @Hp35665a;"DISP:PROG LOW"
190    CLEAR SCREEN
200  END IF
210 !
220 ! Setup registers to detect WAITING_FOR_ARM
230 !
240 ! clear any pending events
250  OUTPUT @Hp35665a;"*CLS"
260 ! allow SRQ from operation register
270  OUTPUT @Hp35665a;"*SRE 128"
280 ! allow SRQ from waiting_for_arm bit
290  OUTPUT @Hp35665a;"STAT:OPER:ENAB 64"
300 ! latch waiting_for_arm TRUE
310  OUTPUT @Hp35665a;"STAT:OPER:PTR 64"
320 ! do not latch waiting_for_arm FALSE
330  OUTPUT @Hp35665a;"STAT:OPER:NTR 0"
340 !
350 ! set up interrupts
360  ON INTR Sc GOSUB Check_srq
370  ENABLE INTR Sc;2
380 !
390 OUTPUT @Hp35665a;"FREQ:SPAN 100 Hz"
400 OUTPUT @Hp35665a;"ARM:SOUR MAN"
410 OUTPUT @Hp35665a;"ABOR;:INIT"
420  !
430  ! Wait for SRQ
440  !
450 Hang_out:GOTO Hang_out
460  !
470 Check_srq:    !
480  !
490    PRINT "SRQ Received"
500    Sb=SPOLL(Device)
```

```
510    PRINT "SPOLL(";Device;") = ";Sb
520    Queryarm(@Hp35665a)
530    ENABLE INTR Sc
540    RETURN
550  !
560  END
570  !*****************************************************
580  ! Query Standard Event Status Register and arm
590  ! if waiting_for_arm event detected
600  !*****************************************************
610  SUB Queryarm(@Device)
620   OUTPUT @Device;"STAT:OPER:EVEN?"
630   ENTER @Device;Resp
640   PRINT "STAT:OPER:EVEN?: ";Resp
650   IF Resp=64 THEN
660      INPUT "PRESS ENTER TO ARM (ENTER 'Q' TO QUIT)",A$
670      IF UPC$(A$)="Q" THEN STOP
680      OUTPUT @Device;"ARM"
690      PRINT "ARMED!"
700      PRINT
710   END IF
720   SUBEND
```

OPC_SYNC

```
10 ! HP Instrument BASIC program: OPCSYNC - Measurement synchronization
20 ! ----------------------------------------------------------------
30 ! This program demonstrates how to use the *OPC command to
40 ! allow an SRQ to interrupt program execution.  *OPC will set
50 ! the OPERATION_COMPLETE bit in the EVENT STATUS register
60 ! when all pending HP-IB commands have finished.  With the proper
70 ! register masks, this will generate a service request.
80 ! ----------------------------------------------------------------
90 !
100 Scode=8                              ! Interface select code
110 Address=0
120 Hp35665a=Scode*100+Address
130 !
140 OUTPUT Hp35665a;"FREQ:SPAN 50 HZ" !Measurement will take 8 seconds
150 OUTPUT Hp35665a;"*CLS"             !Clear the STATUS BYTE register
160 OUTPUT Hp35665a;"*ESE 1"           !Program the EVENT STATUS ENABLE reg.
170 OUTPUT Hp35665a;"*SRE 32"          !Program the STATUS BYTE ENABLE reg.
180 !
190 ON INTR Scode,2 GOTO Srq_handler   !Set up interrupt branching
200 ENABLE INTR Scode;2                !Allow SRQ to generate an interrupt
210 !
220 OUTPUT Hp35665a;"ABORT; INIT"      !Start the measurement
230 OUTPUT Hp35665a;"*OPC"             !Generate SRQ when all commands have
240                                    !finished.
250 Start_time=TIMEDATE
260 LOOP                               !Do something useful while waiting
270   DISP USING "14A, 2D.D";"Elapsed time :",TIMEDATE-Start_time
280   WAIT .1
290 END LOOP
300 !
310 Srq_handler:                        !Got an SRQ
320  Stb=SPOLL(Hp35665a)                !Read STATUS BYTE and clear SRQ
330  BEEP
340  OUTPUT Hp35665a;"*ESR?"            !Read and clear EVENT STATUS reg.
350  ENTER Hp35665a;Esr
360  DISP "Got the SRQ! SPOLL returns:";Stb;"   ESR returns:";Esr
370 END
```

OPCQSYNC

```
10 ! HP Instrument BASIC program: OPCQSYNC -  Measurement synchronization
20 ! ----------------------------------------------------------------------
30 ! This program demonstrates how to use the *OPC? HP-IB command
40 ! to hang the bus on a query before continuing on with the
50 ! program.  After all pending HP-IB commands have finished,
60 ! the HP 35665a will return a '1' in response to *OPC?. 70 !
----------------------------------------------------------------------
80 !
90 Scode=8
100 Hp35665a=Scode*100
110 !
120 OUTPUT Hp35665a;"SYST:PRES"          !Preset the HP35665a 130 OUTPUT
    Hp35665a;"*OPC?"              !Pause on ENTER statement until
140 ENTER Hp35665a;Opc                   !'*RST' command has finished
150 !
160 OUTPUT Hp35665a;"FREQ:SPAN 50 Hz"  !Measurement will take 8 seconds
170 DISP "Measurement started ..."
180 OUTPUT Hp35665a;"ABOR; INIT"         !Start the measurement
190 OUTPUT Hp35665a;"*OPC?"              !Pause until all pending HP-IB
    commands
200 ENTER Hp35665a;Opc                   !have finished.
```

RPNCALC

```
10 !=============================================
20 !
30 ! RPNCALC
40 ! Reverse Polish Notation Waveform Calculator
50 !
60 ! This HP Instrument BASIC program runs on the
70 ! HP 35665a Dynamic Signal Analyzer.
80 !
90 ! It uses data registers D1 through D4 to emulate
100 ! the stack common to most HP RPN calculators.
110 ! Traces from either the upper or lower trace
120 ! displays may be loaded into the X register (D1).
130 !
140 ! Unary operations (e.g., FFT and CONJ) operate
150 ! on the X register.  Binary operations (+-/*)
160 ! operate on the X and Y registers (D1 and D2).
170 ! All math operations place results in D1.
180 !
190 ! Rotate and Enter functions are available as well
200 ! as the ability to display various trace types
210 ! and to adjust trace coordinates and autoscale.
220 !
230 !=============================================
240 COM @Hp35665a
250 DIM Trca$[20],Trcb$[20],Trc$[20]
260 ASSIGN @Hp35665a TO 800
270 OUTPUT @Hp35665a;"DISP:FORM ULOW"
280 OUTPUT @Hp35665a;"INP2 ON"
290 Activetrc=1
300 OUTPUT @Hp35665a;"CALC1:ACT ON"
310 Keys:!
320   ON KEY 0 LABEL "LOAD X " CALL Loadx
330   ON KEY 1 LABEL "ENTER" GOSUB Ent
340   ON KEY 2 LABEL "ROTATE     UP" GOSUB Rot_up
350   ON KEY 3 LABEL "ROTATE    DOWN" GOSUB Rot_down
360   ON KEY 4 LABEL "" GOTO Waiting
370   ON KEY 5 LABEL "FUNCTIONS" CALL Functions
380   ON KEY 6 LABEL "+ - / * " CALL Ops
390   ON KEY 7 LABEL "DISPLAY  TRACE" CALL Select_trc
400   ON KEY 8 LABEL "" GOTO Waiting
410   ON KEY 9 LABEL "EXIT" GOTO Exit1
420 !
430 Waiting:GOTO Waiting !
440 !
450 Ent:      ! shifts up without replacing D1
460    Shift(1)
470    RETURN
480 !
490 Rot_up:   !shifts D1-D4 up and copies D5 to D1
500   Shift(1)
```

```
510   OUTPUT @Hp35665a;"TRAC D1, D5"
520   RETURN
530 Rot_down: !copies D1 to D5 and shifts D5-D2 down
540   OUTPUT @Hp35665a;"TRAC D5, D1"
550   Shift(0)
560   RETURN
570 !
580 Exit1:!
590 END
600 SUB Loadx
610 !-----------------------------------------
620 ! Loads D1 register with the contents of
630 ! Trace A, Trace B, or a Constant
640 ! and shifts D2-D4 up
650 !-----------------------------------------
660 COM @Hp35665a
670 !
680   ON KEY 0 LABEL "TRACE A",2 GOSUB Channel1
690   ON KEY 1 LABEL "TRACE B",2 GOSUB Channel2
700   ON KEY 2 LABEL "",2 GOTO Twiddle
710   ON KEY 3 LABEL "CONSTANT",2 GOTO Constant
720   ON KEY 4 LABEL "",2 GOTO Twiddle
730   ON KEY 5 LABEL "",2 GOTO Twiddle
740   ON KEY 6 LABEL "",2 GOTO Twiddle
750   ON KEY 7 LABEL "",2 GOTO Twiddle
760   ON KEY 8 LABEL "",2 GOTO Twiddle
770   ON KEY 9 LABEL "RETURN",2 GOTO Sub_exit
780 !
790  Twiddle:GOTO Twiddle
800 !
810 Channel1:!
820   CALL Shift(1)
830   OUTPUT @Hp35665a;"TRAC D1,TRAC1"
840   SUBEXIT
850 !
860 Channel2:!
870   CALL Shift(1)
880   OUTPUT @Hp35665a;"TRAC D1,TRAC2"
890   SUBEXIT
900 !
910 Constant:!
920   CALL Shift(1)
930   CALL Sel_const
940 Sub_exit:!
950 SUBEND
960 !
970 SUB Shift(Direction)
980 !-----------------------------------------
990 ! Shifts stack (D1-D4) up or down by one
1000 ! using D5 as a temporary buffer
1010 !-----------------------------------------
1020 COM @Hp35665a
1030 IF Direction=1 THEN    ! shift up
```

```
1040    OUTPUT @Hp35665a;"TRAC D5, D4"
1050    OUTPUT @Hp35665a;"TRAC D4, D3"
1060    OUTPUT @Hp35665a;"TRAC D3, D2"
1070    OUTPUT @Hp35665a;"TRAC D2, D1"
1080  ELSE                    ! shift down
1090    OUTPUT @Hp35665a;"TRAC D1, D2"
1100    OUTPUT @Hp35665a;"TRAC D2, D3"
1110    OUTPUT @Hp35665a;"TRAC D3, D4"
1120    OUTPUT @Hp35665a;"TRAC D4, D5"
1130  END IF
1140  SUBEND
1150  !
1160  SUB Functions
1170  !-----------------------------------------
1180  ! Performs immediate unary math on D1 trace
1190  !-----------------------------------------
1200     COM @Hp35665a
1210     ON KEY 0 LABEL "CONJ",2 GOSUB Conj
1220     ON KEY 1 LABEL "MAG",2 GOSUB Mag
1230     ON KEY 2 LABEL "REAL",2 GOSUB Realpart
1240     ON KEY 3 LABEL "IMAG",2 GOSUB Imagpart
1250     ON KEY 4 LABEL "SQRT",2 GOSUB Sqroot
1260     ON KEY 5 LABEL "FFT",2 GOSUB Fft
1270     ON KEY 6 LABEL "IFFT",2 GOSUB Ifft
1280     ON KEY 7 LABEL "PSD",2 GOSUB Psd
1290     ON KEY 8 LABEL "LN",2 GOSUB Ln
1300     ON KEY 9 LABEL "EX",2 GOSUB Expn
1310  Waiting:GOTO Waiting
1320  Conj:!
1330     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (CONJ(D1))"
1340     GOTO Calc
1350  Mag:!
1360     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (MAG(D1))"
1370     GOTO Calc
1380  Realpart:!
1390     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (REAL(D1))"
1400     GOTO Calc
1410  Imagpart:!
1420     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (IMAG(D1))"
1430     GOTO Calc
1440  Sqroot:!
1450     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (SQRT(D1))"
1460     GOTO Calc
1470  Fft:!
1480     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (FFT(D1))"
1490     GOTO Calc
1500  Ifft:!
1510     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (IFFT(D1))"
1520     GOTO Calc
1530  Psd:!
1540     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (PSD(D1))"
1550     GOTO Calc
1560  Ln:!
```

```
1570     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (LN(D1))"
1580     GOTO Calc
1590 Expn:!
1600     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (EXP(D1))"
1610 Calc:!  Performs calculation and stores it in D1
1620   OUTPUT @Hp35665a;"CALC:MATH:STATE ON"
1630   OUTPUT @Hp35665a;"TRAC D1, TRAC1"
1640   OUTPUT @Hp35665a;"CALC:MATH:STATE OFF"
1650 SUBEND
1660 !
1670 SUB Tracedisplay(Trace)
1680 !-----------------------------------------
1690 ! Allows selected trace (A or B)
1700 ! to be stored to D6-D8, replaced by D1-D8,
1710 ! autoscaled, or have its trace coordinates
1720 ! changed.
1730 ! Called by: Select_trc
1740 !-----------------------------------------
1750 COM @Hp35665a
1760     ON KEY 0 LABEL "",3 GOTO Waiting
1770     ON KEY 1 LABEL "STORE IN D6",3 GOSUB D6
1780     ON KEY 2 LABEL "STORE IN D7",3 GOSUB D7
1790     ON KEY 3 LABEL "STORE IN D8",3 GOSUB D8
1800     ON KEY 4 LABEL "",3 GOTO Waiting
1810     ON KEY 5 LABEL "RECALL   D1 - D8",3 GOSUB Disp
1820     ON KEY 6 LABEL "",3 GOTO Waiting
1830     ON KEY 7 LABEL "TRACE    COORD",3 GOSUB Coord
1840     ON KEY 8 LABEL "AUTOSCALE",3 GOSUB Auto
1850     ON KEY 9 LABEL "RETURN",3 GOTO Sub_exit
1860 !
1870 Waiting:GOTO Waiting
1880 !
1890 D6:!  copy trace to D6
1900   OUTPUT @Hp35665a;"TRAC D6, TRAC"&VAL$(Trace)
1910   RETURN
1920 D7:!  copy trace to D7
1930   OUTPUT @Hp35665a;"TRAC D7, TRAC"&VAL$(Trace)
1940   RETURN
1950 D8:!  copy trace to D8
1960   OUTPUT @Hp35665a;"TRAC D8, TRAC"&VAL$(Trace)
1970   RETURN
1980 Disp:! call Disp_regs
1990   Disp_regs(Trace)
2000   RETURN
2010 Coord:! call Trace_coord
2020   Trace_coord(Trace)
2030   RETURN
2040 Auto:! autoscale trace
2050   OUTPUT @Hp35665a;"DISP:WIND"&VAL$(Trace)&":TRAC:Y:AUTO ONCE"
2060   RETURN
2070 Sub_exit:!
2080 SUBEND
2090 !
```

```
2100 SUB Select_trc
2110 !-----------------------------------------
2120 ! Selects either Trace A or Trace B as active
2130 ! trace for Tracedisplay routines
2140 !-----------------------------------------
2150    ON KEY 0 LABEL "",2 GOTO Waiting
2160    ON KEY 1 LABEL "UPPER",2 GOTO Sel_trc1
2170    ON KEY 2 LABEL "LOWER",2 GOTO Sel_trc2
2180    ON KEY 3 LABEL "",2 GOTO Waiting
2190    ON KEY 4 LABEL "",2 GOTO Waiting
2200    ON KEY 5 LABEL "",2 GOTO Waiting
2210    ON KEY 6 LABEL "",2 GOTO Waiting
2220    ON KEY 7 LABEL "",2 GOTO Waiting
2230    ON KEY 8 LABEL "",2 GOTO Waiting
2240    ON KEY 9 LABEL "RETURN",2 GOTO Sub_exit
2250 Waiting:GOTO Waiting
2260 Sel_trc1:!
2270    Tracedisplay(1)
2280    SUBEXIT
2290 Sel_trc2:!
2300    Tracedisplay(2)
2310    SUBEXIT
2320 Sub_exit:!
2330 SUBEND
2340 !
2350 SUB Disp_regs(Trc)
2360 !-----------------------------------------
2370 ! Displays selected register in trace A or B
2380 ! Called by: Tracedisplay
2390 !-----------------------------------------
2400 COM @Hp35665a
2410    ON KEY 0 LABEL "D1",4 GOSUB D1
2420    ON KEY 1 LABEL "D2",4 GOSUB D2
2430    ON KEY 2 LABEL "D3",4 GOSUB D3
2440    ON KEY 3 LABEL "D4",4 GOSUB D4
2450    ON KEY 4 LABEL "D5",4 GOSUB D5
2460    ON KEY 5 LABEL "D6",4 GOSUB D6
2470    ON KEY 6 LABEL "D7",4 GOSUB D7
2480    ON KEY 7 LABEL "D8",4 GOSUB D8
2490    ON KEY 8 LABEL "",4 GOTO Waiting
2500    ON KEY 9 LABEL "RETURN",4 GOTO Sub_exit
2510 !
2520 Waiting:GOTO Waiting
2530 !
2540 D1:!
2550   OUTPUT @Hp35665a;"CALC"&VAL$(Trc)&":FEED 'D1'; *WAI"
2560   RETURN
2570 D2:!
2580   OUTPUT @Hp35665a;"CALC"&VAL$(Trc)&":FEED 'D2'; *WAI"
2590   RETURN
2600 D3:!
2610   OUTPUT @Hp35665a;"CALC"&VAL$(Trc)&":FEED 'D3'; *WAI"
2620   RETURN
```

```
2630 D4:!
2640   OUTPUT @Hp35665a;"CALC"&VAL$(Trc)&":FEED 'D4'; *WAI"
2650   RETURN
2660 D5:!
2670   OUTPUT @Hp35665a;"CALC"&VAL$(Trc)&":FEED 'D5'; *WAI"
2680   RETURN
2690 D6:!
2700   OUTPUT @Hp35665a;"CALC"&VAL$(Trc)&":FEED 'D6'; *WAI"
2710   RETURN
2720 D7:!
2730   OUTPUT @Hp35665a;"CALC"&VAL$(Trc)&":FEED 'D7'; *WAI"
2740   RETURN
2750 D8:!
2760   OUTPUT @Hp35665a;"CALC"&VAL$(Trc)&":FEED 'D8'; *WAI"
2770   RETURN
2780 Sub_exit:!
2790 SUBEND
2800 !
2810 SUB Trace_coord(Trace)
2820 !-----------------------------------------
2830 ! Selects trace coordinates for trace
2840 ! Called by: Tracedisplay
2850 !-----------------------------------------
2860    COM @Hp35665a
2870    T$=VAL$(Trace)
2880    !
2890    ON KEY 0 LABEL "LINEAR   MAGNITUDE",4 GOSUB Lin
2900    ON KEY 1 LABEL "LOG      MAGNITUDE",4 GOSUB Log
2910    ON KEY 2 LABEL "DB       MAGNITUDE",4 GOSUB Db
2920    ON KEY 3 LABEL "PHASE",4 GOSUB Phase
2930    ON KEY 4 LABEL "UNWRAPPEDPHASE",4 GOSUB Unwrapped
2940    ON KEY 5 LABEL "REAL     PART",4 GOSUB Realpart
2950    ON KEY 6 LABEL "IMAGINARYPART",4 GOSUB Imagpart
2960    ON KEY 7 LABEL "NYQUIST  DIAGRAM",4 GOSUB Nyquist
2970    ON KEY 8 LABEL "",4 GOTO Waiting
2980    ON KEY 9 LABEL "RETURN",4 GOTO Sub_exit
2990 Waiting:GOTO Waiting
3000 !
3010 Lin:!
3020    OUTPUT @Hp35665a;"CALC"&T$&":FORM MLIN; *WAI"
3030    RETURN
3040 Log:!
3050    OUTPUT @Hp35665a;"CALC"&T$&":FORM MLIN"
3051    OUTPUT @Hp35665a;"DISP:WIND"&T$&":TRAC:Y:SPAC LOG; *WAI"
3060    RETURN
3070 Db:!
3080    OUTPUT @Hp35665a;"CALC"&T$&":FORM MLOG; *WAI"
3090    RETURN
3100 Phase:!
3110    OUTPUT @Hp35665a;"CALC"&T$&":FORM PHAS; *WAI"
3120    RETURN
3130 Unwrapped:!
```

```
3140     OUTPUT @Hp35665a;"CALC"&T$&":FORM UPH; *WAI"
3150     RETURN
3160 Realpart:!
3170     OUTPUT @Hp35665a;"CALC"&T$&":FORM REAL; *WAI"
3180     RETURN
3190 Imagpart:!
3200     OUTPUT @Hp35665a;"CALC"&T$&":FORM IMAG; *WAI"
3210     RETURN
3220 Nyquist:!
3230     OUTPUT @Hp35665a;"CALC"&T$&":FORM NYQ; *WAI"
3240     RETURN
3250 Sub_exit:!
3260 SUBEND
3270 !
3280 SUB Ops
3290 !-----------------------------------------
3300 ! Performs binary operations on D1 and D2
3310 !-----------------------------------------
3320     COM @Hp35665a
3330     ON KEY 0 LABEL "   +",4 GOSUB Plus
3340     ON KEY 1 LABEL "   -",4 GOSUB Minus
3350     ON KEY 2 LABEL "   /",4 GOSUB Divide
3360     ON KEY 3 LABEL "   *",4 GOSUB Mult
3370     FOR I=4 TO 9
3380     ON KEY I LABEL "",4 GOTO Waiting
3390     NEXT I
3400 !
3410 Waiting:GOTO Waiting
3420 !
3430 Plus:!
3440     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (D1 + D2)"
3450     GOTO Calc
3460 Minus:!
3470     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (D2 - D1)"
3480     GOTO Calc
3490 Divide:!
3500     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (D2 / D1)"
3510     GOTO Calc
3520 Mult:!
3530     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (D1 * D2)"
3540     GOTO Calc
3550 Calc:!  perform calculation and place in D1
3560     OUTPUT @Hp35665a;"CALC:MATH:STATE ON"
3570     OUTPUT @Hp35665a;"TRAC D1, TRAC1"
3580     OUTPUT @Hp35665a;"CALC:MATH:STATE OFF"
3590 SUBEND
3600 !
3610 SUB Sel_const
3620 !-----------------------------------------
3630 ! Allows a constant to be loaded into D1
3640 ! Called by: Loadx
3650 !-----------------------------------------
3660     COM @Hp35665a
```

```
3670     ON KEY 0 LABEL "K1",4 GOTO K1
3680     ON KEY 1 LABEL "K2",4 GOTO K2
3690     ON KEY 2 LABEL "K3",4 GOTO K3
3700     ON KEY 3 LABEL "K4",4 GOTO K4
3710     ON KEY 4 LABEL "K5",4 GOTO K5
3720     ON KEY 5 LABEL "",4 GOTO Waiting
3730     ON KEY 6 LABEL "ENTER    CONSTANT",4 GOTO Enter_const
3740     ON KEY 7 LABEL "",4 GOTO Waiting
3750     ON KEY 8 LABEL "",4 GOTO Waiting
3760     ON KEY 9 LABEL "",4 GOTO Waiting
3770 Waiting:GOTO Waiting
3780     !
3790 K1:!
3800     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (K1)"
3810     GOTO Calc
3820 K2:!
3830     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (K2)"
3840     GOTO Calc
3850 K3:!
3860     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (K3)"
3870     GOTO Calc
3880 K4:!
3890     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (K4)"
3900     GOTO Calc
3910 K5:!
3920     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (K5)"
3930     GOTO Calc
3940 Enter_const:! enter constant value into K1
3950     INPUT "Enter constant value",C$
3960 OUTPUT @Hp35665a;"CALC:MATH:CONS1 "&C$
3970     OUTPUT @Hp35665a;"CALC:MATH:EXPR1 (K1)"
3980 Calc:!
3990     OUTPUT @Hp35665a;"CALC:MATH:STATE ON"
4000     ! constant stored in trace1; now copy to D1
4010     OUTPUT @Hp35665a;"TRAC D1, TRAC1"
4020     OUTPUT @Hp35665a;"CALC:MATH:STATE OFF"
4030 SUBEND
210 BEEP
220 DISP "Measurement done"
221 OUTPUT Hp35665a;"DISP:FORM ULOW"
223 OUTPUT Hp35665a;"INIT:CONT OFF"
230 END
```

RS232CNF

```
1 ! RS232CNF,  $Revision: 1.1 $
2 !
3 ! display the current rs232 interface
4 ! configuration parameters and provide
5 ! softkeys to allow the configuration
6 ! to be changed.
7 !
8 COM Parity_on,Baud,Bits,Sbits
9 COM In_pace$[10],Out_pace$[10],Parity$[10]
10 !
11 ! activate the IBASIC display
12 CLEAR SCREEN
13 OUTPUT 800;"DISP:PROG FULL"
14 !
15 Show_config
16 !
17 ! setup configuration change softkeys
18 ON KEY 0 LABEL "change   baud rate" GOSUB Baud_rate
19 ON KEY 1 LABEL "change   data bits" GOSUB Data_bits
20 ON KEY 2 LABEL "change   stop bits" GOSUB Stop_bits
21 ON KEY 3 LABEL "parity   enable" GOSUB Parity_enab
22 ON KEY 4 LABEL "parity   type" GOSUB Parity_type
23 ON KEY 5 LABEL "transmit pacing" GOSUB Out_pacing
24 ON KEY 6 LABEL "receive  pacing" GOSUB In_pacing
25 ON KEY 9 LABEL "pause    program" GOSUB Pause_prog
26 !
27 ! wait for a softkey press
28 LOOP
29 END LOOP
30 !
31 ! subroutines to handle configuration changes
32 Baud_rate: !
33 INPUT "enter new baud rate: ",Baud
34 OUTPUT 800;"SYST:COMM:SER:BAUD";Baud
35 Show_config
36 RETURN
37 !
38 Data_bits: !
39 INPUT "enter new number of data bits: ",Bits
40 OUTPUT 800;"SYST:COMM:SER:BITS";Bits
41 Show_config
42 RETURN
43 !
44 Stop_bits: !
45 INPUT "enter new number of stop bits: ",Sbits
46 OUTPUT 800;"SYST:COMM:SER:SBITS";Sbits
47 Show_config
48 RETURN
49 !
```

```
50 Parity_enab: !
51 INPUT "enter parity enable (0 => no, 1 => yes):",Parity_on
52 OUTPUT 800;"SYST:COMM:SER:PAR:CHEC";Parity_on
53 Show_config
54 RETURN
55 !
56 Parity_type: !
57 I=-1
58 INPUT "enter parity type (0 => none, 1 => even, 2 => odd):",I
59 IF I=0 THEN Parity$="NONE"
60 IF I=1 THEN Parity$="EVEN"
61 IF I=2 THEN Parity$="ODD"
62 OUTPUT 800;"SYST:COMM:SER:PAR ";Parity$
63 Show_config
64 RETURN
65 !
66 Out_pacing: !
67 I=-1
68 INPUT "enter output pacing type (0 => none, 1 => xon/xoff):",I
69 IF I=0 THEN Out_pace$="NONE"
70 IF I=1 THEN Out_pace$="XON"
71 OUTPUT 800;"SYST:COMM:SER:TRAN:PACE ";Out_pace$
72 Show_config
73 RETURN
74 !
75 In_pacing: !
76 I=-1
77 INPUT "enter receive pacing type (0 => none, 1 => xon/xoff):",I
78 IF I=0 THEN In_pace$="NONE"
79 IF I=1 THEN In_pace$="XON"
80 OUTPUT 800;"SYST:COMM:SER:PACE ";In_pace$
81 Show_config
82 RETURN
83 !
84 Pause_prog:PAUSE
85 RETURN
86 !
87 END
88 !
89 ! display the current rs-232 configuration
90 !
91 SUB Show_config
92 COM Parity_on,Baud,Bits,Sbits
93 COM In_pace$[10],Out_pace$[10],Parity$[10]
94 Parity_on$="NO"
95 OUTPUT 800;"SYST:COMM:SER:BAUD?"
96 ENTER 800;Baud
97 OUTPUT 800;"SYST:COMM:SER:BITS?"
98 ENTER 800;Bits
99 OUTPUT 800;"SYST:COMM:SER:PACE?"
100 ENTER 800;In_pace$
101 OUTPUT 800;"SYST:COMM:SER:PAR:CHEC?"
102 ENTER 800;Parity_on
```

```
103 IF Parity_on THEN Parity_on$="YES"
104 OUTPUT 800;"SYST:COMM:SER:PAR?"
105 ENTER 800;Parity$
106 OUTPUT 800;"SYST:COMM:SER:SBIT?"
107 ENTER 800;Sbits
108 OUTPUT 800;"SYST:COMM:SER:TRAN:PACE?"
109 ENTER 800;Out_pace$
110 !
111 CLEAR SCREEN
112 PRINT "RS-232 Interface Configuration"
113 PRINT
114 PRINT "baud rate:            ";Baud
115 PRINT "data bits:            ";Bits
116 PRINT "stop bits:            ";Sbits
117 PRINT "parity check enabled: ";Parity_on$
118 PRINT "parity type:           ";Parity$
119 PRINT "transmitter pacing:   ";Out_pace$
120 PRINT "receiver pacing:      ";In_pace$
121 SUBEND
```

TWO_CTLR

```
10  !HP BASIC program:  TWO_CTLR - Two controller operation
20  !-------------------------------------------------------------
30  !This program demonstrates how an external controller
40  !and HP Instrument BASIC can work together.  This program
50  !will download a BASIC program to the HP 35665A and run it two
60  !times.  After each run, two BASIC program variables will
70  !will be read from the HP 35665A and displayed.
80  !-------------------------------------------------------------
90  !
100   Scode=7                              !Select code for interface
110   Address=11                           !Address for HP 35665A
120   Hp35665a=Scode*100+Address
130 !
140   CLEAR Hp35665a
150   OUTPUT Hp35665a;"PROG:DEL:ALL"       !Scratch the program space
160 !
170   DISP "Downloading the program..."
180   ASSIGN @Prog TO Hp35665a;EOL CHR$(10) !Change EOL character
190   OUTPUT @Prog;"PROG:DEF #0";           !Send program
200   OUTPUT @Prog;"10 COM INTEGER Times_run,Test$[10]"
210   OUTPUT @Prog;"20 Times_run=Times_run +1"
220   OUTPUT @Prog;"30 IF Times_run=1 THEN Test$=""PASS"""
230   OUTPUT @Prog;"40 IF Times_run=2 THEN Test$=""FAIL"""
240   OUTPUT @Prog;"50 BEEP"
250   OUTPUT @Prog;"60 END"
260   OUTPUT @Prog;CHR$(10) END            !Terminate the data block
270 !
280 !Set up registers for interrupt on PROGRAM_RUNNING going false
290   OUTPUT Hp35665a;"*CLS"               !Clear the STATUS register
291 !Program NTR reg and OPERATION ENABLE reg for PROGRAM_RUNNING bit
300   OUTPUT Hp35665a;"STAT:OPER:NTR 16384"
310   OUTPUT Hp35665a;"STAT:OPER:ENAB 16384"
320   OUTPUT Hp35665a;"*SRE 128"     !Allow SRQ on bit 7 of STATUS reg
330 !
340   DISP "Running the program..."
350   OUTPUT Hp35665a;"PROG:STAT RUN"  !Run Program
360   Display_res(Hp35665a,Scode)      !Read and display variables
370   OUTPUT Hp35665a;"PROG:STAT RUN"  !Run Program  again
380   Display_res(Hp35665a,Scode)      !Read and display variables
390 !
400   END                              !End of this program
410 !
420   SUB Display_res(Hp35665a,Scode)
430 ! This subprogram waits for an SRQ interrupt to signal that a
440 ! BASIC program has finished.  It then clears the HP-IB registers
450 ! by reading them.  Once that is done, the values of two IBASIC
460 ! variables are read and displayed.
470 !
480     ON INTR Scode GOTO Read_results     !Set up interrupt branching
490     ENABLE INTR Scode;2                 !Allow interrupt on SRQ
```

```
500 Idle:GOTO Idle
510     !
520 Read_results:                              !Program has finished
530     A=SPOLL(Hp35665a)                      !Read and clear the SRQ
540     OUTPUT Hp35665a;"STAT:OPER?" !Read and clear OPERATION STATUS reg.
550     ENTER Hp35665a;Event
560     WAIT .5
570     !
580     OUTPUT Hp35665a;"FORM:DATA ASCII,3"
590     OUTPUT Hp35665a;"PROG:NUMB? 'Times_run'"!Read the first variable
600     ENTER Hp35665a;Times_run
610     !
620     OUTPUT Hp35665a;"PROG:STR? 'Test$'"     !Read the second variable
630     ENTER Hp35665a;Test$
640     !
650     PRINT "Times_run: ";Times_run,"Test$: ";Test$
660   SUBEND
```

WAI_SYNC

```
10  ! HP Instrument BASIC program: WAI_SYNC - Measurement synchronization
11  ! ------------------------------------------------------------------
12  ! This program demonstrates how to use the *WAI command to
13  ! prevent execution of an HP-IB command until all previous
14  ! commands have finished.  In this example, the trace display
15  ! measurement has finished.
16  ! The *WAI command does not affect program operation. The
17  ! program will run to completion, sending all of the commands to
18  ! to the HP35665A without waiting for them to be executed.
19  ! ------------------------------------------------------------------
100  Scode=8                              !Interface select code
110  Address=0
120  Hp35665a=Scode*100+Address
160  !
170  DISP "Sending HP-IB commands..."
171  OUTPUT Hp35665a;"SYST:PRES"
172  OUTPUT Hp35665a;"AVER:COUN 1"
174  OUTPUT Hp35665a;"AVER ON"
190  OUTPUT Hp35665a;"FREQ:SPAN 50 HZ"!Set narrow span
200  OUTPUT Hp35665a;"ABORT; INIT"    !Start the measurement
210  OUTPUT Hp35665a;"*WAI"           !Tell analyzer to wait here until
220                                   !all HP-IB commands have finished
230  OUTPUT Hp35665a;"DISP:FORM ULOW" !Go to upper/lower after waiting
240  BEEP
250  DISP "Finished. Display will go to UPPER/LOWER when meas. done"
260  END
```

11

Instrument-Specific HP Instrument BASIC Features

Introduction

The *HP Instrument BASIC Users Handbook* that accompanies this manual is divided into the following sections:

- "HP Instrument BASIC Programming Techniques"
- "HP Instrument BASIC Interfacing Techniques"
- "HP Instrument BASIC Language Reference"

The *HP Instrument BASIC Users Handbook* is included with all Hewlett-Packard instruments that use HP Instrument BASIC. Since each instrument is different, the way that HP Instrument BASIC interfaces and interacts with its host often changes from one instrument to another.

For example, some instruments employ editors, while others do not, and front panel interfaces often vary a great deal from one instrument to another. For this reason, many parts of the *HP Instrument BASIC Users Handbook* are either generic in nature, or apply to only one of many possible instrument interfaces.

This chapter describes how to use the *HP Instrument BASIC Users Handbook* for the HP 35665A. Global exceptions apply throughout the handbook. These differences are discussed by category. Specific differences for each command are listed in table 11-4.

Global Exceptions

Global Exceptions can be categorized as follows:

- HP-IB, GPIO and RS-232-C interfaces
- Display and keyboard interfaces
- Disk I/O
- Miscellaneous command differences

Each of these categories is explained in detail in the following sections.

HP-IB, GPIO and RS-232-C Interfaces

The *HP Instrument BASIC Users Handbook* refers to various interface types, particularly in chapter 2 of "HP Instrument BASIC Interfacing Techniques." HP Instrument BASIC in the HP 35665A supports the HP-IB interface and the RS-232-C interface. It does not support the GPIO interface.

In addition, HP Instrument BASIC in the HP 35665A supports only three interface select codes: 7 for the external bus, 8 for the internal bus, and 9 for the RS-232-C serial port. This affects the following commands:

- ABORT
- ENABLE INTR
- DISABLE INTR
- ON INTR
- OFF INTR
- ON TIMEOUT
- OFF TIMEOUT

Select code 8, the internal HP-IB interface, is not valid with the LOCAL LOCKOUT and REMOTE commands. Only select code 7, the external HP-IB interface may be used with these commands.

Select code 9, the serial RS-232-C interface is not valid with the CONTROL and STATUS commands. HP Instrument BASIC does not support these commands for select code 9.

When using the statements ENABLE INTR and ON INTR, the line containing the ON INTR statement must always precede the line containing ENABLE INTR.

The select codes for GPIO (select code 12) is not valid in the HP 35665A. The internal default select code for the display (select code 1) and for the keyboard (select code 2) are valid. The display function returns 1. "ENTER CRT" is not supported. The KBD function returns 2. Select code 2 may be used with the ENTER statement, as in ENTER 2, which allows text to be entered from the keyboard.

Display and Keyboard Interfaces

The following section describes the differences between the standard interface assumed by the *HP Instrument BASIC Users Handbook* and the HP 35665A's display and keyboard interface. In addition, the HP 35665A's keys that emulate command line execution of HP Instrument BASIC keywords are listed.

Display Differences

Most references to the display (CRT) in the *HP Instrument BASIC Users Handbook* assume a standard 80 column terminal. The HP 35665A has a 58 column display for text. This affects references to the width of the default PRINTER IS device (the display) in the LIST, PRINT and PRINTER IS commands.

You must allocate a display partition to view output to the display because the instrument shares the display with HP Instrument BASIC. This affects both the text commands listed above, as well as the graphics commands, MOVE and DRAW. Three different display partitions, UPPER, LOWER or FULL, may be allocated. The text width for all three is the same. The only change for the text commands is how much text is displayed at one time.

For the MOVE and DRAW graphics commands, the maximum x and y coordinates are dependent on the current partition. The minimum coordinates are always 0,0. See chapter 8, "Graphics and Display Techniques," for more information on using the MOVE and DRAW commands with the display partitions.

The PEN command is implemented in the HP 35665A with two parameters, 0 and 1 (or any non-zero number). PEN 1 causes any following DRAW commands to draw a line on the display. PEN 0 erases all line segments that any following DRAW commands encounter. PEN 0 does *not* perform a pixel complement function as it does in HP BASIC.

Keyboard Differences

The *HP Instrument BASIC Users Handbook* assumes the use of a standard HP BASIC Series 200 workstation keyboard. It also assumes that HP Instrument BASIC works in "command line execution mode," where individual commands may be entered and executed from the keyboard.

HP Instrument BASIC in the HP 35665A works with an HP approved PC keyboard and does *not* use command line execution mode. When using the HP Instrument BASIC editor, the HP 35665A's front-panel hardkeys become alpha keys. Softkey menus emulate many of the keywords that are executable from the command line on an HP BASIC workstation (such as RUN, CONTINUE, and SCRATCH).

The following keypath of HP 35665A hardkeys and softkeys correspond to HP Instrument BASIC keywords:

CONTINUE

[BASIC] [CONTINUE]

[BASIC] [DEBUG] [CONTINUE]

DEL

[BASIC] [EDIT] [DELETE LINE]

EDIT

[BASIC] [EDIT]

LIST

[BASIC] [PRINT PROGRAM]

PAUSE

[BASIC]

RUN

[BASIC] [RUN PROGRAM1] . . . [RUN PROGRAM5]

[BASIC] [INSTRUMNT BASIC] [RUN PROGRAM]

[BASIC] [DEBUG] [RUN]

SCRATCH

[BASIC] [UTILITIES] [SCRATCH]

SECURE

[BASIC] [UTILITIES] [SECURE]

STOP

[Local/HP-IB]

REN

[BASIC] [UTILITIES] [RENUMBER]

The following HP Instrument BASIC keywords are described in the "Language Reference" section of the *HP Instrument BASIC Users Handbook* in terms of a standard workstation keyboard. Their use with the HP 35665A is described below:

EDIT

Ignore all documentation in the "HP Instrument BASIC Language Reference" for the EDIT command. See chapter 5, "Developing Programs," for information on using the HP Instrument BASIC editor in the HP 35665A.

ON KEY and OFF KEY

10 softkeys are available for use in the HP 35665A. They appear on the right side of the display where instrument softkeys normally appear. Key selector values range from 0 through 9. In addition, the softkeys are loaded into the function keys ([*F1*] - [*F10*]) with the keyboard.

INPUT

When an INPUT statement is encountered in an HP Instrument BASIC program, the alpha entry menu appears on the display. You can use your keyboard, or the front panel alpha keys, the numeric keypad and the symbol softkeys to enter a response.

To enter an input response press the [ENTER] softkey in the alpha entry menu or the [*Enter*] key on the keyboard. Disregard all keys mentioned in the "HP Instrument BASIC Language Reference" description of this key.

You have two options for terminating an INPUT command:

Press [ENTER] (either the softkey or the [*Enter*] key on the keyboard)
Press the [PAUSE] softkey ([*F10*] on the keyboard).

To continue the program after pressing the [PAUSE] softkey, press the [CONTINUE] softkey in the **[BASIC]** menu or the [SINGLE STEP] softkey in the [DEBUG] menu. The INPUT statement is re-executed.

You *cannot* press the **[BASIC]** hardkey to pause the program *nor* the **[Local/HP-IB]** hardkey to stop the program because these are redefined as alpha keys whenever the alpha entry menu appears.

To enter an input response while a program is under remote control—that is, an external controller is executing the program—the program *must be returned* to local (front panel) control. Press the **[Local/HP-IB]** hardkey to return the program to local control. Enter the input response as described above. The instrument remains in local control after terminating the input command. Pressing the **[Local/HP-IB]** hardkey again, resets the program. It is recommended that you specify the exact key sequence expected in your input prompt.

ENTER

For a description of using ENTER with a keyboard (ENTER KBD) see the previous description for the INPUT statement.

Note

In the "HP Instrument BASIC Language Reference" every command contains a line stating whether or not the command is "keyboard executable." Disregard this information for HP Instrument BASIC in the HP 35665A. The keyboard for the HP 35665A does not operate in this manner.

Disk I/O

The following section specifies the HP 35665A's implementation of disk I/O functions.

Disk I/O Commands

Many HP Instrument BASIC commands that pertain to the disk I/O (SAVE, RE-SAVE, COPY, MSI, etc.) have similar functions executed by normal HP 35665A front-panel operations. *These front-panel operations are not considered to be HP Instrument BASIC functions.*

For example, the MASS STORAGE IS command when executed in a program is totally independent of the current storage device found under the [Save/Recall] [DEFAULT DISK] key. Conversely, using the [DEFAULT DISK] key to change the default storage device has no impact on any MSI statements within an HP Instrument BASIC program.

Volume Specifiers

HP Instrument BASIC in the HP 35665A supports four mass storage devices; the internal disk drive, volatile RAM disk (memory unit 0) and non-volatile RAM disk (memory unit 1) and an external disk drive (Hewlett-Packard Subset/80). This affects the volume specifier parameter in the following commands:

- ASSIGN
- CAT
- COPY
- CREATE
- CREATE ASCII
- CREATE BDAT
- CREATE DIR
- GET
- INITIALIZE
- MASS STORAGE IS
- PRINTER IS
- PURGE
- RENAME
- RE-SAVE
- SAVE

Valid volume specifiers for each mass storage device are shown in table 11-1.

Table 11-1. Mass Storage Volume Specifiers

Disk	Volume Specifier	MSI Specifier
EXTERNAL	:EXTERNAL,7XX,uu	EXT:
	:EXTERNAL,7XX	
	:EXTERNAL	
	:,7XX	
INTERNAL	:INTERNAL,4,0	INT:
	:,4,0	
	:,4	
VOLATILE RAM	:MEMORY,0,0	RAM:
	:MEMORY,0	
	:,0	
NON-VOLATILE RAM	:MEMORY,0,1	NVRAM:
	:,0,1	

XX bus address
uu unit number

Disk Format

HP Instrument BASIC in the HP 35665A recognizes two types of disk and file formats; LIF (Logical Interface Format) and DOS (Disk Operating System). Although the HP 35665A recognizes DOS format and directories, it does not support HFS (Hierarchical File System). The *HP Instrument BASIC Users Handbook* addresses LIF, DOS, and HFS file systems. In general, disregard all references to HFS throughout the *HP Instrument BASIC Users Handbook.*

Note The HP 35665A does not support HFS.

File Types

HP Instrument BASIC can create three types of files: ASCII, BDAT, and untyped. These files can exist on either a DOS or a LIF formatted disk.

If you catalog a DOS disk, these three types show up as "ASCII," "BDAT," and "DOS." If you catalog a LIF disk, these three types show up as "ASCII," "BDAT," and "HP-UX." Table 11-2 indicates these configurations.

Table 11-2. HP Instrument BASIC File Types

File Type	Appears on a LIF disk as	Appears on a DOS disk as
ASCII	ASCII	ASCII
BDAT	BDAT	BDAT
untyped	HP-UX	DOS

HP Instrument BASIC supports LIF protect codes only on BDAT files. An error is generated if a LIF protect code is encountered on an ASCII file. HP Instrument BASIC ignores a LIF protect code on an untyped file.

A special note about file types and file systems:

The "HP Instrument BASIC Language Reference" sometimes uses the terms HP-UX file and HFS interchangeably or refers to HP-UX files only in context of HFS volumes. In fact, HP-UX files can exist on a LIF volume, which the HP 35665A supports. Be careful when reading the descriptions in the "HP Instrument BASIC Language Reference." The HP 35665A supports HP-UX files on LIF volumes only. The HP 35665A does not support HP-UX files on HFS volumes.

The ASCII file type described in this section is specific to HP's LIF file system and is *not* the same as the standard "ASCII" file type in a DOS environment. If you copy an ASCII file from a LIF disk to a DOS disk, the file appears as an "ASCII" file. However, the file is not usable with DOS-system editors. *Untyped files* are the *only files* you can edit with a DOS ASCII editor on a PC.

An untyped file is automatically generated whenever an HP Instrument BASIC program is SAVEd from the HP 35665A to a DOS-formatted disk. A RE-SAVE maintains the original file type if a file exists, otherwise it performs the same action as SAVE.

Formatting Disks

Formatting a disk prepares it for use. The HP 35665A recognizes both LIF and DOS formats and has the capability to format either type in the [FORMAT DISK] menu.

You can also use the INITIALIZE statement to format a disk.

Caution

Existing files on the media are destroyed when formatting or executing the INITIALIZE command.

The [FORMAT DISK] Menu

The [FORMAT DISK] menu under the [**Disk Utility**] hardkey allows you to define format parameters and to format a disk using these parameters. See figure 11-1.

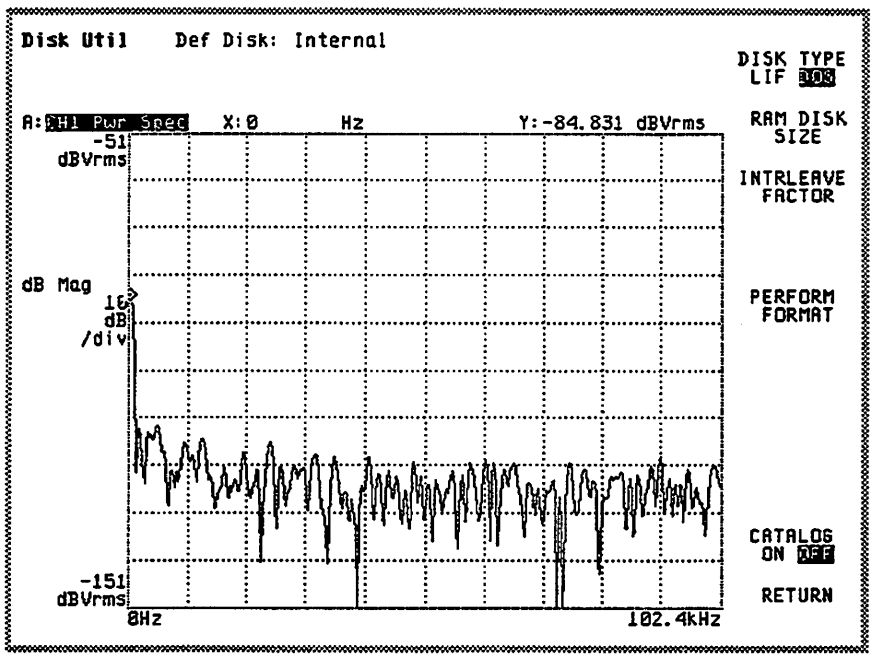

Figure 11-1. The [FORMAT DISK] Menu

The [DISK TYPE LIF **DOS**] softkey allows you to select the type of format. The default type is DOS. Press this key to toggle the selection to LIF.

The [RAM DISK SIZE] softkey specifies the storage capacity of the Volatile RAM disk. The default value is 64 KBytes of storage. Use the numeric keypad or the arrow keys to enter a new value. Values are rounded up to the next highest KByte (1024 byte) increment. The entry window appears at the top of the display when you press the softkey.

Note

You can use the [FORMAT DISK] menu to format non-volatile RAM (NVRAM), the internal disk drive or the external disk drive. However, the [RAM DISK SIZE] specification is ignored.

The [INTRLEAVE FACTOR] softkey defines the ordering of the sectors on the 3.5 inch flexible disks. To specify the default value for the disk, specify 0. This value is ignored when formatting the non-volatile RAM (NVRAM).

Press [PERFORM FORMAT] to start the format process. An entry window at the top of the screen displays the specifier for the default disk. See table 11-1 for the MSI specifiers. Use the alpha entry keys or the keyboard to modify the field. The current values for [DISK TYPE], [RAM DISK SIZE] and [INTRLEAVE FACTOR] are used.

The INITIALIZE Statement

The disk format, LIF or DOS, is specified when the media is initialized. The INITIALIZE statement takes the following form:

```
INITIALIZE <disk format>: <volume specifier> <interleave factor>,
          <format option>
```

If *<disk format>* is not specified, the default format is LIF.

The *<volume specifier>* for the volatile RAM disk includes a *<unit size>* parameter,that specifies the number of 256-byte sectors. The actual size is memory dependent and ranges from 4 thru 32767.

The *<format option>* parameter specifies the capacity of the flexible disk drive (internal or external). See table 11-3 for the valid format options.

Table 11-3. Flexible Disk Format Options

Media	Format Option	Bytes/Sector	Sectors/Track	Tracks/Surface	Maximum Capacity (bytes)
1-MByte	0	256	16	77	630,784
	1*	256	16	77	630,784
	2	512	9	77	709,632
	3	1,024	5	77	788,480
	4**	256	16	77	270,336
	16	512	9	80	737,280
2-MByte	0	256	32	77	1,261,568
	1***	256	32	77	1,261,568
	2	512	18	77	1,419,264
	3	1,024	10	77	1,576,960
	4***	256	32	77	1,261,568
	16	512	18	80	1,474,560

* Same as Format Option 0 (default) when using 1-MByte media.
** Format Option 4 (singled sided disk format) is not supported in internal disk drive (INT:).
*** Same as Format Option 0 (default) when using 2-MByte media.

Note

Table 11-3 specifies the *maximum* capacity for each format option. Actual capacity is dependent upon the file system type, LIF or DOS.

For example, to INITIALIZE a flexible disk in the internal disk drive in LIF format, use the following:

```
INITIALIZE "LIF:,4,0",1
```

To INITIALIZE a DOS disk in an external disk drive, use the following:

```
INITIALIZE "DOS:,700,0",1,16
```

In this example, the format option, 16, is important when initializing a DOS disk. An incorrect format option results in a disk that the HP 35665A can use, but other DOS systems cannot use. This potential problem can be avoided by formatting the disk on a DOS system, rather than the HP 35665A.

Caution

An incorrect format option may prevent other DOS systems from using the DOS disk.

You can use the INITIALIZE statement to format the non-volatile disk (NVRAM). However, the size of NVRAM is fixed and the interleave factor is ignored. You can only change the file format.

Once initialization is complete, file format, "LIF" or "DOS", is not specified in any other file operations. HP Instrument BASIC automatically determines the format of the disk.

Miscellaneous Command Differences

COS

The range of the COS command is all absolute values less than 1.7083127722 e + 10 degrees.

SYSTEM$

The HP 35665A does not support the topic specifier, (SYSTEM VERSION:).

Note

Since the *HP Instrument BASIC Users Handbook* is continually revised to support all implementations of HP Instrument BASIC, there may be other commands that appear in that documentation that are not supported in the HP 35665A. Table 11-4 in the following section lists all HP Instrument BASIC keywords supported by HP Instrument BASIC in the HP 35665A.

Specific Exceptions

Table 11-4 summarizes the HP Instrument BASIC keyword implementation in the HP 35665A. The table indicates if the keyword has front panel support. (If it does, the key path is given.) Table 11-4 also lists the major differences between the descriptions of these keywords in the "HP Instrument BASIC Language Reference" and the way they are implemented in the HP 35665A. Where differences are too extensive to be summarized, references to their explanation in the "Global Exceptions" section are given.

Any keywords or functions found in the "HP Instrument BASIC Language Reference" that do not appear in this table, do not apply to HP Instrument BASIC in the HP 35665A and should be ignored.

Table 11-4. HP 35665A Keyword Implementation

Command	Front Panel Support	Exceptions
ABORT	None	Interface Select Code = 7 or 8
ABS	None	None
ACS	None	None
AND	None	None
ASN	None	None
ASSIGN	None	One HP-IB device per ASSIGN statement LIF protect code supported in BDAT files only Does not support HFS volumes See "Disk I/O"
ATN	None	None
BASE	None	None
BEEP	None	None
BINAND	None	None
BINCMP	None	None
BINEOR	None	None
BINIOR	None	None
BIT	None	None
CALL	None	None
CAT	None (Independent of **[Disk Utility]** and **[Save/Recall]** functions)	Does not support HFS catalogs See "Volume Specifiers" in "Disk I/O"
CHR$	None	None
CLEAR	None	None
CLEAR SCREEN	None	None
COM	None	None

Table 11-4. HP 35665A Keyword Implementation (Continued)

Command	Front Panel Support	Exceptions
CONT	[**BASIC**] [CONTINUE] or [**BASIC**] [DEBUG] [CONTINUE]	No line number or label support
CONTROL	None	Not supported
COPY	None (Independent of [**Disk Utility**] functions)	LIF protect code in BDAT files only Does not support HFS volumes See "Volume Specifiers" in "Disk I/O"
COS	None	Absolute range values less than 1.7083127722 e+10
CREATE	None	Does not support HFS volumes See "Disk I/O"
CREATE ASCII	None	Does not support HFS volumes See "Disk I/O"
CREATE BDAT	None	LIF protect code allowed Does not support HFS volumes See "Disk I/O"
CREATE DIR	None	Does not support HFS volumes See "Volume Specifiers" in "Disk I/O"
CRT	None	ENTER CRT (ENTER 1) not supported
DATA	None	None
DEF FN	None	None
DEG	None	None
DEL	[**BASIC**] [EDIT][DELETE LINE]	Deletes only the current line
DIM	None	None
DISABLE	None	None
DISABLE INTR	None	Interface Select Code = 7 or 8
DISP	None	None
DIV	None	None
DRAW	None	Maximum x,y coordinates: Full partition (474,345) Upper partition (474,171) Lower partition (474,171)
DROUND	None	None
DVAL	None	None
DVAL$	None	None
EDIT	[**BASIC**][EDIT]	Editing functions described in chapter 5.
ENABLE	None	None

Table 11-4. HP 35665A Keyword Implementation (Continued)

Command	Front Panel Support	Exceptions
ENABLE INTR	None	Interface Select Code = 7 or 8 Must not precede an ON INTR statement.
END	None	None
ENTER	None	Select Code = 2 (keyboard)
ERRL	None	None
ERRLN	None	None
ERRM$	None	None
ERRN	None	None
EXOR	None	None
EXP	None	None
FN	None	None
FOR...NEXT	None	None
FRACT	None	None
GCLEAR	None	None
GET	None (Independent of **[Save/Recall]** functions)	Does not support HFS volumes See "Volume Specifiers" in "Disk I/O"
GOSUB	None	None
GOTO	None	None
IF...THEN	None	None
IMAGE	None	None
INITIALIZE	None (Independent of **[Disk Utility]** functions)	Does not support HFS volumes See "Disk I/O"
INPUT	None	See INPUT command in "Keyboard Differences" section
INT	None	None
INTEGER	None	None
IVAL	None	None
IVAL$	None	None
KBD	None	External Keyboard (2) or front-panel alpha keys
LEN	None	None
LET	None	None
LGT	None	None
LIST	**[BASIC]**[PRINT PROGRAM]	Default width = 58 (see PRINTER IS)

Table 11-4. HP 35665A Keyword Implementation (Continued)

Command	Front Panel Support	Exceptions
LOCAL	None	None
LOCAL LOCKOUT	None	Interface Select Code = 7 only
LOG	None	None
LOOP	None	None
LWC$	None	None
MASS STORAGE IS	None (Independent of **[Save/Recall]** and **Disk Utility]** functions)	Does not support HFS volumes See "Disk I/O" External disks must be online
MAX	None	None
MAXREAL	None	None
MIN	None	None
MINREAL	None	None
MOD	None	None
MODULO	None	None
MOVE	None	Maximum x, y coordinates: Full partition (474,345) Upper partition (474,171) Lower partition (474,171)
NOT	None	None
NUM	None	None
OFF CYCLE	None	None
OFF ERROR	None	None
OFF INTR	None	Interface Select Code = 7 or 8 Must precede ENABLE INTR statement.
OFF KEY	None	Key selectors are 0 thru 9
OFF TIMEOUT	None	Interface Select Code = 7, 8 or 9
ON CYCLE	None	None
ON ERROR	None	None
ON INTR	None	Interface Select Code = 7 or 8
ON KEY	None	Key selectors are 0 thru 9
ON TIMEOUT	None	Interface Select Code = 7, 8 or 9
OR	None	None
OUTPUT	None	Select Code = 1, 7, 8 or 9
PEN	None	0 = erase 1 = draw
PASS CONTROL	None	Interface Select Code 8 (pass control of external bus to analyzer)

Table 11-4. HP 35665A Keyword Implementation (Continued)

Command	Front Panel Support	Exceptions
PAUSE	None	None
PI	None	None
POS	None	None
PRINT	None	PRINTER IS default width = 58
PRINTER IS	None	default width = 58 LIF protect code in BDAT files only Does not support HFS volumes See "Volume Specifiers" in "Disk I/O"
PROUND	None	None
PRT	None	None
PURGE	None (Independent of **[Disk Utility]** functions)	LIF protect code in BDAT files only Does not support HFS volumes See "Volume Specifiers" in "Disk I/O"
RAD	None	None
RANDOMIZE	None	None
RANK	None	None
READ	None	None
REAL	None	None
REM	None	None
REMOTE	None	Does not support Interface Select Code 8 or 9
REN	**[BASIC]** [UTILITIES] [RENUMBER]	No line label support
RENAME	None (Independent of **[Disk Utility]** functions)	LIF protect code in BDAT files only Does not support HFS volumes See "Volume Specifiers" in "Disk I/O"
REPEAT...UNTIL	None	None
RE-SAVE	None (Independent of **[Save/Recall]** functions)	Does not support HFS volumes See "Disk I/O"
RESTORE	None	None
RETURN	None	None
RETURN . . .	None	None
REV$	None	None
RND	None	None
ROTATE	None	None
RPT$	None	None

Table 11-4. HP 35665A Keyword Implementation (Continued)

Command	Front Panel Support	Exceptions
RUN	[BASIC] [RUN] or [BASIC] [DEBUG] [RUN]	None
SAVE	None (Independent of [Save/Recall] functions)	Does not support HFS volumes See "Disk I/O"
SCRATCH	[BASIC] [UTILITIES] [SCRATCH]	Does not support HFS volumes
SECURE	[BASIC] [UTILITIES] [SECURE]	None
SELECT...CASE	None	None
SGN	None	None
SHIFT	None	None
SIN	None	None
SIZE	None	None
SPOLL	None	None
SQRT	None	None
STATUS	None	Not Supported
STOP	None	None
SUB	None	None
SYSTEM PRIORITY	None	None
SYSTEM$	None	Does not support "VERSION:"
TAN	None	None
TIMEDATE	None	None
TRIGGER	None	None
TRIM$	None	None
UPC$	None	None
VAL	None	None
VAL$	None	None
WAIT	None	None
WHILE	None	None
WILDCARDS	None	Does not support UX

Index

A

aborting I/O operations 2-6, 8-11
active controller
  defined 8-3
  HP Instrument BASIC as 8-18
active program 2-11
address
  extended 8-2
  primary 8-2
  secondary 8-2
allocating display partitions 5-25
  See also display partitions
alpha keys on the front panel 5-12
appending programs 4-9
arbitrary block data 8-27, 8-29
arrays 6-4
ASSIGN statement 2-4
* in program line 5-24
ATN signal line 8-4
AUTO_BAS program 4-11
autoloading a program 4-11

B

Back Space hardkey 5-10
baud rate 9-9
breakpoints, setting 6-5
bus
  See HP-IB
  See also RS-232-C

C

chaining programs 4-9
clearing
  input buffer 2-2
  input buffer (RS-232-C) 9-13
  memory 5-20
  screen 5-27
configuring the RS-232-C port 9-8 - 9-11
continuing a program 3-2, 6-7
control codes 7-4
controller
  active 8-3
  changing status 2-6
  See also external controller
  See also HP-IB
  non-active 8-14
  system 8-3
conventions 1-4
copying lines 5-10

D

de-allocating display partitions 5-25
  See also display partitions
DEBUG menu
  CONTINUE softkey 3-5, 6-6 - 6-7
  EXAMINE VARIABLE softkey 6-4
  LAST ERROR softkey 6-7
  RESET softkey 6-7
  RUN softkey 3-3, 6-7
  SINGLE STEP softkey 6-6
debugging programs 6-2
deleting
  characters 5-16 - 5-17
  lines 5-17
device selectors
  description 8-2
  in a recorded program 2-4
Device State Register set 9-15
directories 4-3
disabling autoload programs 4-11
disk
  catalog 4-7
  default 4-5
  format 4-2 - 4-4, 11-7
  I/O functions 11-6
display partitions
  allocating 7-1
  commands that use 7-1
  de-allocating 7-2
  pixel limits 7-5
  size of 7-3, 7-5
  writing graphics to 7-5
  writing text to 7-3
DISPLAY SETUP menu, use during program development 5-27, 7-1
downloading programs 8-27
  See also example programs
DRAW statement 7-5

E

echoing HP-IB commands 2-10
EDIT menu
  DELETE CHARACTER softkey 5-17
  DELETE LINE softkey 5-17
  DELETE TO LINE END softkey 5-17
  END EDIT softkey 5-6
  ENTER softkey 5-8 - 5-9
  GOTO LINE softkey 5-9
  INSERT LINE softkey 5-11
  INSERT SPACE softkey 5-10
  RECALL LINE softkey 5-11, 5-17
  UPPERCASE lowercase softkey 5-12
editing
  EDIT command 11-5
  with the EDIT softkeys 5-8
  with the keyboard 5-4
ENABLE RECORDING softkey 2-1
ENTER softkey 5-9
ENTER statement 8-3, 8-5, 9-13, 11-5
  See also example programs
entering lines 5-10
error messages
  out of memory 5-19
  viewing 4-8
examining variables
  arrays 6-4
  by name 6-4
  strings 6-4
example programs
  appending files 4-9
  arbitrary source 10-2 - 10-6
  custom softkeys 10-26 - 10-28
  display RS-232-C configuration 9-6
  displaying graphics and text 7-6
  downloading a program 10-29 - 10-30
  downloading data 8-28
  group delay 10-7 - 10-13
  marker functions 10-7 - 10-13
  RS-232-C configuration 10-26 - 10-28
  SRQ interrupt 10-16
  synchronization 10-17, 10-31
  transferring active control 8-23
  uploading data 8-29
  use of *OPC 10-16
  use of *OPC? 10-17
  use of *WAI command 10-31
  use of data registers 10-2 - 10-6, 10-18 - 10-25
  use of ENTER 10-17
  use of external controller 10-29 - 10-30
  use of status registers 10-14 - 10-15
  waveform math functions 10-18 - 10-25
extended addressing 8-2
external controller
  configuring the RS-232-C port 8-22
  See also example programs
  input under remote control 11-5
  querying HP Instrument BASIC variables 8-26
  setting HP Instrument BASIC variables 8-26
  transferring data 8-23

F

file
  appending 4-9
  backup 4-7
  closing 3-5
  systems 11-7
  transferring 4-2 - 4-3
  See also transferring data
  translating 4-2
  types 4-2, 11-8
file names
  DOS 4-3
  LIF 4-4
format options 11-11
formatting disks 11-9

G

GCLEAR statement 7-5
GET statement 4-1, 4-9
GPIB Echo 2-10
graphics
  commands 7-5
  in partitions 7-5

H

HP Instrument BASIC
  applications 1-2
  as active controller 8-18
  as non-active controller 8-21
  editor 5-3 - 5-17
  HP-IB model 8-15 - 8-21
  online help 1-1
  resetting 3-6
  RS-232-C interface 9-2 - 9-7
  using with external controllers 8-22 - 8-30
  vs. HP BASIC 1-2
HP Instrument BASIC Users Handbook
  disk I/O exceptions 11-6
  display exceptions 11-3
  exceptions by keyword 11-14 - 11-19
  I/O exceptions 11-2
  keyboard differences 11-3
  keyword exceptions 11-5, 11-13

HP-IB
ABORT statement 8-11
active controller 8-3
ATN 8-4
buffer 2-2
bus 8-1
CLEAR statement 8-10
commands 2-4, 8-6
controlling the bus 8-1
device selectors 8-2
DISP:PROG command 7-2
example bus sequences 8-5
See also example programs
extended commands 8-22
external port 8-15
general structure 8-3
internal port 8-15
interrupts 8-12
listener 8-4
LOCAL LOCKOUT statement 8-8
LOCAL statement 8-9
managing the bus 8-6 - 8-14
PASS CONTROL statement 8-14
primary address 8-2
REMOTE statement 8-7
SCPI commands 2-4
secondary address 8-2
select code 8-2
service routines 8-11
SPOLL statement 8-14
statement summary 8-6
status registers 8-17
system controller 8-3
talker 8-4
TRIGGER statement 8-10
unlisten 8-4
HP-IB Interface
HP Instrument BASIC to HP 35665A 2-4
HP-Instrument BASIC
See also RS-232-C

I

INITIALIZE
command 11-11
statement 11-9
input buffer 2-2
input buffer (RS-232-C) 9-13
INPUT statement 11-5
inserting
keywords 5-15
lines 5-11
measurement sequence 5-16
spaces 5-10
symbols 5-14
INSTRUMNT BASIC menu
CONTINUE softkey 3-5
DEBUG softkey 6-2
See also EDIT menu
EDIT softkey 5-3
ENABLE RECORDING softkey 2-1
PRINT PROGRAM softkey 5-25
RUN softkey 3-3
UTILITIES softkey 5-18
interrupts
servicing SRQ 8-13

K

keyboard
deleting characters 5-16
installing 5-6
keys 5-4
using the editor 5-4
keystroke recording
See recording
keywords
HP Instrument BASIC 11-14 - 11-19
inserting 5-15
knob, using 5-9

L

lines
copying 5-10
deleting 5-17
inserting 5-11
moving 5-10
renumbering 5-10, 5-22
loading a program 4-8

M

managing the bus
See HP-IB
mass storage
devices 4-5
volume specifiers 11-6
memory 4-5
available 5-19
clearing 5-20
program buffer 2-11 - 2-12
sizing 5-19
stack size 4-8
MOVE statement 7-5
moving lines 5-10 - 5-11

N

naming files
See file names
non-active controller
defined 8-14
HP Instrument BASIC as 8-21

O

OFF KEY command 11-5
ON INTR statement 8-12
ON KEY command 11-5
ON TIMEOUT statement 9-16
OUTPUT statement 2-3, 7-3, 8-3, 8-5, 9-12

P

parity (RS-232-C) 9-9
passing control
to the controller 8-14
to the instrument 2-8, 8-19
PAUSE statement 3-4
pausing a program 3-5
PEN statement 7-5
pixel coordinates
See display partitions
Port 1
See RS-232-C
prerun operation 3-2
Preset hardkey 2-9
primary address 8-2
program
appending 4-9
AUTO_BAS 4-11
autoloading 4-11
buffers 2-11 - 2-12, 4-5
chaining 4-9
continuing 3-2, 6-7
listing 5-25
loading 4-8
pausing 3-5
printing 5-25
recalling 4-8
resetting 6-7
running 3-2
saving to disk 4-7
securing 5-23
selecting 2-11
stopping 3-6
transferring 4-1

R

RAM unit size 11-11
RE-SAVE
command 11-8
statement 4-1
RECALL PROGRAM softkey 4-8
recalling
lines 5-11
programs 4-8
recording
avoiding errors 2-9
default states 2-7
generated program statements 2-3 - 2-4
how it works 2-3, 2-5
how to 2-1
into an existing program 5-16
operations not recorded 2-6 - 2-8
save and recall operations 2-7
remote control
with HP-IB commands 8-22
removing text 5-16
renumbering
lines 5-3, 5-10
programs 5-22
resetting a program 6-7
RS-232-C
access 9-2
baud rate 9-9, 9-18
character format 9-4, 9-8, 9-23
character length 9-19
clearing the buffer 9-13
configuring 9-8 - 9-11
detecting errors 9-14 - 9-15
Device State Register 9-15
See also example programs
flow control 9-10
handshake (receiver) 9-20
handshake (transmitter) 9-24
handshaking 9-10
input 9-13
lines 9-3
output 9-12
pacing 9-20, 9-24
parity 9-9, 9-21 - 9-22
pin designators 9-7
protocols 9-10
timeouts 9-16
transferring data 9-12 - 9-16
XON/XOFF 9-10

running a program
at turn on 4-11
from BASIC menu 3-2
from DEBUG menu 6-7
from external controller 8-22
from INSTRUMNT BASIC menu 3-3

S

sample programs
See example programs
SAVE
command 11-8
statement 4-1
Save/Recall menu
RE-SAVE PROGRAM softkey 4-7
SAVE PROGRAM softkey 4-7
saving a program 4-7
SCPI compliance 2-4
screen
clearing 5-27
See also display partitions
secondary address 8-2
securing programs 5-23
selecting
active programs 2-11
devices 2-4, 8-2
serial port
See RS-232-C
service requests, HP-IB 8-11
sizing memory 5-19
softkey labels, changing 2-12
SRI, Service Request Indicators 8-16
SRQ interrupts
defined 8-12
See also example programs
generating 8-16
servicing 8-13
stack size 4-8, 5-19
* in program line 5-24
status registers
Device State 9-15
See also example programs
See also HP-IB
overview 8-17
See also RS-232-C
stopping a program 3-6
storage devices 11-6
storing a program 4-7
string variables 6-4
symbols, entering 5-14
synchronizing
See also example programs
measurement events 2-8
program with instrument 7-2
system controller
defined 8-3
HP Instrument BASIC as a 1-2
System Utility menu
MEMORY USAGE softkey 5-19

T

transferring
data 4-2, 8-23
data over RS-232-C 9-12 - 9-16
data with a PC 4-3
data with an HP BASIC computer 4-4
See also example programs
programs 8-27
translating file types 4-2
TYPING UTILITIES menu
INSERT ~ %'!?'_ softkey 5-14
INSERT KEYWORD softkey 5-15

U

uploading programs 8-27
UTILITIES menu
AUTO MEMORY softkey 5-19
MEMORY SIZE softkey 5-19
RENUMBER softkey 5-22
SCRATCH softkey 5-20
SECURE softkey 5-23

X

XON/XOFF 9-10