Using HP Instrument BASIC with the HP 3589A

(Includes HP 35689A/B)

HEWLETT
PACKARD

HP Part Number 03589-90009
Microfiche Part Number 03589-90209
Printed in U.S.A.

Print Date: July 1991

8600 Soper Hill Road, Everett, WA 98205-1298

HEWLETT
PACKARD

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

Table of Contents

Chapter 1: Introduction

Chapter 2: Installing HP Instrument BASIC

Chapter 3: Recording Programs

Chapter 4: Running, Pausing and Stopping Programs

Chapter 5: Saving and Recalling Programs

Chapter 6: Developing Programs

Chapter 7: Debugging Programs

Chapter 8: Graphics and Display Techniques

Chapter 9: Interfacing with the HP-IB

Chapter 10: Example Programs

Chapter 1

Introduction

1

Introduction

Welcome

This manual will help you learn about setting up and using your HP Instrument BASIC software on the HP 3589A. It guides you through the installation process and then 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 certain instrument-specific HP Instrument BASIC features are implemented in the HP 3589A.

If HP Instrument BASIC is already installed in your HP 3589A and you would like to perform a few exercises first to get acquainted with the language and environment, the *HP 3589A Getting Started Guide* contains a section demonstrating keystroke recording, editing, saving and recalling HP Instrument BASIC programs. Working through these example tasks will give you a good overview of the HP Instrument BASIC environment in the HP 3589A.

As an additional aid to learning to use HP Instrument BASIC in the HP 3589A, be sure to try the on-line help text available for key-specific information on HP Instrument BASIC features. This help text is accessed in the same manner as it for other features of the HP 3589A, by pressing the [**Help**] hardkey followed by the hardkey or softkey you want help on.

Overview of HP Instrument BASIC

When installed in your HP 3589A, 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. In recording mode, 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 immediately put your program to work controlling and automating your HP 3589A analyzer.

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

The HP Instrument BASIC command set is similar to the command set of HP Series 200/300 BASIC. In fact, HP Instrument BASIC programs can be run on any HP BASIC workstation with few, if any, changes. Refer to chapter 9, "Interfacing with the HP-IB" for information on interfacing the HP Series 200/300 BASIC and HP Instrument BASIC environments. Porting information can be found in the HP Instrument BASIC Language Reference section of the *HP Instrument BASIC Users Handbook.*

How to Use This Manual

If you want to use HP Instrument BASIC with a minimum of editing and programming, you will find all the information you need to record, run, save, and recall programs in chapters 1 through 5. This information is generally adequate for those who only require HP Instrument BASIC to autosequence their measurement tasks.

To learn to use the front panel interface (or the optional keyboard) to edit and debug programs, read chapters 6, "Developing Programs," and 7, "Debugging Programs."

To understand how HP Instrument BASIC's graphics features apply to the HP 3589A, refer to chapter 8 "Display and Graphics Techniques."

To understand how the HP Instrument BASIC controller in the HP 3589A interfaces with external devices (such as plotters) and external controllers (such as HP Series 200/300 controllers), read chapter 9, "Interfacing with the HP-IB."

Application programs written to run in HP Instrument BASIC on the HP 3589A are listed in chapter 10.

Finally, chapter 11 helps to couple this manual with the three sections of the *HP Instrument BASIC Users Handbook*: HP Instrument BASIC Programming Techniques, HP Instrument BASIC Interfacing Techniques and HP Instrument BASIC Language Reference. The handbook is designed to serve users of HP Instrument BASIC on all instrument platforms. Chapter 11 makes clear which parts of these manuals do not apply to the HP 3589A.

Typographical Conventions

The following conventions are used in this manual when referring to various parts of the HP Instrument BASIC and HP 3589A operating environments:

[**Hardkey**]	Brackets [ ] surrounding a bold-faced name indicate the name of a hardkey on the front panel of the HP 3589A.
[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.
Bold	A bold typeface is used to emphasize a particular word or phrase.
[*Key*]	Brackets [ ] surrounding an italic typeface indicate the name of a key on the keyboard that you can use to edit HP Instrument BASIC programs.
Italic	Italic typeface is used when referring to the name of a different manual.
<element>	Angle brackets are used to signify a syntax element in a statement.

Chapter 2

Installing HP Instrument BASIC

Installing HP Instrument BASIC

Installing HP Instrument BASIC in your HP 3589A is an easy operation that normally only needs to be performed once to initialize the internal code in the instrument. Once initialized, HP Instrument BASIC is always available at power-up by pressing the [**BASIC**] hardkey.

Since the initialization procedure alters internal non-volatile memory, you will need to repeat the installation if you replace the A81 CPU assembly.

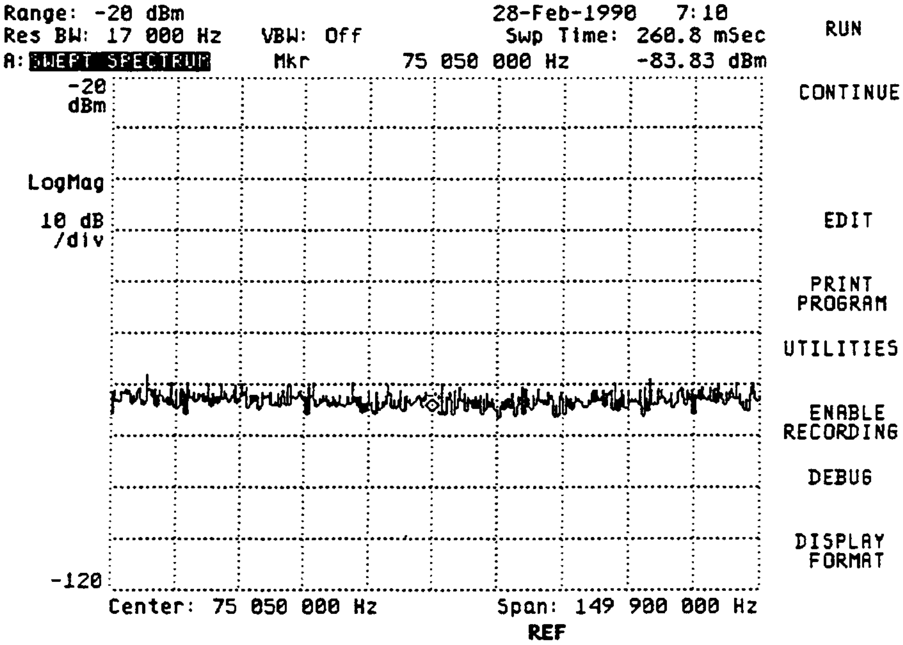
Installation Steps

1. First, insert the disk labeled "HP Instrument BASIC Installation Disk/Programming Examples" into the internal disk drive. Depending on whether your HP 3589A is powered on or off, proceed with either step 1.a or 1.b.

   a. If your instrument is already powered on, press the [**BASIC**] hardkey. The HP 3589A will automatically load the installation program from the disk and enable HP Instrument BASIC.

   b. If your instrument is powered off, turn the power on. HP Instrument BASIC will automatically be installed as the instrument powers up.

2. Press the [**BASIC**] hardkey and you should see the HP Instrument BASIC menu as shown in figure 2-1.



```
Range: -20 dBm                              28-Feb-1990    7:10
Res BW: 17 000 Hz    VBW: Off                 Swp Time: 260.8 mSec
A: SWEPT SPECTRUM    Mkr        75 050 000 Hz        -83.83 dBm
Center: 75 050 000 Hz                   Span: 149 900 000 Hz
```

Figure 2-1. HP Instrument BASIC menu

If for some reason you do not see this menu when you press [**BASIC**], try the installation again and make sure the proper disk is inserted in the drive. If you are still unsuccessful installing HP Instrument BASIC, proceed to the next section.

Troubleshooting Installation Problems

Since very little can go wrong with the installation of HP Instrument BASIC, the steps to troubleshooting are fairly simple. The problem may be with the disk, the disk drive or the non-volatile memory. Walk through the following procedures to verify which of these components appear to be functional and note your results. Then, if you still cannot install HP Instrument BASIC, call your local authorized Hewlett-Packard Service Center and describe the problem and troubleshooting steps you have performed.

1. First, check to see that the disk in the internal drive is labeled "HP Instrument BASIC Installation Disk/Programming Examples"
2. To see if the disk has been corrupted, catalog the contents using the [**Save/Recall**] keys, press :

   [**Save/Recall**]
   [DEFAULT DISK]
   [INTERNAL DISK]
   [CATALOG **ON** OFF]
3. Note whether or not you can catalog the disk.

   a. If you cannot catalog the installation disk and you have a known good LIF-formatted double-sided disk available, insert it in the drive and repeat the steps to catalog the disk. If you still cannot catalog, you may have a defective disk drive.

   b. If you can catalog a known good disk but not the installation disk, the installation disk has probably been corrupted and you will need a replacement.

   c. If you can catalog the disk, proceed to the next step to check the non-volatile memory.
4. To check the non-volatile memory, press the [**Spcl Fctn**] hardkey followed by the [SERIAL NUMBER] softkey. If a proper serial number is displayed, the non-volatile memory is functional, since the serial number is stored there.

Chapter 3

Recording Programs

3

Recording Programs

Keystroke Recording

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

What is Keystroke Recording?

Keystroke recording is a way to automatically create HP Instrument BASIC measurement sequence programs by simply enabling recording ([**BASIC**] [ENABLE RECORDING] softkey) and pressing the normal key sequences of a measurement on the analyzer. Press the [**BASIC**] hardkey to terminate the recording. The resultant program can then be run by pressing the [RUN] softkey in the [**BASIC**] menu.

HP Instrument BASIC programs communicate with the HP 3589A over an internal bus, using the same set of commands used by external controllers for remote operation of the instrument. Keystroke recording works by finding the bus command, called a SCPI mnemonic, that fits each operation you perform from the front panel and then building a program line that duplicates that operation when executed.

All program lines built by keystroke recording are entered into the analyzer's program buffer. If the program buffer contains no existing program code, a complete executable program will be inserted. If there are existing program statements in the buffer when recording is turned on, the recorded statements are simply inserted into the existing program. Refer to chapter 6, "Developing Programs," for a description of how to record into existing programs.

HP Instrument BASIC Programs and the HP-IB Buffer

Recorded programs work by sending HP-IB commands to the instrument. The commands are queued into the instrument's input buffer. An HP Instrument BASIC program generally outputs the commands much faster than the instrument can execute them. This often causes the program to complete while the instrument is still executing the commands in the input buffer. The instrument then continues to process these commands until the buffer is empty.

This may have some side-effects if you are not aware of this interaction. For example, it may not be immediately obvious that the program has actually finished, since the instrument is still functioning "remotely." This could cause confusion if you try to pause and continue a program that has actually completed.

You can clear the buffer from within your program by inserting the statement "CLEAR 8" at the beginning of your program (see chapter 6 for information on editing programs).

What's in a Recorded Program

If you look at any program created using recording, you will find that it is composed of three fundamental HP Instrument BASIC statements: ASSIGN, OUTPUT, and END. The following simple program demonstrates these statements:

```
1 ASSIGN @Hp3589a TO 800
2 OUTPUT @Hp3589a;"INST:SEL FFT"
10 END
```

There will only be one ASSIGN statement at the beginning of a program and one END statement at the end, but in a typical program there will be many OUTPUT statements. Since the OUTPUT statement does the actual work of controlling the HP 3589A, let's take a closer look at how it is used.

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 instrument.

Although the OUTPUT command is very flexible it is used only one way when generated by a recording. The following represents a typical OUTPUT command from a recording session:

```
OUTPUT @Hp3589a;"INST:SEL FFT"
```

Notice that the OUTPUT command is followed by a name representing a device selector (@Hp3589a), followed by a semicolon and the data ("INST:SEL FFT").

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 "@Hp3589a." The following line appears in all recorded programs before any OUTPUT statements:

```
ASSIGN @Hp3589a TO 800
```

The ASSIGN statement allows you to substitute an I/O path name (a variable preceded by the @ symbol) for a device selector number. Therefore, after the above ASSIGN statement, the program line

```
OUTPUT @Hp3589a;"INST:SEL FFT"
```

is equivalent to

```
OUTPUT 800;"INST:SEL FFT"
```

The device selector 800 specifies the host instrument as the destination of any data sent by the OUTPUT command. The program communicates with the HP 3589A via select code 8, the internal HP-IB interface, which is used solely for communication between HP Instrument BASIC programs and the analyzer. The analyzer will respond to any address on the internal interface from 800 to 899 (800 is typically used).

SCPI Mnemonics

The data sent to the HP 3589A by the OUTPUT command is called a SCPI mnemonic and is found in quotes following the device selector path name and semicolon:

```
2 OUTPUT @Hp3589a;"INST:SEL FFT"
```

The SCPI mnemonic codes used in HP Instrument BASIC are the same ones used to control the instrument remotely via an external computer. External computers communicate with the HP 3589A over the external HP-IB bus while HP Instrument BASIC programs communicate with it over the internal bus. In our example, the mnemonic "INST:SEL FFT" tells the instrument to set the measurement type to narrow band zoom mode.

For more information on HP-IB interfacing using HP Instrument BASIC, refer to chapter 9, "Interfacing with the HP-IB." The SCPI mnemonics for the HP 3589A are documented in the *HP 3589A HP-IB Programmer's Reference.*

How Recording Works

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

Note

SCPI mnemonics 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.

The fact that the generated SCPI mnemonics do not exactly correspond to the keys actually pressed is important to remember. As you press a sequence of keys to perform an operation, the corresponding SCPI mnemonic for that **operation** is generated. The operation may take one keystroke or several, but the mnemonic is not generated until after a valid sequence of keystrokes is completed.

In other words, it is the functional operation of the instrument that is recorded as a mnemonic, not the keystrokes that it takes to perform that operation.

For example, recording the simple key sequence: [**Meas Type**], [NARROW BAND ZOOM] requires two keystrokes and produces only one mnemonic, "INST:SEL FFT," which is generated **after** the sequence is completed. This is then automatically formatted into the command:

```
    OUTPUT @Hp3589a;"INST:SEL FFT"
```

and inserted into the program.

This means that if you accidentally press the wrong key in a sequence, it may not show up in the recorded program. It also means that you cannot exactly mimic keystrokes to leave the instrument in a specific front-panel state, unless it is a state that appears as a natural consequence of a completed operation.

For example, in the above example, pressing [**Meas Type**] in a recording session has the effect of bringing up the [**Meas Type**] menu but does not, by itself, generate a program line. You could not therefore leave the instrument with the [**Meas Type**] menu displayed.

Operations That Do Not Record

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

- Front panel operations with no corresponding SCPI mnemonic (such as help text operations, HP-IB controller status, and transitional key sequences).

- HP Instrument BASIC front panel operations; that is, any front panel operation added to the instrument by installing HP Instrument BASIC.

- Operations requiring additional programming steps, such as passing control to the instrument for plotting or special handling of armed measurement operations.

- HP-IB operations with no front panel equivalence (for example, HP-IB query commands).

Front Panel Operations Without Mnemonics

There are some areas of front panel operation for which corresponding SCPI mnemonics do not exist.

The help sequences available through the [**Help**] hardkey have no corresponding mnemonics and cannot be performed via HP-IB. They cannot, therefore, be keystroke recorded.

The controller status of the instrument cannot be changed remotely. This has two significant consequences: First, you cannot remotely change the state of the HP-IB interface (i.e., from Addressable Only to System Controller); second, when the instrument has active control of the interface for printing, plotting or external disk I/O, you cannot remotely abort that I/O operation. Therefore, any front-panel sequences that perform these operations do not generate an SCPI mnemonic and cannot be keystroke recorded.

Most operations on the front panel that require numeric entry allow you to use the knob to increment or decrement the current value. This will not record as a program line. You must always use the numeric key pad to enter any value if you want the operation to be recorded.

During a measurement sequence it may take several key presses to cause an operation that will generate a mnemonic. The transitional sequences between actual instrument events are not recordable.

Any default states you encounter while recording (and consequently do not select) are not recorded.

Note Instrument states that are not specifically selected or changed are not recorded.

Since these default states are not recorded, you must either actively select them to generate a program statement or make sure the instrument is in the same exact state when the program is run as it was when it was recorded. This is discussed further in the next section, "Avoiding Recording Errors."

HP Instrument BASIC Operations

Softkeys under the [**BASIC**] key cannot be recorded since pressing [**BASIC**] turns keystroke recording off. It is also true that [**Save/Recall**] operations on programs do not record. You can, however, record all other save and recall operations not having to do with HP Instrument BASIC programs.

Although HP Instrument BASIC operations cannot be recorded, many do have corresponding SCPI mnemonics that allow an external controller to control and communicate with internal HP Instrument BASIC programs. For more information refer to the *HP 3589A HP-IB Programmer's Reference.*

Operations Requiring Additional Programming

Some operations that work well when performed from the front panel have special circumstances that require special attention when used in a program. This is due to two kinds of problems, synchronization and active control.

Synchronization

Timing and synchronization must always be anticipated where 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, say you want your program to manually arm a measurement, but only after a sweep has completed. You can record the command to set the instrument to manual arm mode, and the command to manually arm, by pressing key sequences. However, to detect when the instrument has completed a sweep, 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 AVER_SRQ program in chapter 10.)

Active Control

There are a few operations that require the analyzer to be the active controller on the external bus. This generally means that the analyzer must be set as System Controller before the program is run (or active control must be passed to it from an external controller, if one is connected). When an HP Instrument BASIC program begins running, however, the instrument's active control of the external interface is automatically passed to the program, so active control must be passed back to the analyzer before these operations can be performed. These operations include:

- [PRINT ALL]
- [PLOT TRACE]
- [PLOT TRACE MKR]
- [PLOT OFFSET MKR]
- [PLOT GRATICULE]
- [ABORT PRINT/PLOT]

You can keystroke record any of these operations but you will not be able to successfully run the program that is generated. You will need to enter the program lines necessary to first pass control to the analyzer and then wait for control to be passed back to the program. See "Passing and Regaining Control" in chapter 9 for an example of passing control to the analyzer.

Mnemonics With No Corresponding Front panel Operation

Several of the HP 3589A SCPI mnemonics for the instrument perform operations that are not available from the front panel and which, therefore, cannot be recorded. These include operations such as querying instrument status, transferring data over HP-IB, setting and clearing status registers and general HP-IB housekeeping.

These operations are useful for the more advanced HP-IB programming using HP Instrument BASIC, however because they fall outside the direct operating realm of the HP 3589A, they cannot be recorded. See the *HP 3589A HP-IBProgrammer's Reference* manual for a complete description of the analyzer's HP-IB command set.

Avoiding Recording Errors

Use Preset

In most cases, you should perform a preset before recording a measurement sequence, and again before running the recorded program. This sets the instrument to its default state and 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 is run.

You can include the command to perform a preset in your program by pressing [**Preset**] immediately after turning recording on. This inserts the following line prior to all other OUTPUT statements in your program:

```
OUTPUT @Hp3589a;"SYST:PRES"
```

Specifically Select Parameters

If you do not want the instrument preset before a recorded program is run (for example, you may be recording a section of a larger measurement sequence), be sure to specifically activate every instrument setting that you will need in your automated sequence. For example, if you want the format to be SINGLE, press [**Format**] and then [SINGLE], even though SINGLE is already the default setting. This will generate a program line to specifically set 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, if you want to generate a program line to set the active trace to be A, and you discover that it is already set to A when you start recording, press [**Active Trace**] twice — once to select B and then again to select A. You can easily remove unwanted program lines generated by this procedure in the editor.

Use GPIB Echo

GPIB Echo is a useful HP 3589A facility that allows you to view the SCPI mnemonic or mnemonics corresponding to any operation executed from the front panel. To turn on GPIB Echo, press [**Local/HP-IB**] and [ECHO **ON** OFF]. After doing this you will see a mnemonic appear at the bottom of the screen as you complete any key sequence that has a matching SCPI mnemonic. This is the exact mnemonic that is generated in your recorded program during a recording session.

Using GPIB Echo you can preview the SCPI mnemonic commands that will be stored in your program 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 a mnemonic is completed.

Chapter 4

Running, Pausing and Stopping Programs

4

Running, Pausing and Stopping Programs

Program control — starting, pausing and stopping an HP Instrument BASIC program — can be controlled from the HP 3589A front panel using various hardkeys and softkeys. These actions and their corresponding keys are described in this chapter.

HP Instrument BASIC programs may also be remotely controlled via SCPI commands over the HP-IB. For information on running, pausing and stopping programs from an external controller see chapter 9, "Interfacing with the HP-IB"

Running and Continuing a Program

To run any HP Instrument BASIC program in the HP 3589A program buffer, press the [RUN] softkey in the [**BASIC**] menu. You can also run a program from the DEBUG environment by pressing the [RUN] softkey in the [**BASIC**] [DEBUG] menu. Both of these [RUN] 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 require more than one program line to detect. Included in this are errors such as incorrect array references, and mismatched parameter or COM lists.

After [RUN] is pressed, and the prerun has successfully completed, the program will continue executing until one of the following events occurs:

- An END or STOP statement is encountered in the program.
- The [**Local/HP-IB**] hardkey is pressed to stop the program.
- The [**BASIC**] hardkey is pressed to pause the program.
- The [**Preset**] hardkey is pressed to reset the instrument.
- A PAUSE statement is encountered.

Programs can be continued from a paused state by pressing the [CONTINUE] softkey found in both the [**BASIC**] and the [**BASIC**] [DEBUG] menus. Continuing a paused program resumes program operation from where it was paused, retaining the current program context (variable values, etc.).

Pausing a Program

A program can be suspended temporarily by pressing the [**BASIC**] hardkey. To continue the program, press the [CONTINUE] softkey in the resulting [**BASIC**] menu or in the [**BASIC**] [DEBUG] menu.

The program can also be paused by inserting a PAUSE statement in the program, in which case the instrument will respond the same as if you had pressed [**BASIC**]. Refer to chapter 6, "Developing Programs," to learn how to insert statements in your recorded program.

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

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

To close all open files, you must perform an HP Instrument BASIC RESET. This can be done either by pressing [**Local/HP-IB**] while the program is running or by pressing the [RESET] softkey in the [**BASIC**] [DEBUG] menu when the program is paused. Keystroke recorded programs do not open files and therefore avoid this problem.

Stopping a Program

To stop a program completely, press the [**Local/HP-IB**] hardkey at any time while the program is running. This causes an HP Instrument BASIC RESET. Placing a STOP statement in your program will also terminate the program, but does not perform a BASIC RESET operation. After pressing [**Local/HP-IB**], all variable values remain intact and may be examined using the [**BASIC**] [DEBUG] [EXAMINE VARIABLE] softkey.

Press [**Preset**] to stop a running program and also reset the analyzer.

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

Chapter 5

Saving and Recalling Programs

5

Saving and Recalling Programs

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

The transferring of program data between memory and disk can be accomplished from the front panel using the [**Save/Recall**] menus. In addition, the GET, SAVE and RE-SAVE commands can be used within a program to transfer program files to and from disk. An autoload feature also exists to allow for a program to be automatically recalled from disk at power-up and run.

Another mode of program transfer is between the analyzer and an external controller, such as an HP Series 200/300 controller. You can upload a program into an external controller for easier editing—this is often necessary if you do not have an external keyboard attached to the HP 3589A. Fully developed programs may be downloaded from an external controller as well. The methods of transferring programs between the HP 3589A and an external controller are described in detail in chapter 9, "Interfacing with the HP-IB."

This chapter describes all program transfer operations between program memory and the HP 3589A volatile, non-volatile and internal disk drives.

Saving a Program to Disk

To save the current contents of the program buffer to a disk file, use the same [**Save/Recall**] menu system used for all disk access in the HP 3589A.

If you are saving a program to a new file name, press [**Save/Recall**], [SAVE MORE], and [SAVE PROGRAM] and then enter the name of the disk file using the front-panel keys or the optional keyboard. All HP Instrument BASIC programs are stored as ASCII files.

If you are re-saving a program – that is, saving a file to a disk that already contains the file name – you can press [**Save/Recall**], [SAVE MORE], and [RE-SAVE PROGRAM] and then enter the file name. To make this process a little easier, you can catalog the disk first ([**Save/Recall**], [CATALOG **ON** OFF]), use the knob to highlight the desired file name and then press [SAVE MORE] and [RE-SAVE PROGRAM]. The re-save file name will automatically be entered for you.

Recalling a Program from Disk

To recall a program file from the disk to the program buffer, use the [**Save/Recall**], [RECALL MORE], and [RECALL PROGRAM] key sequence. As with any recall operation, you can catalog the disk first ([**Save/Recall**], [CATALOG **ON** OFF]) and select the file name that will then appear in the [RECALL PROGRAM] prompt.

Note that any program recalled to the program buffer using the [**Save/Recall**] menus will overwrite the current contents of the program buffer.

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 and the HP Instrument BASIC syntax error is displayed briefly in an error message and written to the CRT at the same time. To view errors messages logged to the CRT, use the [**BASIC**] [DISPLAY FORMAT] menu to allocate a screen partition.

When the [**Save/Recall**] menus are used to recall a program, the memory size is automatically allocated. Memory size is used for the program's variables and working space and is commonly called the "stack." For certain kinds of programs, the memory size may need to be enlarged. This can be accomplished with the [AUTO MEMORY] or [MEMORY] softkeys in the [**BASIC**] [UTILITIES] menu.

See chapter 6, "Developing Programs" for more information on allocating partitions or allocating memory.

Appending Program files from Disk

The only method offered by HP Instrument BASIC in the HP 3589A for appending program files from disk to the current program in memory is to use the GET command within a program. The GET command will recall a specified file from the disk and append it at a specified line in the current program (or at the beginning if no line is specified).

The following example program appends three program files to itself to build one functional program. It not only demonstrates how to merge files but 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 and a cleanup section that contains error- and timeout-traps. The core six line program chains the other three program segments to itself and must be deleted or commented out before the program is run.

All of these files are on the HP Instrument BASIC Installation Disk/Programming Examples:

- SHELLBEGIN provides the setup and initialization
- SHELLDEMO is a typical keystroke recorded program
- SHELLEND provides error-handling routines and cleanup
- SHELLCHAIN pulls all files together using GET commands

The file SHELLCHAIN contains the following program:

```
1   ! NOTE: DELETE LINES 1 THRU 5 IMMEDIATELY AFTER RUNNING PROGRAM
2   GET "SHELLBEGIN:,4",X,3
3   GET "SHELLDEMO:,4",X,4
4   GET "SHELLEND:,4",X,5
5   GOTO Endlabel
6 X: END
```

Line 2 performs a GET of the file "SHELLBEGIN" from the internal drive, appends it at the current label "X:" (line 6), overwriting that line, and then instructs the program to continue at line 3.

The "SHELLBEGIN" file has a label "X:" as its last line so now the program in memory also has that label as its last line. Line 3 performs a GET of the file "SHELLDEMO", appending it at the current label "X:" and then continues the program at line 4.

Line 4 performs a GET to append the SHELLEND program file to the current program. Since the SHELLDEMO file contained a label "X:" at its end as well, the SHELLEND program is appended to the end of the program in memory.

Finally, line 5 skips to the label "Endlabel", which is at the end of the SHELLEND file and now at the end of the program in memory. You could make the program run itself immediately by omitting this line.

Note

It is important to remember to comment out or delete the first five lines before running the combined program.

To use the SHELLCHAIN program with your own recorded program you must do two things. First you must insert the label “X:” in the line containing the END statement of your recorded program. Then you must recall the SHELLCHAIN program and change the file name in line 3 from “SHELLDEMO” to the name of your program file. Make sure you have not used the label “X:” elsewhere in your program.

Autoloading a Program

HP Instrument BASIC allows you to designate a program to be automatically loaded and run when the instrument is first powered up. To make an autoloading program, save it with the file name "AUTO_BAS" either in non-volatile memory or on a floppy disk in the internal drive. At powerup, HP Instrument BASIC searches first the non-volatile drive and then the internal disk drive for the file "AUTO_BAS." If it is found, it is loaded and executed immediately.

Chapter 6

Developing Programs

6

Developing Programs

Overview

For many applications, you can easily record and run programs without any need to alter the program code that is generated. However, with some knowledge of the HP Instrument BASIC language and the program development capabilities in the HP 3589A, you can make recorded programs more powerful or create your own programs from the ground up.

This chapter describes the operation of the following keys in the [**BASIC**] menu, and any softkeys found in their underlying menus:

- [EDIT]
- [PRINT PROGRAM]
- [UTILITIES]
- [DISPLAY FORMAT]

The ability to change and enhance your program and its operating environment is found primarily under the [EDIT] and [UTILITIES] menus in the [**BASIC**] menu. Pressing [EDIT] places you in the editor where you can make changes to your program on a line-by-line basis using a keyboard or the front panel alpha keys. Pressing [UTILITIES] presents a menu of various utilities, allowing you to make global changes to the program and its environment. These include the ability to renumber lines, allocate memory size, remove the program, and so on.

The [PRINT PROGRAM] softkey allows you to obtain a hard copy printout from an attached printer and the [DISPLAY FORMAT] softkey menu lets you manage which part of the CRT display is used for program output and also lets you clear the current display partition.

Using the HP Instrument BASIC Editor

The editor is used for creating and altering program text in an HP Instrument BASIC program. Those familiar with the editor found in HP Series 200/300 BASIC will find it somewhat similar to the HP Instrument BASIC editor; others should find it easy to learn and use. This section tells you how to edit and enter an HP Instrument BASIC program.

To start the editor, press the [EDIT] softkey in the [**BASIC**] menu. You will see the program, if one exists, appear on the display with a 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 begin entering text.

Two Ways to Edit Programs

You can edit the program using the [EDIT] softkeys and the analyzer's front panel alpha-numeric keys, or you can edit with the optional keyboard. You may use a combination of all three to edit.

The [EDIT] menu contains the following softkeys:

- [ENTER]
- [INSERT SPACE]
- [INSERT LINES]
- [DELETE LINE]
- [RECALL LINE]
- [DELETE CHARACTER]
- [TYPING UTILITIES]
- [GOTO LINE]
- [END EDIT]

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 if needed. All other lines have only their first 51 characters displayed (excluding line numbers).

Each line has a numeric field in the first 6 columns in which program line numbers are right justified. Although program lines are automatically numbered by the editor, you can 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 from the [UTILITIES] [RENUMBER] softkey menu. The range of line numbers is from 1 to 32766.

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.

To end an editing session, return to the editing menu and press [END EDIT] ([*F10*] on the keyboard). This returns you to the [**BASIC**] menu.

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 6-2.

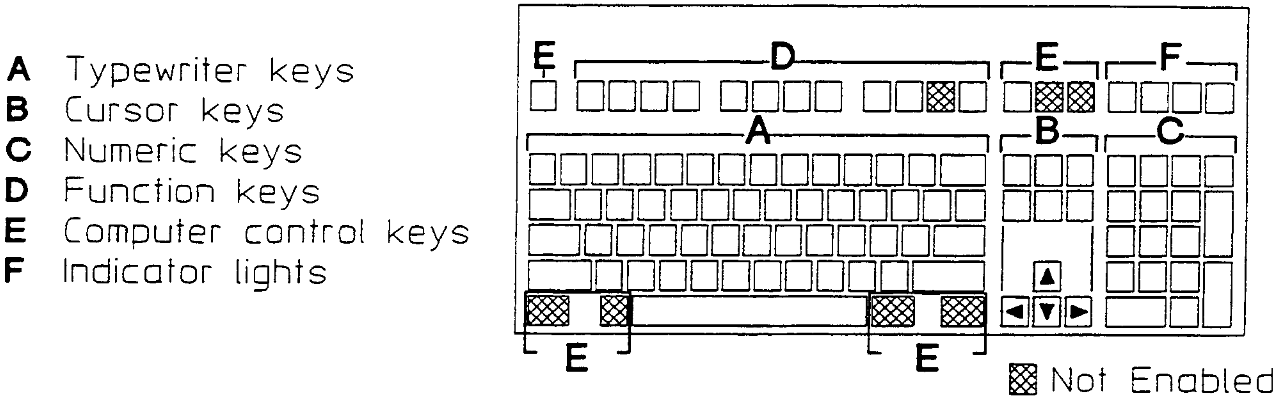


Figure 6-1. Using a keyboard with the HP 3589A

You use the [*Enter*] key 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 3589A softkey menus load into the keyboard function keys, [*F1*] through [*F10*]. The [**Help**] hardkey loads into [*F12*]. [*F11*] is not enabled. See figure 6-1.

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.

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!)

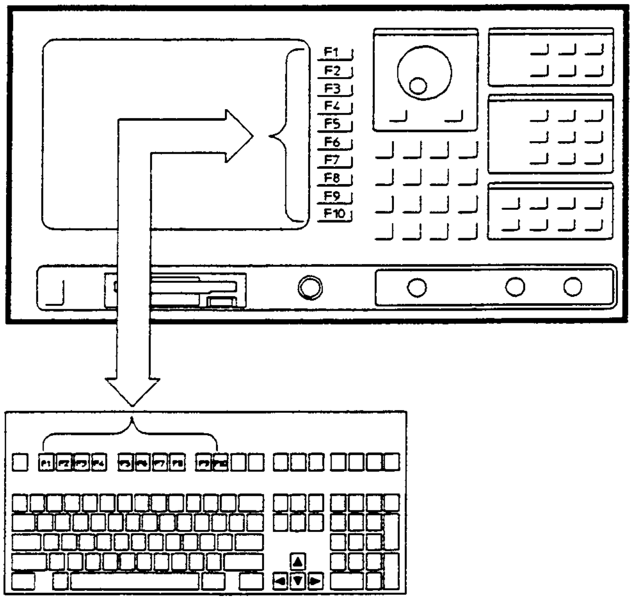


Figure 6-2. Mapping of the HP 3589A softkeys

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 3589A, plug the round connector into the front of the analyzer. See figure 6-3.

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

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.

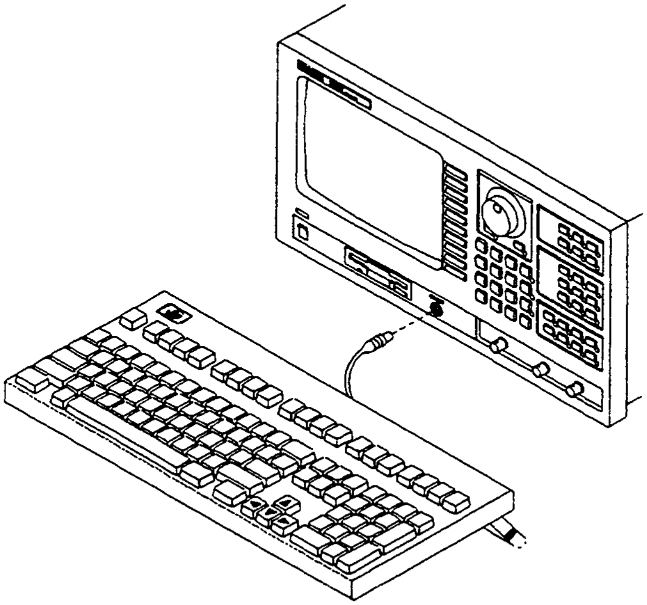


Figure 6-3. Connecting the keyboard

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

Most incremental cursor movement in the EDIT mode is accomplished with the knob. The only exception is the [ENTER] softkey, which moves the cursor down one line at a time. All other keys that cause cursor movement also alter program text.

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

Scrolling up or down in the program is made very easy since the cursor stays in the far left line position once it is reached for counterclockwise rotation, and at the far right when rotating the knob clockwise. However, you may notice that stopping the scroll at a particular line requires a little practice since only one cursor jump (approximately 1/30th of one rotation) is required from line to line. An easy workaround for this is to stop scrolling a few lines above the desired line and use the [ENTER] softkey to move the cursor down one line at a time to the line you want to edit.

Using GOTO LINE

You can jump immediately to any line or label in the program by pressing the [GOTO LINE] softkey. After pressing [GOTO LINE], you are prompted to enter a line number or label. Enter the line number or the label of the line on which you want the cursor positioned and press [ENTER]. To specify a label, use the front-panel keys and the optional [(_)] underscore softkey. You can enter the label in capital letters and it will be automatically converted.

If the specified line exists, it will appear in the middle of the display as the current program line. If you have specified a line number that doesn't exist, the cursor will be placed on the line number closest to it. Specifying a non-existent line label causes an error to be generated.

A common 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).

Entering Program Lines

When you finish entering or changing a program line, store it into the program by pressing the [ENTER] softkey. The computer checks the line for syntax errors and converts letter case to the required form for names and keywords (HP Instrument BASIC commands). If no errors are detected, it then 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 will be lost.

Renumbering, Copying and Moving Lines

If you want to change the line number of an edited program line, simply move the cursor to the line number field and enter the line number you want. Changing the line number causes a copy operation, not a move. Therefore, if you only want to move the line, change the line number first, press [ENTER] and then delete the original line.

If you want to create an edited copy of the current line, edit the line and then change the line number and press [ENTER]. The edits will only appear in the copied line.

If you are in insert mode and you change the line number, you will be placed in insert mode at the new location of the program specified by the new line number.

You will notice that when the cursor is in the line number field, entries operate in an overtype fashion rather than in the insert fashion as in the text portion of the program line. Also the [**Back Space**] key simply moves the cursor over line numbers without deleting the number.

Using the Front-Panel Alpha Entry Keys

If you do not have a keyboard, you can use the alpha keys on the HP 3589A. Nearly every hardkey on the instrument is labeled with a corresponding alpha key. These should be familiar to you if you have performed any editing function on the HP 3589A, 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. All HP Instrument BASIC keywords can be entered via softkeys and do not have to be entered using the alpha keys (see [TYPING UTILITIES] description). The alpha keys are mostly necessary to enter variable names, constants, labels and strings.

When the analyzer is in edit mode, the front-panel keys are automatically in alpha mode; pressing an alpha key will enter its character at the cursor position in the current program line. When you exit the editor ([END EDIT]) the front-panel keys return to the instrument mode.

Changing the Letter Case

The case of letters entered by pressing an alpha key is determined by the state of the [TYPING UTILITIES] [**UPPERCASE** lowercase] softkey, which defaults to uppercase. To enter lowercase alpha letters, press the [UPPERCASE **lowercase**] softkey first to toggle to lowercase.

Inserting Spaces

The [INSERT SPACE] softkey places a space at the position of the cursor, causing text to the right of the cursor to move one place to the right. This is similar to a space bar on a standard keyboard. Since this function is used quite often, it can be found in more than one menu.

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 LINES]. The [INSERT LINES] softkey functions as a toggle to turn insert mode on and off.

As an example, assume you want to insert some lines between two adjacent programs lines numbered 90 and 100. Place line 100 in the current line position and press [INSERT LINES]. The program display "opens" and a new line number appears between line 90 and line 100, numbered 91. Enter and store the inserted line and another inserted line will appear numbered 92. If, after continuing to enter lines in this manner, the inserted line number increments to 100, then the current line 100 will be renumbered one higher to accommodate the inserted line.

You can get out of insert mode either by pressing [INSERT LINES] again or by using the knob to move off of the current line. Remember, any edits you have made to the currently inserted line will be lost when you leave insert mode. You should therefore make sure you have entered any changes to your final inserted line (by pressing [ENTER]) before exiting insert mode.

Recalling Deleted Lines

The [RECALL LINE] softkey automatically recalls the last line that was deleted from the program (using [DELETE LINE]) since entering the editor. This can be useful for recovering lines deleted by mistake. It is also handy for moving a line when you are not sure of the line number it will be moved to. Simply delete the line, move to the desired program area, press [INSERT LINES] (note the current line number), [RECALL LINE] and then renumber the recalled line to the current line number.

Note

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

Using [TYPING UTILITIES]

To enter non-alphabetic symbols and insert HP Instrument BASIC keywords, press [TYPING UTILITIES]. This softkey contains the following softkeys:

- [ENTER]
- [INSERT SPACE]
- [INSERT KEYWORD]
- [DELETE CHARACTER]
- [**UPPERCASE** lowercase]
- [INSERT + – * ^ /=( )]
- [INSERT " & # ; :, @ |]
- [INSERT $ < > [ ]{ }\ ]
- [INSERT ~ % ' ! ? ‘_]
- [CANCEL/RETURN]

Entering Symbois

The symbols are arranged in four sub-menus with the label "INSERT" followed by a list of the available symbols in that menu. For example, to enter an exclamation symbol (!) to specify a comment line, press [TYPING UTILITIES], then press [INSERT ~%'!?'_]. This will bring up a softkey menu with each of the symbols listed in the label as a separate softkey. Then, simply press the softkey labeled [!].

Notice that the [ENTER], [INSERT SPACE], and [DELETE CHARACTER] softkeys are carried over from the [EDIT] menu. In addition are four symbol entry softkeys, the [**UPPERCASE** lowercase] softkey and the [INSERT KEYWORD] softkey.

Entering Keywords

To enter any HP Instrument BASIC keyword, first press [TYPING UTILITIES] and then press [INSERT KEYWORD]. At this point, pressing any alpha key will present a softkey menu of keywords beginning with that letter. For example, pressing [INSERT KEYWORD] and then the alpha key "A" (hardkey [**Trace Type**]) 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**], will enter 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] will appear to allow 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 belonging to a specific alpha key.

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(". This identifies that keyword as requiring an argument. When one of these keywords is selected, the keyword and both parentheses are inserted in the program line with the cursor placed automatically between them.

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, simply position the cursor on the line in front of which 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.

Note that the "ASSIGN @Hp3589Aa to 800" statement is **not** generated when you are recording into an existing program and **must** be included in your program prior to any recorded OUTPUT commands. If you initially created the program using recording, this statement should already exist. If it does not exist, you will need to enter it.

Removing Program Text

You can remove individual characters or entire lines from within the editor. To learn how to remove the entire program see the description of the [UTILITY] [SCRATCH] softkey later in this chapter.

Deleting Characters

The [DELETE CHARACTER] softkey removes the character under the cursor and moves all characters to the left one place. Repeatedly pressing [DELETE CHARACTER] will cause text to the right of the cursor to be pulled in and removed. The [DELETE CHARACTER] softkey functions the same in both the line number and program statement fields. When used in the line number field, it pulls in and deletes only line numbers to the right of the cursor (not program statement characters).

Another way to remove text on a line is with the [**Back Space**] key on the numeric entry 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.

Deleting Lines

The [DELETE LINE] softkey allows you to remove the current program line and place 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 [DELETE LINE] is 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 of line. This puts the editor in insert mode on the last line of the program (see "Inserting Lines"). (To get out of insert mode, simply move the cursor up one line with the knob.)

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

To recall the last line that was deleted, press the [RECALL LINE] softkey.

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. The [UTILITIES] menu contains the following softkeys:

- [MEMORY SIZE]
- [AUTO MEMORY]
- [SCRATCH]
- [RENUMBER]
- [SECURE]
- [CANCEL/RETURN]

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 operating space.

MEMORY SIZE

The [MEMORY SIZE] softkey allows you to manually specify the amount of working space used by the active program. This lets you adjust the "stack space" of your program. The size of the stack determines the total storage space for all variables not in COM. It is also where context information is stored when calls to subprograms are made. For this reason, recursive subprograms, which continually save variable and context information on the stack, can use up allocated memory size very quickly. Memory space does not contain the program code.

To adjust or display the memory size, press [MEMORY SIZE] and the current memory size will be displayed for several seconds. At any time, after the [MEMORY SIZE] softkey is highlighted, you can enter a new memory size from the numeric key pad. The first numeric key touched brings up the MEMORY SIZE prompt; then use the numeric key pad to enter numbers. You may also use the [EXP] softkey in conjunction with the numeric keys to enter numbers in engineering notation.

After entering the memory size press [ENTER]. The closest available memory allotment size to the entered number is then assigned to the active program. Memory allotment sizes are determined by HP Instrument BASIC for the most efficient use of memory resources.

Acceptable values for memory size range from 1,122 to 500,000 bytes in 2 byte increments for the standard HP 3589A without the additional memory option. Entering a number greater than the maximum causes memory size to default to the largest available stack space. Entering a number less than the minimum (1122) has no effect on the current memory size. With additional memory added, the memory size can be as large as 3 MByte.

Note

The maximum memory size may be limited by factors such as volatile RAM disk size and program size if the 2 Mbyte memory option is not installed. For example, if the analyzer does not have the 2 Mbyte option installed and you have a large disk formatted in volatile memory, you may not be able to allocate a memory size of 500,000.

AUTO MEMORY

The [AUTO MEMORY] softkey resizes memory automatically to fit the currently active program. This is similar to the operation that occurs when a program is loaded via the [**Recall**] menus. This is faster than using the [MEMORY SIZE] key and works well for most programs. Programs that use recursive procedures will most likely need to have memory sized manually.

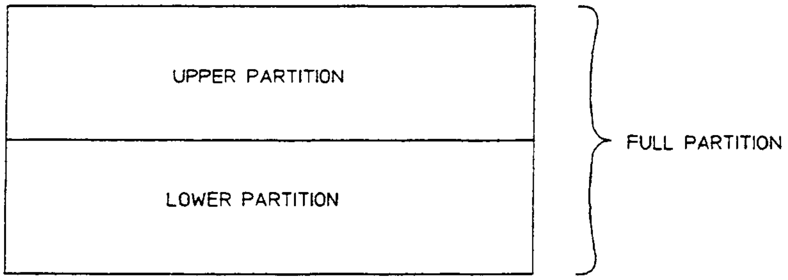


Figure 6-4. The Display Partitions

SCRATCH

Pressing the [SCRATCH] softkey brings up a menu allowing you to clear the current program and/or variables. The menu contains the softkeys:

- [SCRATCH]
- [SCRATCH C]
- [SCRATCH A]
- [PERFORM SCRATCH]
- [CANCEL/RETURN]

First select a combination of program and/or variables to clear by pressing [SCRATCH], [SCRATCH C], or [SCRATCH A]. Then press [PERFORM SCRATCH].

SCRATCH

This softkey selects the HP Instrument BASIC program and all variables not in COM.

SCRATCH C

This softkey selects all variables including those in COM, but not the program.

SCRATCH A

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

To cancel the SCRATCH operation, press [CANCEL/RETURN] at any time prior to pressing [PERFORM SCRATCH].

RENUMBER

Pressing [RENUMBER] will bring up a menu allowing you to change the line numbering in the entire active program. The [RENUMBER] menu contains the following softkeys:

- [START LINE #]
- [INCREMENT]
- [PERFORM RENUMBER]
- [CANCEL/RETURN]

When renumbering lines in a program you may select the number that will be assigned to the first line in the program by pressing [START LINE #]. Press [INCREMENT] to specify the increment between the renumbered line numbers. For example, if [START LINE #] is 10 and [INCREMENT] is 10, the new line numbers will be 10..20..30..40... and so on.

When the start and increment are defined, press [PERFORM RENUMBER] to execute the command. To cancel the renumbering operation, press [CANCEL/RETURN] at any time prior to pressing [RENUMBER PROGRAM].

SECURE

Pressing [SECURE] brings up a menu allowing you to protect program lines so that they cannot be listed to a printer or viewed in EDIT mode. The [SECURE] menu contains the following softkeys:

- [START LINE #]
- [END LINE #]
- [PERFORM SECURE]
- [CANCEL/RETURN]

To secure a block of the active program, press [START LINE #], enter the beginning line number of the program block, then press [END LINE #], enter the ending line number of the block, and then press [PERFORM SECURE]. If you set [START LINE #] to 1 and [END LINE #] to 32766, you can secure the entire program by pressing [PERFORM SECURE].

Secured lines cannot be listed, or viewed in the editor and appear in both with only an asterisk following the line number (*). Secured lines can, however, be deleted from the program using the editor. You may exit this menu at any time by pressing [CANCEL/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 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]. Note that the analyzer must also be set as System Controller to enable printing over the HP-IB.

Note

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

Using [DISPLAY FORMAT]

Pressing the [**BASIC**] [DISPLAY FORMAT] softkey allows you to allocate a partition of the analyzer's display to be used by your programs or, alternately, to return any allocated partition to the analyzer. The [DISPLAY FORMAT] menu contains the following softkeys:

- [OFF]
- [FULL]
- [UPPER]
- [LOWER]
- [CLEAR SCREEN]
- [CANCEL/RETURN]

The HP 3589A 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.

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 FORMAT] softkey.

Allocating display partitions can be accomplished from within your program using the SCPI mnemonic "DISP:PART" and specifying the parameter UPP, LOW or FULL. For example the statement

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

allocates the single trace area of the display, corresponding to selecting [FULL] from the [DISPLAY FORMAT] menu. Table 6-1 shows the relationship of the [DISPLAY FORMAT] menu softkeys and their functions, to the corresponding SCPI mnemonics required to program the same functions.

Table 6-1.The Display Partitions

MENU	ALLOCATES	SCPI MNEMONIC
OFF	NO DISPLAY	DISP:PART OFF
FULL	SINGLE TRACE AREA	DISP:PART FULL
UPPER	UPPER TRACE AREA	DISP:PART UPPER
LOWER	LOWER TRACE AREA	DISP:PART LOWER

Most display allocation should be handled by your program via the SCPI mnemonics. These softkeys are best utilized during program development.

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

For more information refer to chapter 8, "Graphics and Display Techniques."

Chapter 7

Debugging Programs

7

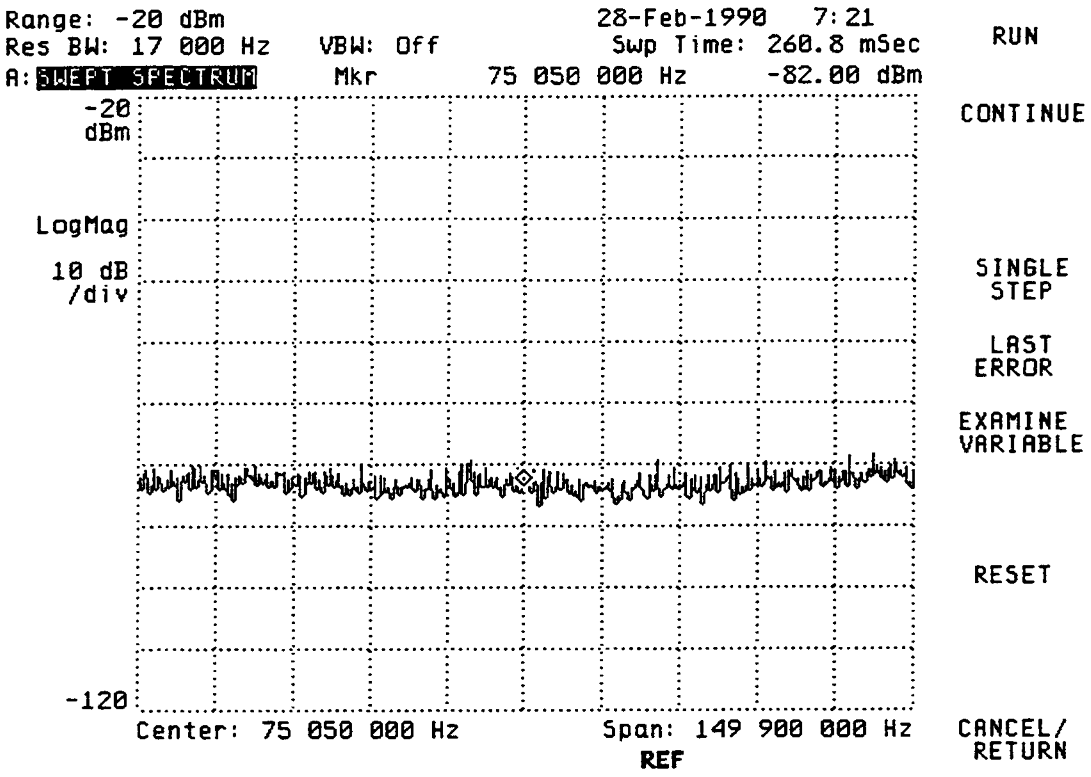
Debugging Programs

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 and HP Instrument BASIC contains some very useful features to help you to track them down.

Overview

The HP Instrument BASIC tools provided for program debugging are simple and, used properly, can be very helpful. The [**BASIC**] menu contains the [DEBUG] softkey. Figure 7-1 shows the [DEBUG] menu.



```
Range:  -20 dBm                                   28-Feb-1990    7:21
Res BW:  17 000 Hz    VBW:  Off                    Swp Time:  260.8 mSec
A: SWEPT SPECTRUM      Mkr          75 050 000 Hz        -82.00 dBm
Center:  75 050 000 Hz                        Span:  149 900 000 Hz
```

Figure 7-1. 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.

Using the [SINGLE STEP] softkey, you can execute the program one line at a time to study its 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. You can then press [CONTINUE] to resume operation to the next PAUSE statement (or the program end), or you can press [SINGLE STEP] to walk through program lines following the PAUSE statement.

As you can see, a variety of combinations can be used to effectively examine the program's operation and solve your particular problems.

Using [EXAMINE VARIABLE]

Pressing [EXAMINE VARIABLE] displays a prompt box to allow you to enter the name of the variable you want to examine. The name of last variable name examined is the default variable. It also brings up the alpha entry menu, which allows you to enter the variable name.

To examine the value assigned to any variable in your program you must first perform a prerun and then press the [EXAMINE VARIABLE] key and enter the name of an existing variable in your program. A prerun is accomplished by pressing either [RUN] or [SINGLE STEP].

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

Note that you can enter the variable as all uppercase letters if you want. When you are finished entering variable names, press [ENTER].

Examining Strings

Enter string variables as you would any other variable. Any string variable entered without delimiters will display as much of the string as will fit in the display box. The display box will wrap to display up to 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 of 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

To select an array to be examined you can either select individual elements or the entire array. For example the entry:

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

selects the elements 1 through 3 of the array *I_array* to be displayed.

You may select an entire array to be examined by entering the array variable name and specifying (•) (for example, I_array(•)). For example, if I_array(20) is an integer array, and the first and second elements are set to 100, entering "I_array(•)" after pressing [EXAMINE VARIABLES] would display:

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

Individual array elements (e.g., I_array(17)) can also be specified the same as any single variable.

Setting Breakpoints

A common method of debugging a program involves 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 can be accomplished by inserting PAUSE statements in the program code. When the program is then run, you can use [EXAMINE VARIABLE] to check or change variable values and 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 6, "Developing Programs," for a description of the HP Instrument BASIC editing capabilities in the HP 3589A.

Using [SINGLE STEP]

The [SINGLE STEP] softkey allows you to execute your program one line at a time. The line about to be executed appears on the top line of the CRT.

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. You can press [CONTINUE] to resume program execution at any time.

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

The [RUN] softkey operates the same as pressing the [**BASIC**] [RUN] softkey. 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 simply displays the error number and message of the last error encountered by the program. This is the front panel equivalent to the ERRM$ command.

Using [RESET]

The [RESET] softkey is included in this menu to allow 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] closes all open files, sends an abort message to the HP-IB interface and resets the program counter to the first program line.

Chapter 8

Graphics and Display Techniques

8

Graphics and Display Techniques

HP Instrument BASIC programs have the ability to allocate portions of the instrument's display for text and graphics display. 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 CRT 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, using the [**BASIC**] [DISPLAY FORMAT] softkey menu, when the program is not currently running. To perform equivalent actions from within a running program requires sending an SCPI 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 areas (upper and lower). The upper and lower trace areas can be combined into one large trace area for single trace displays. Any of these three main areas, 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, you simply send the instrument the corresponding SCPI mnemonic. "DISP:PART UPPER" allocates the upper partition, "DISP:PART LOWER" allocates the lower partition, and "DISP:PART FULL" allocates the full screen partition.

For example, to print a message to the upper partition area, you might use a program segment like this:

```
30  ASSIGN @Hp3589a TO 800
40  OUTPUT @Hp3589a;"DISP:PART UPPER"
50  CLEAR SCREEN
60  PRINT "This is the upper partition"
```

To be sure that you are not writing to a partition that has not yet been assigned, you should include a WAIT statement or, even better, add a SCPI query command followed by an ENTER statement to synchronize the program with the instrument. The previous example might then look like this:

```
30  ASSIGN @Hp3589a TO 800
40  OUTPUT @Hp3589a;"DISP:PART UPPER"
50  OUTPUT @Hp3589a;"DISP:PART?"
60  ENTER  @Hp3589a;Part$
70  CLEAR SCREEN
80  PRINT "This is the upper partition"
```

The mnemonic DISP:PART? (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:PART OFF” mnemonic. 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 the user to view instrument trace data. The following example example demonstrates this command:

```
830  OUTPUT @Hp3589a;"DISP:PART 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. Note that causing the screen to scroll does not effect any graphics displayed on the screen, since 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, and 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"
 .
 .
```

A useful technique to get text onto the screen quickly is to 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. The following program segment demonstrates this:

```
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 data to the screen.

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

Note

Using external computers to develop programs, it is sometimes a practice to embed these control characters in your PRINT statements. 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 characters and its actions may be unpredictable.

Using Graphics

The graphics commands of HP Instrument BASIC are easy to understand and use. You can 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
```

This 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 8-1.

Table 8-1. Maximum Pixel Coordinates

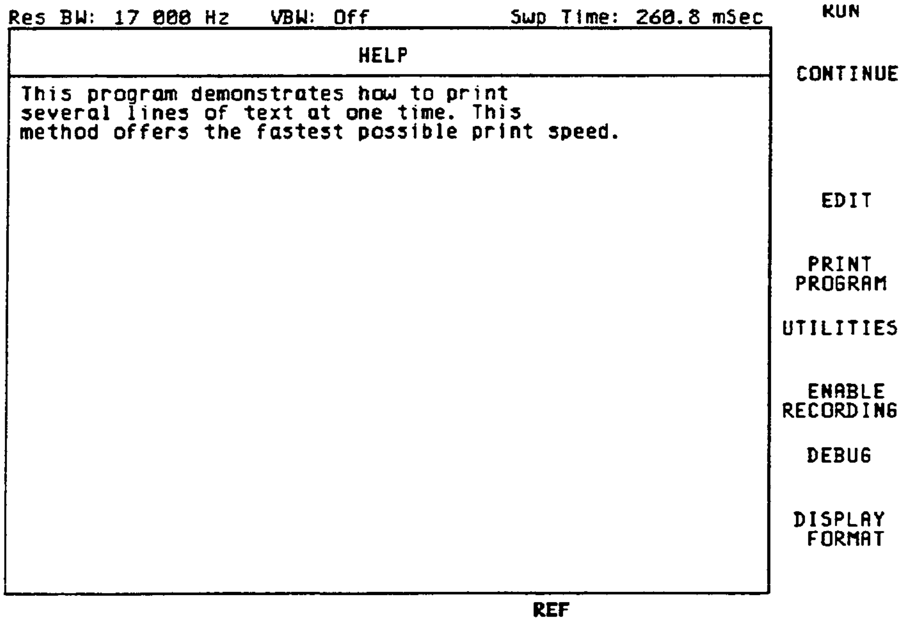
Partition	X Max	Y Max
FULL	475	355
UPPER	475	173
LOWER	475	173

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:PART FULL"
30   OUTPUT 800;"DISP:PART?"
40   ENTER 800;P$
50   GCLEAR
60   CLEAR SCREEN
70   MOVE 0,0
80   DRAW 475,0
90   DRAW 475,371
100  DRAW 0,371
110  DRAW 0,0
120  MOVE 0,330
130  DRAW 475,330
140  PRINT TABXY(28,1);"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 following screen display:



```
Res BW: 17 000 Hz   VBW: Off          Swp Time: 260.8 mSec
                          HELP
This program demonstrates how to print
several lines of text at one time. This
method offers the fastest possible print speed.
```

Figure 8-1. Program Output

Chapter 9

Interfacing with the HP-IB

9

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 used 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 standard. 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 also briefly described in this chapter.

HP-IB Device Selectors

Since the HP-IB allows the interconnection of several devices, each device must have a means of being uniquely accessed. Specifying just the interface select code of the HP-IB interface to which a device is connected is not sufficient to uniquely identify a specific device on the bus.

Each device on the bus has an primary address by which it is identified. This address must be unique to allow individual access of each device. Each HP-IB device has a configurable address. Thus, 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, which is addressed with select code 7.

The second part of an HP-IB device selector is the device's primary address, which is in the range of 0 through 30. For example, to specify the device on the interface at select code 7 with a primary address of 22, use device selector = 722.

Remember that 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 8 (e.g., 800, 811, 822, etc.).

Moving Data Through the HP-IB

Data is output and entered into the program through the HP-IB with the OUTPUT and ENTER statements, respectively. The only difference between the OUTPUT and ENTER statements for the HP-IB and those for other interfaces is the addressing information within HP-IB device selectors.

Examples

```
100    Hpib=7
110    Device_addr=22
120    Device_selector=Hpib * 100 + Device_addr
130    !
140    OUTPUT Device_selector;"F1R7T2T3"
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 help to ensure that only orderly communication may take place on the bus. For conceptual purposes, the organization of the HP-IB can be compared to that of a committee. A committee has certain "rules of order" that govern the manner in which business is to be conducted. For the HP-IB, these rules of order are the IEEE 488.1 standard.

On the HP-IB, the System Controller corresponds to the committee chairman. The system controller is generally designated before running a program and cannot be changed under program control. However, it is possible to designate an "acting chairman" on the HP-IB. On the HP-IB, this device is called the Active Controller, and may 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. Thus, only one device can be designated System 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 a committee, only one person at a time may speak. It is the chairman's responsibility to "recognize" which one member is to speak. Usually, all committee members present always listen; 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 an individual device or a group 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 are to be the listeners. Before these assignments are given, he must get the attention of all members. The talker and listeners are then designated, and the next data message is presented to the listeners by the talker. When the talker has finished the message, the designation process may be repeated.

On the HP-IB, the Active Controller takes similar action. When talker and listeners are to be designated, the attention signal line (ATN) is asserted while the talker and listeners 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 by the ATN line being true, while data messages are sent with the ATN line false.

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

The data transfer, or data message, allows for the exchange of information between devices on the HP-IB. Our 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 listeners 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. For instance, when an OUTPUT statement is used (by the Active Controller) to send data to an HP-IB device, the following sequence of commands and data is sent through the bus.

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

1. The unlisten command is sent.
2. The talker's address is sent (here, the address of the active controller), which is also a command.
3. The listener's address (01) is sent, which is also a command.
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) is sent, which is a command.
3. The listener's address is sent (here, the active controller's address), also a command.
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. Here is a summary of the HP Instrument BASIC statements that invoke these control mechanisms.

ABORT is used to abruptly terminate all bus activity and reset all devices to 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 discussion. However, the actions that a device takes upon receiving each of the above commands are, in general, different for each device. For external devices, refer to the particular device's manuals to determine how it will respond.

Note that all of the bus management commands, with the exception of ABORT, require that the program be the Active Controller on the interface. A running HP Instrument BASIC program is always Active Controller on the internal interface (select code 8). For the program to be active controller on the external interface (select code 7), the host instrument must either be set as System Controller or have control passed to it from an external controller. The program automatically assumes the controller status of the host instrument. For more 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. If the device's front-panel controls are currently functional, it is in the Local state. If it is being controlled through the HP-IB, it is in the Remote state. Pressing the front-panel "Local" key will return the device to local (front-panel) control, unless the device is in the Local Lockout state, or is the host instrument.

The Remote message is automatically sent to all devices whenever the System Controller is powered on, reset, or sends the Abort message. A device also enters the Remote state automatically whenever it is addressed. The REMOTE statement also outputs the Remote message, which 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.

Examples

```
REMOTE 7

ASSIGN @Device TO 700
REMOTE @Device

REMOTE 700
```

Host Instrument

The REMOTE statement has no effect on the host instrument since it is always in remote control whenever an HP Instrument BASIC program is running. Specifying the internal interface in a REMOTE statement will not generate an error, but will have no effect.

LOCAL LOCKOUT

External Devices

The Local Lockout message effectively locks out the "local" switch present on most HP-IB device front panels. This prevents a device's user from interfering with system operations by pressing buttons and thereby maintaining 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. However, executing the ABORT statement does not cancel the Local Lockout message.

Host Instrument

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

LOCAL

External Devices

During system operation, it may be necessary for an operator to interact with one or more external devices. For instance, an operator might need to work from the front panel to make special tests or to troubleshoot. And, in general, it is good systems practice to return all devices to local control upon conclusion of remote-control operations. 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(s). However, if only the interface select code is specified (LOCAL 7), the Local message is sent to all devices on the external interface and 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 since it is always in remote control whenever an HP Instrument BASIC program is running. Specifying the internal interface in a LOCAL statement will 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. Because 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 outputs a Trigger message to all devices currently addressed to listen on the bus. Including a device 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. Note that the instrument must be set to trigger on the HP-IB for this command to be effective.

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 may be used to terminate all activity on the external bus and return all the HP-IB interfaces of all devices 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 either the Active 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 causes active control to be returned. Only the interface select code may be specified; primary-addressing information (such as 724) must not be included.

Examples

```
ASSIGN @Hpib TO 7
ABORT @Hpib

ABORT 7
```

Aborting the Internal Bus

ABORT is not supported for select code 8. Executing ABORT 8 will 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 operating and/or programming manuals for each device describe the device's capability to request service and conditions under which the device will request 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 may be executing 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 (executes remedial action).

Setting Up and Enabling SRQ Interrupts

In order for an HP-IB device to be able to initiate a service routine in the Active Controller, two prerequisites must be met: the SRQ interrupt event must have a service routine defined, and 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 the current line is exited (either when the line's execution is finished or when the line is exited by a call to a user-defined function). The service routine, in general, must perform the following operations:

1. Determine which device(s) are 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 preceed 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, if an SRQ is present momentarily but does not remain long enough to be sensed by the controller, an interrupt will not be generated. The level-sensitive nature of the SRQ line also has further implications, which are described in the following paragraphs.

Example

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. Since only service request interrupts are enabled in HP Instrument BASIC, the type of interrupt need not be determined either. 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 ! Use previous mask.
540  RETURN
```

The IEEE standard has defined 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 SRQ).

Had the SRQ line not been released, the controller would have branched to the service routine immediately upon re-enabling interrupts on this interface. This is another implication of the level-sensitive nature of the SRQ interrupt.

It is also important to note that once an interrupt is sensed and logged, the interface cannot generate another interrupt until the initial interrupt is serviced. The controller disables all subsequent interrupts from an interface until a pending interrupt is serviced. For this reason, it was 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. One entire byte of device-specific status is returned in response to a Serial Poll. This byte is called the "Status Byte" message and, depending on the device, may indicate an overload, a request for service, or a printer being 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 program must currently be the Active Controller in order to execute this function.

Examples

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

Spoll_724=SPOLL(724)
```

The Serial Poll is meaningless for an interface since it must poll individual devices on the interface. 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. Control can only be passed on the external interface (select code 7). The following statements first define the HP-IB Interface's select code and 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 passing control assumes the role of a non-Active Controller on the specified HP-IB Interface. **HP Instrument BASIC programs cannot act as a device when in the role of non-Active controller.** Note that active control of the internal HP-IB interface cannot be passed. The statement "PASS CONTROL 800" actually passes control of the external interface to the instrument. These concepts are discussed next.

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 3589A. 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 3589A. See figure 9-1.

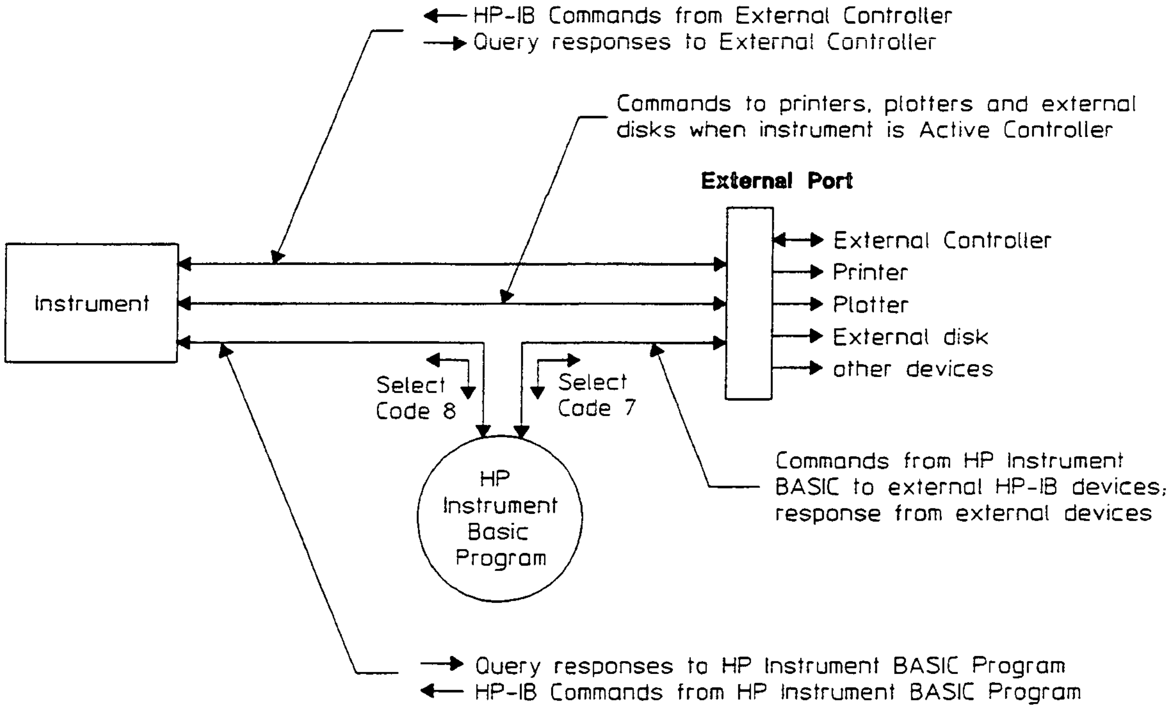

HP-IB Commands from External Controller
Query responses to External Controller
Commands to printers, plotters and external disks when instrument is Active Controller
External Port
External Controller
Printer
Plotter
External disk
other devices
Instrument
Select Code 8
Select Code 7
HP Instrument Basic Program
Commands from HP Instrument BASIC to external HP-IB devices; response from external devices
Query responses to HP Instrument BASIC Program
HP-IB Commands from HP Instrument BASIC Program

Figure 9-1. HP 35665A External and Internal Port

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 3589A's status registers contain information about various analyzer conditions. There are eight register sets. Their reporting structure is summarized in figure 9-2. For more detailed information about the analyzer's register sets, refer to *HP 3589A HP-IB Programmers Reference.*

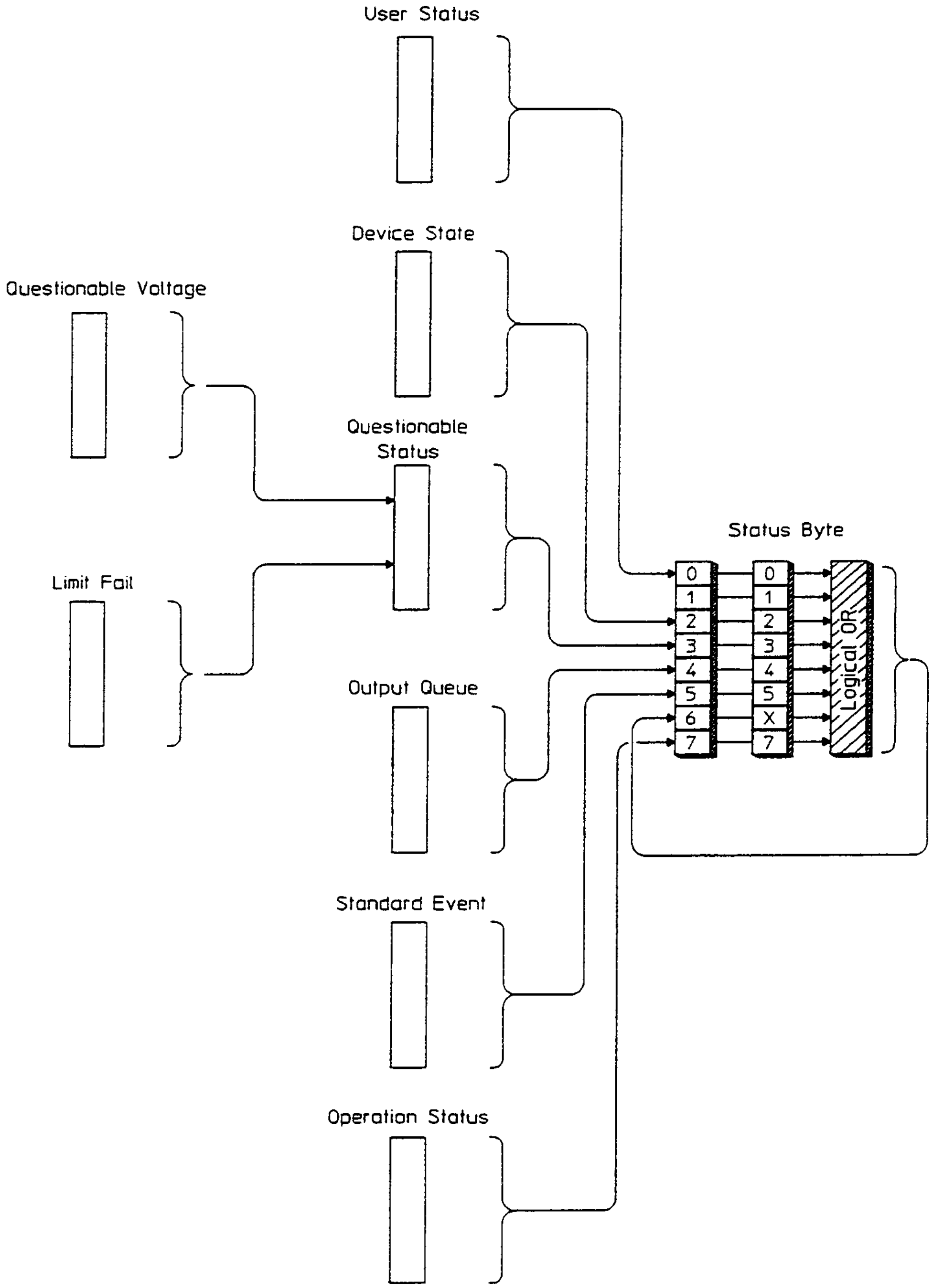


Figure 9-2. HP 35665A Status Registers

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 9-3. 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.

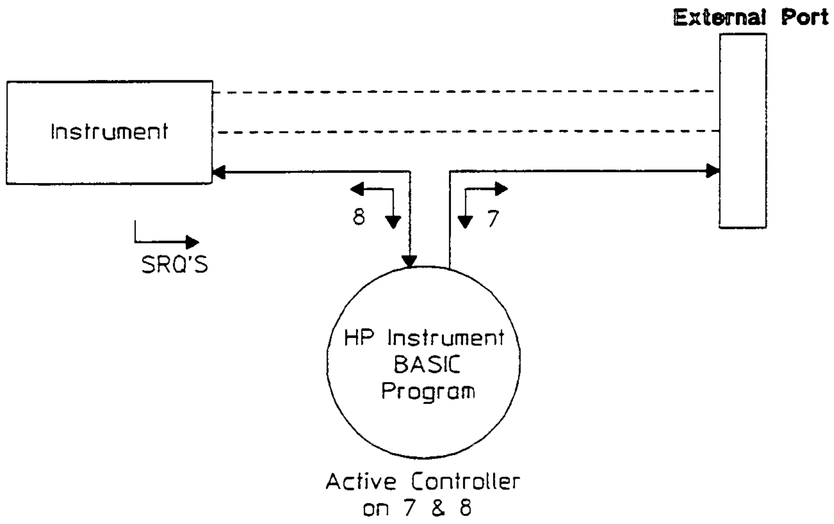


Figure 9-3. The Program as Active Controller on the External Interface

Passing Active Control to the Instrument

The only way that the HP 3589A 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 9-4. 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 9-5.

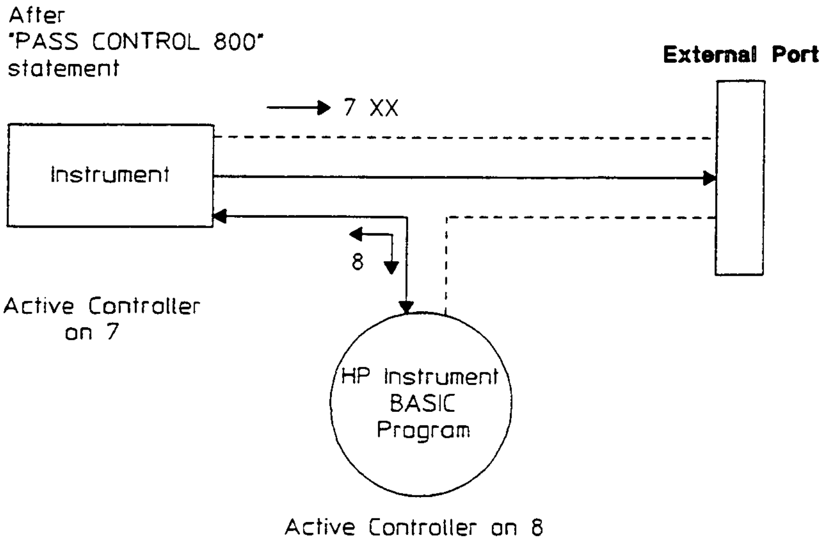


Figure 9-4. Passing Control of the External Interface to the HP 35665A

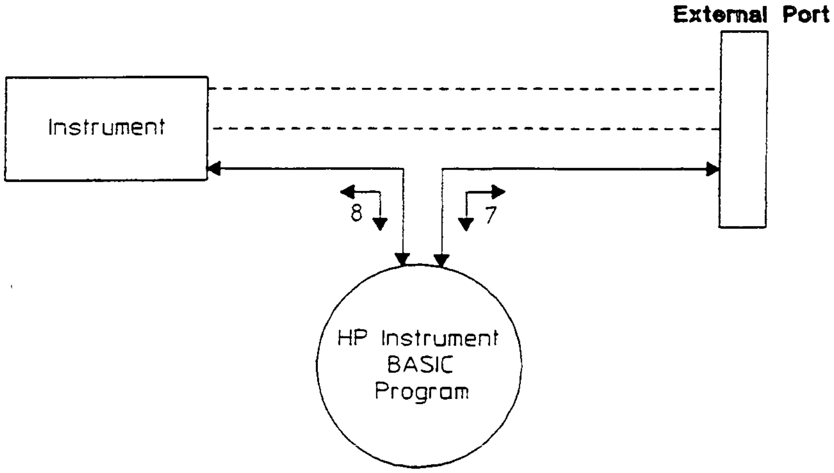


Figure 9-5. Control Passed Back to Program When Instrument Is Done

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 9-6.

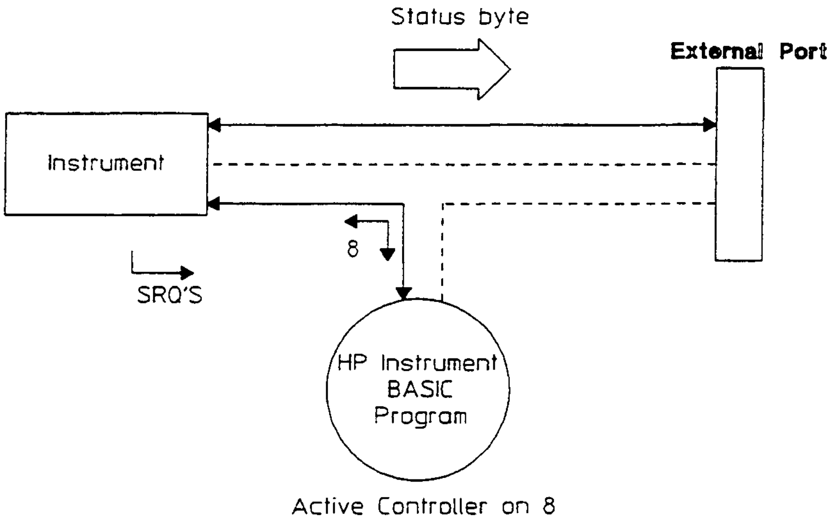


Figure 9-6. The Program as Non-Active Controller

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 3589A 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. 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 3589A 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 3589A HP-IB Programmers Reference.*

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 3589A 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 3589A 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 HP3589A, 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 disc with the program
80   !'DTXFRA' should be in the HP3589A's internal drive.
90    !
100  Scode=7                                !Select code for interface
110  Address=19                             !Address for HP3589A
120   Hp3589a=Scode*100+Address
130  CLEAR Hp3589a
140  OUTPUT Hp3589a;"*CLS"                   !Clear the EVENT registers
150  CLEAR SCREEN                           !Clear the display
160   !
170  DIM Directory$(1:100)[85]              !Array to hold catalog listing
180   !
190  INPUT "Put disk with program 'DTXFRA' into the HP3589A. Press <ENTER>",A$
200  DISP "Loading program on HP3589A..."
210  OUTPUT Hp3589a;"MMEM:GET:PROG 'INT:DTXFRA'" !Load BASIC program from disk
220  OUTPUT Hp3589a;"*OPC?"
230  ENTER Hp3589a;Opc                       !Wait here until program loaded
240  OUTPUT Hp3589a;"*ESR?"                  !Read the EVENT STATUS reg
250  ENTER Hp3589a;Esr
260  IF Esr>0 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 Hp3589a;"*PCB 21"              !Set pass control back address
330  !                                     to HP-IB address for controller
340  DISP "Running the program..."
350  OUTPUT Hp3589a;"PROG:STAT RUN"        !Start the program
360  PASS CONTROL Hp3589a                  !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 Hp3589a
480  DISP ""
490 END
```

```
10   !BASIC program:  DTXFRA -- Data transfer BASIC to BASIC
20    !
30   !This program demonstrates how to transfer data to and from an
40   !external controller.  In this example a catalog listing is transferred
50   !from the HP3589A to the external controller.  For more information
60   !look at the program listing for 'DTXFRB'
70    !
80   !This program is intended to be executed with HP Instrument BASIC
90    !
100  Dsa=800                       !Define IO path to HP3589A
110  DIM Directory$(1:100)[85]     !Create string array for catalog
120   !
130  Host=721                      !Address for external controller
140   !
150  ON ERROR GOTO 150             !Loop until control is passed to the HP3589A
160  ENTER Host;Stor_dev$          !Address Host to talk, read device to catalog
170  OFF ERROR
180   !
190  DISP "Reading catalog..."
200  CAT Stor_dev$ TO Directory$(*)  !Catalog into the string array
210   !
220  DISP "Transferring data..."
230  OUTPUT Host;Directory$(*)        !Address Host to listen, write array
240   !
250  PASS CONTROL Host                !Pass control back to host
260  DISP "DONE"
270  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" mnemonics (and their query counterparts) are part of the HP 3589A 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. Note that if you precede this HP-IB command with the FORMat ASCII command (for example, OUTPUT 719;"FORM ASCII,5") the number will be returned as a readable ASCII number.

The query

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

returns the value of the specified string variable.

Arrays of REAL or INTEGER type 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 mnemonics are run 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 or indefinite length block. The indefinite length block is by far the easiest and 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 mnemonic “PROG:DEF” to download program lines, the “#0” must not be followed by a line-feed. Each program line must then have 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, which runs on an external HP Series 200/300 workstation, demonstrates downloading a short program into the program buffer of the instrument.

```
1  !RE-STORE "DOWNLOAD_PROG"
2   !
3   DIM Load_file$[20],Prog_line$[256]
4   Hp3589a=719
5   ASSIGN @Hp3589a TO Hp3589a
6   !
7   INPUT "ENTER NAME OF FILE TO DOWNLOAD ",Load_file$
8   OUTPUT @Hp3589a;"PROG:DEL"
9   ON ERROR GOTO End_load
10  ASSIGN @File TO Load_file$
11  OUTPUT @Hp3589a;"PROG:DEF #0";
12  LOOP
13    ENTER @File;Prog_line$
14    OUTPUT @Hp3589a;Prog_line$
15  END LOOP
16  !
17 End_load:  !
18  OUTPUT @Hp3589a;CHR$(10) END
19  END
```

Notice that the OUTPUT statements on line 11 is terminated with a semicolon. This suppresses 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 on the CRT 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 will 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 extended command "PROG:DEL."

Uploading

The mnemonic "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 run on an external controller:

```
 10  ! RE-SAVE "UPLOAD88"
 20  ! **********************************************************
 30  ! This program runs on an HP BASIC workstation connected to
 40  ! the HP 3589A with HP Instrument BASIC installed.  The 3589A
 50  ! must have its address set to 719 and must be set up as
 60  ! ADDRESSABLE ONLY on the HP-IB. This program uploads the
 70  ! current program in the HP 3589A's memory to an ASCII file
 80  ! on the workstation's current MSI disk.
 90  ! ***********************************************************
100  ASSIGN @I TO 719
110  DIM A$[256]
120  INPUT "ENTER NAME OF FILE TO UPLOAD PROGRAM TO ",Filename$
130  PRINT Filename$
140  CLEAR @I
150  OUTPUT @I;"PROG:DEF?"
160  ENTER @I USING "#,A,D";A$,Ndigits
170  ENTER @I USING "#,"&VAL$(Ndigits)&"D";Nbytes
180  PRINT Nbytes
190   Openfile(@File,Filename$,Nbytes)
200  ASSIGN @File TO Filename$
210   LOOP
220     ENTER @I;A$
230     EXIT IF LEN(A$)=0
240     PRINT A$
250     OUTPUT @File;A$
260  END LOOP
270  ASSIGN @File TO *
280   STOP
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 to save the uploaded program to. 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
```

Chapter 10

Example Programs

10

Example Programs

This chapter contains listings of some of the example programs on the HP Instrument BASIC disk. Many programs on the disk are not listed here, either because of length or because of similarity to other programs listed. The programs are presented in alphabetical order. Following is a one line synopsis of each of the programs in this chapter:

- AVER_SRQ – demonstrates using the instrument's status registers and SRQs for sensing sweep completions while making a multi-sweep averaged measurement.
- DISCR_SWP – emulates a 20 point discrete sweep with custom user interface.
- LOG_XAXIS – demonstrates dumping, re-scaling and plotting a trace on the HP 3589A CRT
- OPC_SYNC – demonstrates the how to use the *OPC command to allow an SRQ to interrupt program execution.
- OPCQ_SYNC – demonstrates how to use the *OPC? HP-IB command to hang the bus on a query before continuing on with the program.
- PARTITIONS – demonstrates programming of softkeys and partitions.
- PLOT_CTRL – controls setting of rotation, P1 and P2 on HP-GL plotters for report formatting.
- SHAPE – shape factor calculation test.
- SPOOLER – demonstrates fast data collection to RAM disk, then trace recall from disk and printing or plotting of each trace.
- THD – Total Harmonic Distortion (THD) test.
- THREE_DB – determines 3 dB Bandwidth of filter.
- TRC_LOAD – dumps, reverses and reloads non-transform-coordinated trace data.
- TRC_UTILS – dumps and plots transform-coordinated trace data.
- TWO_CTRL – an external program and HP Instrument BASIC program working together.
- WAIT_SYNC – demonstrates how to use the *WAI command to prevent execution of an HP-IB command until all previous commands have finished.

AVER_SRQ

```
10   ! HP Instrument BASIC example program
20   ! -------------------------------------------
30   ! This program demonstrates using the instrument's
40   ! status registers and SRQs for sensing sweep
50   ! completions while making a multi-sweep averaged
60   ! measurement.
70   ! -------------------------------------------
80   !
90   Sc=8
100  Addr=0
110  Device=(Sc*100)+Addr
120  ASSIGN @Hp3589a TO Device
130  !
140  ! Setup registers to detect measurement complete.
150  !
160  OUTPUT @Hp3589a;"*CLS"                ! CLEAR REGISTERS
170  OUTPUT @Hp3589a;"*SRE 128"            ! ENABLE OPERATIONAL STATUS SUMMARY
180  OUTPUT @Hp3589a;"STAT:OPER:ENAB 16"  ! ENABLE MEAS BIT
190  OUTPUT @Hp3589a;"STAT:OPER:PTR 0"    ! ENABLE POS TRANSITIONS
200  OUTPUT @Hp3589a;"STAT:OPER:NTR 16"   ! DISABLE NEG TRANSITIONS
210  !
220  ON INTR Sc GOSUB Check_srq
230  !
240  ! Setup the instrument to take a 10 sweep average, each
250  ! sweep individually armed.  HP-IB trigger (*TRG)
260  ! could also be used.
270  !
280  OUTPUT @Hp3589a;"AVER:COUNT 10"
290  OUTPUT @Hp3589a;"AVER ON"
300  OUTPUT @Hp3589a;"ARM:SOUR MAN"
310  OUTPUT @Hp3589a;"ABORT;INIT"
320  FOR I=1 TO 10
330     OUTPUT @Hp3589a;"ARM:IMM"
340     IF I<10 THEN
350          GOSUB Swp_wait
360          DISP "Hit CONTINUE to take next sweep..."
370          PAUSE
380     END IF
390  NEXT I
400  OUTPUT @Hp3589a;"*OPC?"
410  ENTER @Hp3589a;X
420  DISP "ALL DONE!"
430  STOP
440  !
450 Swp_wait:   !
460   DISP "Sweeping..."
470   ENABLE INTR Sc;255
480  !
490  ! Wait for SRQ
500  !
```

```
510 Hang_out:   !
520   IF NOT Sweep_done THEN
530        GOTO Hang_out
540   END IF
550     Sweep_done=0
560  RETURN
570  !
580 Check_srq:   !
590  !
600    Sb=SPOLL(Device)
610    Send_query(@Hp3589a,"STAT:OPER:EVEN?")
620    Sweep_done=1
630    RETURN
640  !
650  END
660  !**************************************************
670  ! Send a query command and print the return value.
680  !**************************************************
690  SUB Send_query(@Device,Cmd$)
700    OUTPUT @Device;Cmd$
710    ENTER @Device;Resp
720  SUBEND
```

DISCR_SWP

```
1 ! HP Instrument BASIC example program
2 ! -----------------------------------------------------------
3 ! Discrete Sweep Test Program
4 !
5 ! Emulates a 20 point discrete sweep with custom user interface
6 ! -----------------------------------------------------------
7 !
8  REAL Freq(20),Ampl(20)
9  INTEGER Freq_posn(1:20,1:2),Ampl_posn(1:20,1:2)
10 INTEGER Active_entry,I,Command,X,Y
11 INTEGER Cont_mode,Armed,Highlight
12 DIM Entry$[20]
13 DIM Auto_mode$[20],Man_mode$[20],Arm_on$[20],Arm_off$[20]
14 !
15 ASSIGN @Hp3589a TO 800
16 !
17 ! Formatting images
18 !
19 Freq_image1: IMAGE MDD.DDDESZ
20 Freq_image2: IMAGE DDD.DDDESZ
21 Ampl_image: IMAGE MDDD.DDD,"  "
22 !
23 ! Toggling softkey labels
24 !
25 Auto_mode$="STEP MODEON/"&CHR$(129)&"OFF"&CHR$(128)
26 Man_mode$="STEP MODE"&CHR$(129)&"ON"&CHR$(128)&"/OFF"
27 Arm_on$="NEXT PNT"
28 Arm_off$=""
29 !
30 Max_entries=20           ! Must be even number
31 Cont_mode=1              ! Continuous measurement mode
32 Highlight=1              ! Toggle behavior of Highlight_entry sub
33 !
34 GOSUB Setup_inst
35 MASS STORAGE IS ":NVRAM,0,1"
36 ON ERROR GOTO Create_file
37 !
38 ! Delete the following line to maintain entered parameters in a
39 ! non-volatile memory file, i.e. will not lose data entry from
40 ! one program run to the next.
41 !
42 PURGE "FREQ_INFO"
43 !
44 ASSIGN @File TO "FREQ_INFO"
45 GOTO Read_file
46 !
47 Create_file:!
48   OFF ERROR
49   CREATE BDAT "FREQ_INFO",Max_entries+2,8
50   ASSIGN @File TO "FREQ_INFO"
```

```
 51    OUTPUT @File;7.505E+7          ! Default Center
 52    OUTPUT @File;1.499E+8          ! Default Span
 53    FOR I=1 TO Max_entries
 54       OUTPUT @File;5.0E+6*I       ! Default frequencies
 55    NEXT I
 56    ASSIGN @File TO *
 57    ASSIGN @File TO "FREQ_INFO"
 58 !
 59 Read_file:!
 60    OFF ERROR
 61    ENTER @File;Center
 62    ENTER @File;Span
 63    OUTPUT @Hp3589a;"FREQ:CENT ";Center
 64    OUTPUT @Hp3589a;"FREQ:SPAN ";Span
 65    GOSUB Take_sweep
 66    !
 67    FOR I=1 TO Max_entries
 68      ENTER @File;Freq(I)
 69    NEXT I
 70    ASSIGN @File TO *
 71 !
 72 GOSUB Setup_display
 73 !
 74 ON KEY 0 LABEL "NEXT     ENTRY" GOSUB Next_entry
 75 ON KEY 1 LABEL "PREVIOUS ENTRY" GOSUB Prev_entry
 76 ON KEY 2 LABEL "ENTER    FREQUENCY" GOSUB Enter_freq
 77 ON KEY 3 LABEL Auto_mode$ GOSUB Toggle_mode
 78 ON KEY 4 LABEL Arm_off$ GOSUB Arm
 79 ON KEY 6 LABEL "CENTER   FREQUENCY" GOSUB Enter_center
 80 ON KEY 7 LABEL "SPAN" GOSUB Enter_span
 81 ON KEY 9 LABEL "PRINT    RESULTS" GOSUB Print_screen
 82 !
 83 ! Main measurement loop
 84 !
 85 I=1
 86 LOOP
 87 !
 88   LOOP
 89     GOSUB Take_point
 90     EXIT IF Armed OR Cont_mode
 91   END LOOP
 92   Armed=0
 93   !
 94   IF I=Max_entries THEN
 95     I=1
 96   ELSE
 97     I=I+1
 98   END IF
 99 END LOOP
100 !
101 !------------------------------------------------------------
102 Take_point:!  Take one manual frequency data point
103 !
```

```
104   OUTPUT @Hp3589a;"FREQ:MAN ";Freq(I)
105   OUTPUT @Hp3589a;"ABOR; INIT; ARM; *WAI"
106   OUTPUT @Hp3589a;"MARK:AMPL?"
107   ENTER @Hp3589a;Ampl(I)
108   DISABLE
109   OUTPUT Entry$ USING Ampl_image;Ampl(I)
110   PRINT TABXY(Ampl_posn(I,1),Ampl_posn(I,2));Entry$;
111   ENABLE
112 RETURN
113 !-------------------------------------------------------------
114 Next_entry:!  Change active frequency entry to next in table
115 !
116   Highlight=0
117   GOSUB Highlight_entry
118   !
119   IF Active_entry<Max_entries THEN
120     Active_entry=Active_entry+1
121   ELSE
122     Active_entry=1  ! Wrap around
123   END IF
124   !
125   Highlight=1
126   GOSUB Highlight_entry
127 RETURN
128 !-------------------------------------------------------------
129 Prev_entry:!  Change active frequency entry to next in table
130 !
131   Highlight=0
132   GOSUB Highlight_entry
133   !
134   IF Active_entry>1 THEN
135     Active_entry=Active_entry-1
136   ELSE
137     Active_entry=Max_entries
138   END IF
139   !
140   Highlight=1
141   GOSUB Highlight_entry
142 RETURN
143 !-------------------------------------------------------------
144 Toggle_mode:! Change between continuous/single meas modes
145 !
146   Cont_mode=NOT Cont_mode
147   !
148   IF Cont_mode THEN
149     ON KEY 3 LABEL Auto_mode$ GOSUB Toggle_mode
150     ON KEY 4 LABEL Arm_off$ GOSUB Arm
151   ELSE
152     ON KEY 3 LABEL Man_mode$ GOSUB Toggle_mode
153     ON KEY 4 LABEL Arm_on$ GOSUB Arm
154   END IF
155   !
156 RETURN
```

```
157 !----------------------------------------------------
158 Arm:! Go to next measurement point
159 !
160   IF NOT Cont_mode THEN Armed=1
161   BEEP
162 RETURN
163 !----------------------------------------------------
164 Enter_center:!
165   INPUT "Enter Center Freq [Hz] ",Center
166   !
167   IF (Span>1.50E+8) OR (Span<0.) THEN
168     BEEP
169     DISP "Illegal frequency entry, try again."
170     BEEP
171     WAIT 1
172     GOTO Enter_center
173   END IF
174   !
175   OUTPUT @Hp3589a;"FREQ:CENT ";Center
176   GOSUB Take_sweep
177   ASSIGN @File TO "FREQ_INFO"
178   OUTPUT @File,1;Center
179   ASSIGN @File TO *
180 RETURN
181 !----------------------------------------------------
182 Enter_span:!
183   INPUT "Enter Span [Hz] ",Span
184   !
185   IF (Span>1.50E+8) OR (Span<0.) THEN
186     BEEP
187     DISP "Illegal frequency entry, try again."
188     BEEP
189     WAIT 1
190     GOTO Enter_span
191   END IF
192   !
193   OUTPUT @Hp3589a;"FREQ:SPAN ";Span
194   GOSUB Take_sweep
195   ASSIGN @File TO "FREQ_INFO"
196   OUTPUT @File,2;Span
197   ASSIGN @File TO *
198 RETURN
199 !----------------------------------------------------
200 Take_sweep:!
201   OUTPUT @Hp3589a;"DIAG:LOSC SLOOP"
202   OUTPUT @Hp3589a;"SWE:MODE AUTO"
203   OUTPUT @Hp3589a;"REST; ARM; *WAI"
204   OUTPUT @Hp3589a;"SWE:MODE MAN"
205 RETURN
206 !----------------------------------------------------
207 Enter_freq:!
208   INPUT "Enter frequency [Hz] ",Tmp_freq
209   !
```

```
210   IF (Tmp_freq>1.50E+8) OR (Tmp_freq<0.) THEN
211     BEEP
212     DISP "Illegal frequency entry, try again."
213     BEEP
214     WAIT 1
215     GOTO Enter_freq
216   END IF
217   !
218   Freq(Active_entry)=Tmp_freq
219   PRINT TABXY(Freq_posn(Active_entry,1),Freq_posn(Active_entry,2));"     ";
220   GOSUB Highlight_entry
221   ASSIGN @File TO "FREQ_INFO"
222   OUTPUT @File,Active_entry+2;Freq(Active_entry)
223   ASSIGN @File TO *
224 RETURN
225 !-------------------------------------------------------------
226 Print_screen:!
227   OUTPUT @Hp3589a;"PRIN:DUMP:ALL"
228   PASS CONTROL 800
229 RETURN
230 !-------------------------------------------------------------
231 Highlight_entry: !
232 !
233   IF Freq(Active_entry)<1.00E+8 THEN
234     OUTPUT Entry$ USING Freq_image1;Freq(Active_entry)
235   ELSE
236     OUTPUT Entry$ USING Freq_image2;Freq(Active_entry)
237   END IF
238   !
239   ! Highlight or un-highlight entry
240   !
241   IF Highlight THEN
242     Entry$="*"&Entry$
243   ELSE
244     Entry$=" "&Entry$
245   END IF
246  PRINT TABXY(Freq_posn(Active_entry,1)-1,Freq_posn (Active_entry,2));
Entry$;
247 RETURN
248 !-------------------------------------------------------------
249 Setup_display: !
250 !
251   CLEAR SCREEN
252   GCLEAR
253   PRINT
254   PRINT "  Freq [Hz]     Ampl [dBm]     Freq [Hz]     Ampl [dBm]"
255   DISP
256   !
257   ! Draw outline around data area
258   !
259   RESTORE Outline_data
260   LOOP
261     READ Command
```

```
262     EXIT IF Command=-99
263     READ X,Y
264     IF Command=0 THEN     ! MOVE
265        MOVE X,Y
266     ELSE              ! DRAW
267        DRAW X,Y
268     END IF
269   END LOOP
270   !
271   ! Initialize position arrays
272   !
273   RESTORE Posn_data
274   READ Row_offs,Col_offs,Ampl_offs
275    !
276   FOR I=1 TO Max_entries
277     Freq_posn(I,1)=Col_offs
278     Freq_posn(I,2)=Row_offs+I
279     Ampl_posn(I,1)=Col_offs+Ampl_offs
280     Ampl_posn(I,2)=Freq_posn(I,2)
281     !
282     IF Freq(I)<1.00E+8 THEN
283       OUTPUT Entry$ USING Freq_image1;Freq(I)
284     ELSE
285       OUTPUT Entry$ USING Freq_image2;Freq(I)
286     END IF
287     !
288     PRINT TABXY(Col_offs,Freq_posn(I,2));Entry$;
289     IF I=Max_entries/2 THEN READ Row_offs,Col_offs,Ampl_offs
290   NEXT I
291   !
292   OUTPUT @Hp3589a;"DISP:PART LOWER"
293   OUTPUT @Hp3589a;"DISP:PART?"
294   ENTER @Hp3589a;Entry$
295   Active_entry=1
296   GOSUB Highlight_entry
297 RETURN
298 !
299 ! Entry position data
300 Posn_data:!
301   DATA 2,3,15      ! First bank
302   DATA -8,32,15    ! Second bank
303 !
304 ! Result area outline
305 Outline_data:!
306   DATA 0,0,30,1,0,163,1,460,163,1,460,30,1,0,30
307   DATA 0,230,30,1,230,163,0,232,163,1,232,30
308   DATA 0,0,150,1,460,150
309   DATA 0,105,30,1,105,163
310   DATA 0,335,30,1,335,163,-99
311 !
312 !-----------------------------------------------------------
313 Setup_inst:!
314   OUTPUT @Hp3589a;"SYST:PRES"
```

```
315    !
316    ! Remove the following line to prevent instrument from
317    ! measuring internal calibrator signal.
318    !
319    OUTPUT @Hp3589a;"TEST:INP:CONF CAL"
320    !
321    OUTPUT @Hp3589a;"ARM:SOUR MAN"
322    OUTPUT @Hp3589a;"ABOR; INIT; ARM; *WAI"
323    OUTPUT @Hp3589a;"DISP:Y:SCAL:AUTO ONCE"
324    OUTPUT @Hp3589a;"SWE:MODE MAN"
325 RETURN
326 !------------------------------------------------------------
327 END
```

LOG_XAXIS

```
1   ! HP Instrument BASIC example program: LOG_XAXIS
2   ! ------------------------------------------------------------------
3   ! Program to demonstrate dumping, re-scaling and plotting a trace of
4   ! transform-coordinated data on the HP 3589A.
5   !
6   ! ------------------------------------------------------------------
7   !
8   COM REAL X_axis(1:401), Start_freq, Stop_freq
9   DIM Dump_data(1:401)
10  INTEGER Num_pts
11  !
12  ASSIGN @Hp3589a TO 800
13  !
14  GCLEAR
15  CLS
16  OUTPUT @Hp3589a;"DISP:PART LOWER"
17  !
18  LOOP
19    DISP "Dumping trace..."
20    Dump_trace(@Hp3589a,Dump_data(*),Num_pts,"CALC:DATA?")
21    DISP Num_pts;" points dumped."
22    !
23    Log_xaxis(@Hp3589a, Num_pts)  ! Compute log x-axis
24    !
25    OUTPUT @Hp3589a;"ARM:SOUR MAN"
26    DISP "Plotting trace..."
27    Plot_trace(Dump_data(*),X_axis(*),Num_pts)
28    OUTPUT @Hp3589a;"ARM:SOUR IMM"
29    !
30    DISP "Press CONTINUE for next trace"
31    PAUSE
32  END LOOP
33  !
34  END
35  !#####################################################################
36  !
37  SUB Dump_trace(@Hp3589a,REAL Trace_data(*),INTEGER Num_pts,Command$)
38  !--------------------------------------------------------------------
39  ! MODULE DESCRIPTION:
40  !   This module dumps a trace of data from the instrument.
41  !
42  ! INPUTS:    @Hp3558a    : Device selector of instrument.
43  !            Command$    : HP-IB mnemonic used to prompt the instrument
44  !                          for the trace of data.
45  ! OUTPUTS:   Trace_data  : Array of data received from instrument.
46  !            Num_pts     : Number of points dumped into Trace_data.
47  !
48  !--------------------------------------------------------------------
49  DIM A$[10]
50  INTEGER Dig_cnt
```

```
51   CLEAR @Hp3589a
52   OUTPUT @Hp3589a;"FORM:DATA REAL,64"
53   OUTPUT @Hp3589a;Command$
54   ASSIGN @Hp3589a;FORMAT ON
55   Dig_cnt=-1
56   ENTER @Hp3589a USING "%,A,D";A$,Dig_cnt
57   IF (A$<>"#") OR (Dig_cnt<=0) THEN
58     PRINT "NOT CORRECT BLOCK MODE"
59     CLEAR @Hp3589a
60   ELSE
61     ENTER @Hp3589a USING "%,"&VAL$(Dig_cnt)&"D";Num_pts
62     IF (Num_pts MOD 8=0) THEN
63       Num_pts=Num_pts DIV 8
64       ASSIGN @Hp3589a;FORMAT OFF
65       ENTER @Hp3589a;Trace_data(*)
66       ASSIGN @Hp3589a;FORMAT ON
67       ENTER @Hp3589a;A$             ! Read CR/LF
68     ELSE
69       PRINT Data_read;" not float size (divisible by 8)"
70       CLEAR @Hp3589a
71     END IF
72   END IF
73  SUBEND
74  !##########################################################################
75  !
76  SUB Plot_trace(REAL Trace_data(*), Xarray(*), INTEGER Num_pts)
77  !-------------------------------------------------------------------------
78  ! MODULE DESCRIPTION:
79  !   This module plots a trace of data.
80  !
81  ! INPUTS:   Trace_data : Array of data to plot.
82  !           Xarray     : Array of x-axis values (corresponds to
Trace_data)
83  !           Num_pts    : Number of points in Trace_data to plot.
84  !-------------------------------------------------------------------------
85   REAL Y_scaler          ! 1/(Delta per vertical pixel)
86   REAL Max_val,Min_val   ! Maximum and minimum values in data trace
87   REAL Height            ! Display height
88   INTEGER Orig_x,Orig_y  ! Origin (in pixels)
89   !
90   ! Height=340  FULL Partition
91   ! Height=170  HALF Partition
92   Height=170
93   Orig_x=70
94   Orig_y=0
95   !
96   GCLEAR
97   Max_val=MAX(Trace_data(*))
98   Min_val=MIN(Trace_data(*))
99   !
100  ! Perform display auto-scale on data
101  !
102  Y_scaler=Height/(ABS(Max_val-Min_val))
```

```
103  Top_pixel=(Height+Orig_y)
104  MOVE Orig_x,Top_pixel-(ABS(Max_val-Trace_data(1))*Y_scaler)
105  !
106  FOR X=1 TO Num_pts
107    DRAW Orig_x+Xarray(X)-1,Top_pixel-(ABS(Max_val-Trace_data(X))*Y_scaler)
108  NEXT X
109 SUBEND
110 !
111 !#######################################################################
112 !
113 SUB Log_xaxis(@Hp3589a, INTEGER Num_pts)
114 !-----------------------------------------------------------------------
115 ! MODULE DESCRIPTION:
116 !   This module computes a logarithmic x-axis for a linearly spaced
117 !   array of data.
118 !
119 ! INPUTS:   @Hp3558a   : Device selector of instrument.
120 !           Num_pts    : Number of points in X_axis to calculate.
121 ! OUTPUTS:  X_axis     : Array of x-axis values calculated.
122 !
123 !-----------------------------------------------------------------------
124 COM REAL X_axis(1:401), Start_freq, Stop_freq
125 INTEGER I
126 REAL X, K, C, Inc, Temp_start, Temp_stop
127 !
128  Displaywidth = 401
129 !
130   OUTPUT @Hp3589a;"FREQ:STAR?"
131   ENTER @Hp3589a;Temp_Start
132   OUTPUT @Hp3589a;"FREQ:STOP?"
133   ENTER @Hp3589a;Temp_stop
134 !
135 ! Check for changes in frequency coverage
136 !
137 IF (Temp_start <> Start_freq) OR (Temp_stop <> Stop_freq) THEN
138   Start_freq = Temp_start
139   Stop_freq = Temp_stop
140 ELSE
141   GOTO Bail
142 END IF
143 OUTPUT @Hp3589a;"ARM:SOUR MAN"
144 !
145 ! Need to check for ~0 start frequency -> log crash
146 !
147   DISP "Recalculating x-axis scaling"
148 !
149   IF ( Start_freq < 0.000000000001 ) THEN
150     Start_freq = 10.0
151   END IF
152   Inc=(Stop_freq - Start_freq) / (Num_pts-1)
153   K=Displaywidth / (LGT(Stop_freq) - LGT(Start_freq))
154   C=LGT(Start_freq) * (-K)
155   I=1
```

```
156   X=Start_freq
157   LOOP
158     EXIT IF ((X>Stop_freq) OR (I>Num_pts))
159     X_axis(I) = C + K * LGT(X)
160     I=I+1
161     X=X+Inc
162   END LOOP
163 !
164 OUTPUT @Hp3589a;"ARM:SOUR IMM"
165 Bail:SUBEND
```

OPC_SYNC

```
10    ! HP Instrument BASIC program: OPC_SYNC - Measurement synchronization
20    ! ------------------------------------------------------------------
30    ! This program demonstrates the 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.
90    !
100   Scode=8                           ! Interface select code
110   Address=00
120   Hp3589a=Scode*100+Address
130   !
140   OUTPUT Hp3589a;"SWE:TIME 8"        ! Set record length to 8 seconds
150   OUTPUT Hp3589a;"*CLS"              ! Clear the STATUS BYTE register
160   OUTPUT Hp3589a;"*ESE 1"            !Program the EVENT STATUS ENABLE reg.
170   OUTPUT Hp3589a;"*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 Hp3589a;"ABORT; INIT"      !Start the measurement
230   OUTPUT Hp3589a;"*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(Hp3589a)                 !Read STATUS BYTE and clear SRQ
330     BEEP
340     OUTPUT Hp3589a;"*ESR?"             !Read and clear EVENT STATUS reg.
350     ENTER Hp3589a;Esr
360     DISP "Got the SRQ! SPOLL returns:";Stb;"   ESR returns:";Esr
370   END
```

OPCQ_SYNC

```
10    ! HP Instrument BASIC program: OPCQ_SYNC -  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 HP3589A will return a '1' in response to *OPC?.
70    !
80    Scode=8
90    Hp3589a=Scode*100
100   !
110   OUTPUT Hp3589a;"*RST"           !Preset the HP3589A
120   OUTPUT Hp3589a;"*OPC?"          !Pause on ENTER statement until
130   ENTER Hp3589a;Opc               !'*RST' command has finished
140   !
150   OUTPUT Hp3589a;"SWE:TIME 8"     !Set record length to 8 seconds
160   DISP "Measurement started ..."
170   OUTPUT Hp3589a;"ABOR; INIT"     !Start the measurement
180   OUTPUT Hp3589a;"*OPC?"          !Pause until all pending HP-IB commands
190   ENTER Hp3589a;Opc               !have finished.
200   BEEP
210   DISP "Measurement done"
220   END
```

PARTITIONS

```
10    ! HP Instrument BASIC example program
20    !  ----------------------------------------------------------
30    !  File name: PARTITIONS
40    !
60    !  Description:
70    !        Demonstration of softkeys and partitions
80    !  -----------------------------------------------------------
90    !   set up error trap
100   ON ERROR GOTO Error_trap
110   !
120   !   set up HP 3589A timeout trap
130   ON TIMEOUT 8,2 GOTO Timeout_trap
140   !
150   !   initialize variables
160   !
170   INTEGER Err_status
180   !
190   !
200   ! ******* MAIN PROGRAM ********
210   !
220   PRINTER IS CRT
230   OUTPUT 800;"SCR:FORM ULOW"
240   !
250   ON KEY 1 LABEL "OFF   " GOSUB Part0
260   ON KEY 2 LABEL "UPPER" GOSUB Part1
270   ON KEY 3 LABEL "LOWER" GOSUB Part2
280   ON KEY 4 LABEL "FULL" GOSUB Part3
290   ON KEY 9 LABEL "QUIT" GOTO Quit
300   !
310 Waiting:GOTO Waiting
320   !
330   !
340 Part0:  !
350   !
360   Change_part(0)
370   DISP "ALL PARTITIONS ARE OFF"
380   IF Err_status=-1 THEN GOSUB Partition_error
390   RETURN
400   !
410 Part1:  !
420   Change_part(1)
430   IF Err_status=-1 THEN
440     GOSUB Partition_error
450     SUBEXIT
460   END IF
470   Draw_box(475,173)
480   PRINT TABXY(0,14),"(0,0)";
490   PRINT TABXY(23,6),"UPPER PARTITION"
500   PRINT TABXY(50,1),"(475,173)";
510   RETURN
```

```
520   !
530 Part2:  !
540   Change_part(2)
550   IF Err_status=-1 THEN
560     GOSUB Partition_error
570     SUBEXIT
580   END IF
590   Draw_box(475,173)
600   PRINT TABXY(0,14),"(0,0)";
610   PRINT TABXY(23,6),"LOWER PARTITION";
620   PRINT TABXY(50,1),"(475,173)";
630   RETURN
640   !
650 Part3:  !
660   Change_part(3)
670   IF Err_status=-1 THEN
680     GOSUB Partition_error
690     SUBEXIT
700   END IF
710   Draw_box(475,355)
720   PRINT TABXY(0,29),"(0,0)";
730   PRINT TABXY(23,14),"FULL PARTITION"
740   PRINT TABXY(50,1),"(475,355)";
750   RETURN
760   !
770 Partition_error: !
780   BEEP
790   DISP "Unable to allocate partition"
800   Err_status=0 ! clear status
810   RETURN
820  !
830  !
840 Quit:   !
850   Change_part(0)
860   DISP "THE END"
870   STOP
880   !
890 Timeout_trap:  !
900   BEEP
910   DISP "Test failed, due to timeout on 8"
920   WAIT 2
930   STOP
940   !
950 Error_trap: !
960   BEEP
970   DISP "Test failed, to to unexpected error"
980   WAIT 2
990   DISP ERRM$
1000   WAIT 2
1010   END
1020   !
1030   !
1040   SUB Change_part(Part)
```

```
1050   !-------------------------------------------------
1060   ! Changes the partition to the number passed in the
1070   ! variable Part and returns a status number
1080   !-------------------------------------------------
1090  CLEAR SCREEN
1100  GCLEAR
1110  DISP " "
1120  OUTPUT 800;"DISP:PART 0" ! deallocate all partitions first
1130  OUTPUT 800;"DISP:PART?"
1140  ENTER 800;A$               ! wait for partitions to be cleared
1150  SELECT Part
1160  CASE 1
1170    OUTPUT 800;"DISP:PART UPPER"
1180  CASE 2
1190    OUTPUT 800;"DISP:PART LOWER"
1200  CASE 3
1210    OUTPUT 800;"DISP:PART FULL"
1220  END SELECT
1230  SUBEND
1240  !
1250  SUB Draw_box(X,Y)
1260   !-------------------------------------------------
1270   ! Draw a box with the lower left corner at 0,0 and
1280   ! the upper right corner at X,Y
1290   !-------------------------------------------------
1300  MOVE 0,0
1310  DRAW 0,Y
1320  DRAW X,Y
1330  DRAW X,0
1340  DRAW 0,0
1350  SUBEND
```

PLOT_CTRL

```
10    ! HP Instrument BASIC example program
20    ! ------------------------------------------------------------------
30    !
40    ! Plot controller
50    !
60    ! Controls setting of rotation, P1 and P2 on HP-GL plotters for report
70    ! formatting.
80    !
90    ! For HP-GL plotters (at least 7475A) :
100   !
110   !   No rotation gives landscape format.
120   !   P1 specifies the lower left corner of the plot in (X,Y) coords
130   !   P2 specifies the upper right corner of the plot in (X,Y) coords
140   !
150   ! For 7475A :
160   !
170   !   1. 4 per page landscape :
180   !
190   !      * Rotation = 0 degrees
200   !      * Upper left quadrant   P1(1000,4500) P2(4700,7600)
210   !      * Upper right quadrant  P1(6000,4500) P2(9700,7600)
220   !      * Lower left quadrant   P1(1000,500)  P2(4700,3600)
230   !      * Lower right quadrant  P1(6000,500)  P2(9700,3600)
240   !
250   !   2. 2 per page portrait (side by side) :
260   !
270   !      * Rotation = 90 degrees
280   !      * Upper left  P1(100,7000) P2(4000,10000)
290   !      * Upper right P1(4500,7000) P2(7900,10000)
300   !
310   ! ------------------------------------------------------------------
320   !
330   INTEGER Rotate,P1_x,P1_y,P2_x,P2_y,Plot_done
340   ASSIGN @Hp3589a TO 800
350   !
360   LOOP
370     DISP "Setup next screen to plot, then press CONTINUE."
380     PAUSE
390     !
400     ! Prompt user for rotation of plot.
410     !
420     INPUT "Enter rotation (degrees) : ",Rotate
430     OUTPUT 705;"RO ";Rotate;";" ! Send rotation to plotter
440     !
450     ! Prompt user for P1 and P2 for plot.
460     !
470     INPUT "Enter P1X, P1Y, P2X, P2Y : ",P1_x,P1_y,P2_x,P2_y
480     OUTPUT 705;"IP ";P1_x,P1_y,P2_x,P2_y;";"
490     !
500     OUTPUT @Hp3589a;"PLOT:DUMP:ALL"  ! Plot screen
```

```
510    CALL Passcntrl
520  END LOOP
530  !
540  END
550  !
560  !--------------------------------------------
570  ! Passes control to the analyzer and waits
580  ! for control to to be passed back.
590  !--------------------------------------------
600  SUB Passcntrl
610    PASS CONTROL 800
620    ON ERROR GOTO Not_done
630 Not_done:  !
640    WAIT 1
650    CLEAR 7
660    OFF ERROR
670  SUBEND
```

SHAPE

```
1 ! HP Instrument BASIC example program
2 ! ---------------------------------
3 ! Shape factor calculation test
4 ! ---------------------------------
5 !
6 REAL Meas_span,Peak_freq,Search_freq,Freq_per_bin
7 REAL Target_ampl,Left_freq,Right_freq
8 REAL Sixty_db_bw
9 INTEGER Slope_up
10 !
11 Shape_image:IMAGE "Shape factor = ",D.DD
12 !
13 ASSIGN @Hp3589a TO 800
14 !
15 ! Recall "3dB" State
16 !
17 OUTPUT @Hp3589a;"SYST:PRES"
18 !
19 Pass=1
20 LOOP
21   EXIT IF Pass>2
22   IF Pass=1 THEN
23     GOSUB Setup_3db
24   ELSE
25     GOSUB Setup_60db
26   END IF
27   DISP CHR$(129)&"Setting up filter measurement"&CHR$(128)
28   OUTPUT @Hp3589a;"REST; *WAI"     ! Take averaged measurement
29   OUTPUT @Hp3589a;"ARM:SOUR MAN"   ! Stop measurement
30   !
31   OUTPUT @Hp3589a;"DISP1:Y:SCAL:AUTO ONCE"
32   OUTPUT @Hp3589a;"FREQ:SPAN?"
33   ENTER @Hp3589a;Meas_span
34   Freq_per_bin=Meas_span/400.
35   OUTPUT @Hp3589a;"MARK:MAX:GLOB"
36   OUTPUT @Hp3589a;"MARK:FREQ?"
37   ENTER @Hp3589a;Peak_freq
38   !
39   OUTPUT @Hp3589a;"MARK:AMPL?"
40   ENTER @Hp3589a;Target_ampl
41   !
42   IF Pass=1 THEN
43     Target_ampl=Target_ampl-3.    ! (peak - 3 dB)
44     DISP CHR$(129)&"Calculating -3.0 dB Bandwidth"&CHR$(128)
45   ELSE
46     Target_ampl=Target_ampl-60.  ! (peak - 60 dB)
47     DISP CHR$(129)&"Calculating -60.0 dB Bandwidth"&CHR$(128)
48   END IF
49   !
50   ! Search for left target point
```

```
51    !
52    Right_freq=Peak_freq
53    Left_freq=Right_freq-(Meas_span/2.)
54    Slope_up=1
55    GOSUB Bin_search
56    OUTPUT @Hp3589a;"MARK:OFFS ON; OFFS:DELT:X 0; Y 0"
57    !
58    ! Search for right target point
59    !
60    Target_ampl=0.  ! Relative to OFFSET MARKER
61    Right_freq=Peak_freq+(Meas_span/2.)
62    Left_freq=Peak_freq
63    Slope_up=0
64    GOSUB Bin_search
65    !
66    OUTPUT @Hp3589a;"MARK:OFFS:DELT:X?"
67    IF Pass=1 THEN
68      ENTER @Hp3589a;Three_db_bw
69    ELSE
70      ENTER @Hp3589a;Sixty_db_bw
71    END IF
72    !
73    Pass=Pass+1
74    OUTPUT @Hp3589a;"MARK:OFFS OFF"
75  END LOOP
76  !
77  BEEP
78  DISP USING Shape_image;Sixty_db_bw/Three_db_bw
79  !
80  OUTPUT @Hp3589a;"ARM:SOUR IMM"  ! Re-start measurement
81  STOP
82  !
83  Bin_search: !
84  !
85  ! Look for Target_ampl +- 0.01 dB
86  !
87  ! Left_freq and Right_freq bound area of search
88  !
89  Search_freq=Left_freq+(Meas_span/4.)
90  !
91  LOOP
92    OUTPUT @Hp3589a;"MARK:X ";Search_freq
93    IF Slope_up=0 THEN
94      OUTPUT @Hp3589a;"MARK:OFFS:DELT:Y?"  ! Searching for right target
95    ELSE
96      OUTPUT @Hp3589a;"MARK:Y?"
97    END IF
98    ENTER @Hp3589a;Search_ampl
99    EXIT IF (ABS(Target_ampl-Search_ampl)<=.01) OR
(ABS(Left_freq-Right_freq) <=Freq_per_bin)
100    !
101    IF Slope_up THEN
102      IF (Search_ampl>Target_ampl) THEN
```

```
103       Right_freq=Search_freq
104     ELSE
105       Left_freq=Search_freq
106     END IF
107   ELSE
108     IF (Search_ampl>Target_ampl) THEN
109       Left_freq=Search_freq
110     ELSE
111       Right_freq=Search_freq
112     END IF
113   END IF
114   !
115   Search_freq=Left_freq+(Right_freq-Left_freq)/2.
116 END LOOP
117 RETURN
118 !
119 ! Setup for 3 dB measurement
120 Setup_3db:!
121   OUTPUT @Hp3589a;"TEST:INP:CONF CAL"
122   OUTPUT @Hp3589a;"FREQ:CENT 10 MHz"
123   OUTPUT @Hp3589a;"BAND:AUTO OFF"
124   OUTPUT @Hp3589a;"FREQ:SPAN 2 kHz"
125   OUTPUT @Hp3589a;"BAND:RES 580 Hz"
126 RETURN
127 !
128 ! Setup for 60 dB measurement
129 Setup_60db:!
130   OUTPUT @Hp3589a;"ARM:SOUR IMM"  ! Re-start measurement
131   OUTPUT @Hp3589a;"FREQ:SPAN 5 kHz"
132 RETURN
133 END
```

SPOOLER

```
1    ! HP Instrument BASIC example program: SPOOLER
2    ! ------------------------------------------------------------------
3    !
4    ! Print/plot spooler
5    !
6    ! Demonstrates :
7    !
8    !   ** Fast data collection to RAM disk.
9    !   ** Post collection trace recall from disk to display.
10   !   ** Printing/plotting each collected/displayed trace from IBASIC.
11   ! ------------------------------------------------------------------
12   COM / Inst_addr / @Hp3589a
13   !
14   DIM Catalog$(1:650)[80]     ! Catalog string array
15   DIM Msus$[10], IB_msus$[10]
16   INTEGER I
17   !
18   Msus$="'RAM:'"          ! Select default disk to use as spooler
19   IB_msus$=":MEMORY,0"    ! IBASIC msus needs to be set to same device
20   !
21   ! Initialize instrument
22   !
23   ASSIGN @Hp3589a TO 800
24   OUTPUT @Hp3589a;"SYST:PRES"        ! PRESET instrument
25   OUTPUT @Hp3589a;"MMEM:MSI "&Msus$  ! Choose spooler disk as default
26   MSI IB_msus$
27   INITIALIZE IB_msus$,3              ! Format to largest possible
28   !
29   CALL Fill_disk  ! Fill up disk with traces
30   !
31   ! Recall all trace files from disk and print/plot
32   !
33   CAT IB_msus$ TO Catalog$(*)  ! Get list of files on disk
34   I=8                          ! Start of file entries in CAT array
35   LOOP
36     EXIT IF Catalog$(I) = ""    ! At last file entry
37     ENTER Catalog$(I) USING "10A";File_name$
38     Recall_file( File_name$ )  ! Recall the trace
39     !
40     ! Print/plot the file
41     !
42     OUTPUT @Hp3589a;"PRIN:DUMP:ALL"  ! Print screen
43     ! OUTPUT @Hp3589a;"PLOT:DUMP:ALL"  ! Plot screen
44     CALL Passcntrl
45     I=I+1
46   END LOOP
47   !
48   PAUSE
49   !
50   !
```

```
51   END
52   !
53   !**********************************************
54   ! Verifies that a file contains a trace before
55   ! recalling it to the display.
56   !**********************************************
57   SUB Recall_file( File_name$ )
58     COM / Inst_addr / @Hp3589a
59     INTEGER File_kind
60     !
61     OUTPUT @Hp3589a;"DISP1:RES D1" ! Display data register 1
62     !
63     ! Open file and verify that it contains a trace
64     !
65     ASSIGN @File to File_name$
66     ENTER @File;File_kind
67     ASSIGN @File to *
68     !
69     IF ( File_kind = 1500 ) THEN
70       ! Recall trace into data register 1
71       !
72       OUTPUT @Hp3589a;"MMEM:LOAD:TRAC D1,'"&File_name$&"'"
73       OUTPUT @Hp3589a;"TRAC1:TITL '"&TRIM$(File_name$)&"'"
74       !
75       ! Wait for recall completion
76       OUTPUT @Hp3589a;"FREQ:CENT?"
77       ENTER @Hp3589a;A
78     ELSE
79       PRINT File_name$&" does not contain a trace file."
80     END IF
81   SUBEND
82   !
83   !**********************************************
84   ! Passes control to the analyzer and waits
85   ! for control to to be passed back.
86   !**********************************************
87   SUB Passcntrl
88     PASS CONTROL 800
89     ON ERROR GOTO Not_done
90   Not_done: !
91     WAIT 1
92     CLEAR 7
93     OFF ERROR
94   SUBEND
95   !
96   !*********************************************************************
97   ! Function to determine amount of available space on msu in units
98   ! of LIF blocks (256 bytes).
99   !*********************************************************************
100   !
101   DEF FNData( Msus$ )
102     DIM Catalog$(1:4)[80]
103     INTEGER Number
```

```
104     CAT Msus$ TO Catalog$(*)
105     ENTER Catalog$(4) USING "16X,K";Number
106     RETURN Number
107   FNEND
108   !
109   !*****************************************************************
110   ! Fills up disk with different data traces.  Set up instrument
111   ! to NBZ, cover each 40kHz span from DC until RAM disk is full.
112   !*****************************************************************
113   SUB Fill_disk
114     COM / Inst_addr / @Hp3589a
115     DIM Err_resp$[256]
116     !
117     Data_space=FNData(IB_msus$) ! Determine data space available on disk
118     Trace_size=9                ! Size of trace file in LIF sectors
119     Trace_num=1
120     !
121     ! Clear error queue
122     !
123     OUTPUT @Hp3589a;"SYST:ERR?"
124     ENTER @Hp3589a;Err_no,Err_resp$
125     !
126     WHILE ( Err_no )
127       OUTPUT @Hp3589a;"SYST:ERR?"
128       ENTER @Hp3589a;Err_no,Err_resp$
129     END WHILE
130     !
131     OUTPUT @Hp3589a;"INST:SEL FFT"
132     OUTPUT @Hp3589a;"FREQ:CENT 20 kHz"
133     OUTPUT @Hp3589a;"ARM:SOUR MAN"
134     OUTPUT @Hp3589a;"FREQ:STEP:STAT MAN"
135     OUTPUT @Hp3589a;"FREQ:STEP 40 kHz"
136     !
137     LOOP
138       EXIT IF (Data_space < Trace_size) OR (Err_no)
139       OUTPUT @Hp3589a;"ABOR; INIT; ARM; *WAI"
140       !
141       ! DISP "Saving 'TRACE"&VAL$(Trace_num)&"'"
142       OUTPUT @Hp3589a;"MMEM:STOR:TRAC A,'TRACE"&VAL$(Trace_num)&"'"
143       OUTPUT @Hp3589a;"SYST:ERR?"
144       ENTER @Hp3589a;Err_no,Err_resp$
145       !
146       ! Wait for save completion
147       OUTPUT @Hp3589a;"FREQ:CENT?"
148       ENTER @Hp3589a;A
149       !
150       OUTPUT @Hp3589a;"FREQ:CENT UP"
151       Data_space=Data_space-Trace_size
152       Trace_num=Trace_num+1
153     END LOOP
154   SUBEND
```

THD

```
10  ! HP Instrument BASIC example program
20  ! ---------------------------------
30  ! Total Harmonic Distortion (THD) test
40  ! ---------------------------------
50  !
60  COM /Dut/ @Hp3589a
70  DIM Prompt$[60]
80  !
90 Fund_image: IMAGE "     Fundamental : ",K," Hz, ", 3D.2D, " dBm"
100 Thd_image:IMAGE  "     THD         : ", 2D.3D,"%  (", 3D.2D, " dB )"
110 !
120 ASSIGN @Hp3589a TO 800
130 !
140 Prompt$="Move marker past last harmonic, then press MEASURE THD."
150 !
160 Start: !
170 !
180 !OUTPUT @Hp3589a;"SYST:PRES"
190 !
200 CLEAR SCREEN
210 GCLEAR
220 DISP Prompt$
230 OUTPUT @Hp3589a;"DISP:PART LOWER"
240 OUTPUT @Hp3589a;"SYST:RPGLOCK OFF"
250 !
260 ON KEY 1 LABEL "MEASURE    THD" GOSUB Measure
270 !
280 Hang_out: GOTO Hang_out
290 !
300 Measure:      !
310 CLEAR SCREEN
320 !
330 ! MEASURE LAST HARMONIC FREQUENCY LIMIT
340 !
350 OUTPUT @Hp3589a;"MARK:X?"
360 ENTER @Hp3589a;Last_x
370 !
380 ! MEASURE FUNDAMENTAL FREQUENCY
390 !
400 DISP "Frequency counting fundamental."
410 !
420 OUTPUT @Hp3589a;"ARM:SOUR MAN"
430 OUTPUT @Hp3589a;"MARK:MAX:GLOB"
440 OUTPUT @Hp3589a;"MARK:FUNC:FCO ON"
450 OUTPUT @Hp3589a;"REST; :ARM; *WAI"
460 OUTPUT @Hp3589a;"MARK:OFFS ON; OFFS:DELT:X 0; Y 0"
470 OUTPUT @Hp3589a;"MARK:X:FCO?"
480 ENTER @Hp3589a;Fund_x
490 OUTPUT @Hp3589a;"MARK:Y?"
500 ENTER @Hp3589a;Fund_y
```

```
510 OUTPUT @Hp3589a;"MARK:FUNC:FCO OFF"
520 PRINT
530 PRINT
540 PRINT USING Fund_image;Fund_x,Fund_y
550 !
560 DISP "Manually sweeping harmonics."
570 !
580 OUTPUT @Hp3589a;"SWE:MODE MAN"
590 !
600 Thd=0
610 Harm_x=2*Fund_x
620 !
630 LOOP
640   EXIT IF Harm_x>Last_x
650   OUTPUT @Hp3589a;"FREQ:MAN";Harm_x;"HZ"
660   OUTPUT @Hp3589a;"REST; :ARM; *WAI"
670   OUTPUT @Hp3589a;"MARK:X";Harm_x;"Hz"
680   OUTPUT @Hp3589a;"*WAI"
690   OUTPUT @Hp3589a;"MARK:OFFS:DELT:Y?"
700   ENTER @Hp3589a;Y
710   Thd=Thd+10.^(Y/10.)
720   Harm_x=Harm_x+Fund_x
730 END LOOP
740 !
750 Db=10.*LGT(Thd)
760 Per=100.*10.^(Db/20.)
770 PRINT
780 PRINT USING Thd_image;Per,Db
790 !
800 OUTPUT @Hp3589a;"SWE:MODE AUTO"
810 OUTPUT @Hp3589a;"ARM:SOUR IMM"
820 OUTPUT @Hp3589a;"MARK:OFFS OFF"
830 DISP Prompt$
840 RETURN
850 END
```

THREE_DB

```
1    ! HP Instrument BASIC example program
2    ! --------------------------------
3    ! Determines 3 dB Bandwidth of filter.
4    !--------------------------------
5    REAL Start_freq, Stop_freq, Ampl, Off_by
6    REAL Left_freq, Right_freq
7    !
8    ASSIGN @Hp3589a TO 800
9    !
10   Start:!
11   DISP "Choose method for 3 db calculation"
12   BEEP
13   !
14   Main_menu:!
15   !
16   DISABLE
17   ON KEY 0 LABEL "Limit    Line"    GOTO Lim_3db
18   ON KEY 1 LABEL "Marker   Search"  GOTO Mkr`3db
19   ON KEY 9 LABEL "Pause    Program" GOTO Pause_prog
20   ENABLE
21   !
22   Hang_out: GOTO Hang_out
23   !
24   Lim_3db:!
25   GOSUB Blank_keys
26   DISABLE
27   DISP "Defining limit line"
28   !
29   ! Determine active trace and get start and stop frequency
30   !
31   OUTPUT @Hp3589a;"FREQ:STAR?"
32   ENTER @Hp3589a;Start_freq
33   OUTPUT @Hp3589a;"FREQ:STOP?"
34   ENTER @Hp3589a;Stop_freq
35   !
36   ! Get peak value
37   !
38   OUTPUT @Hp3589a;"MARK1:MAX:GLOB"
39   OUTPUT @Hp3589a;"MARK1:Y?"
40   ENTER @Hp3589a;Ampl
41   !
42   ! Define limit line at -3dB from peak value.
43   !
44   Ampl=Ampl-3
45   OUTPUT @Hp3589a;"DISP1:LIM:UPP:DEL"
46   OUTPUT @Hp3589a;"DISP1:LIM:UPP:SEGMENT ";Start_freq,Ampl,Stop_freq,Ampl
47   OUTPUT @Hp3589a;"DISP1:LIM:LINE ON"
48   OUTPUT @Hp3589a;"DISP1:LIM:STAT ON"
49   !
50   ! Stop measurement for result calculation
```

```
51  !
52  OUTPUT @Hp3589a;"ARM:SOUR MAN"
53  OUTPUT @Hp3589a;"REST;ARM;*WAI"
54  !
55  ! Read limit failures to determine left and right 3 dB frequencies
56  !
57  DISP "Scanning for left & right limit failures"
58  Read_failures( @Hp3589a, Left_freq, Right_freq, "DISP1:LIM:UPP:REPORT?" )
59  !
60  IF (Left_freq < Start_freq) OR (Left_freq > Stop_freq) OR (Right_freq <
Start_freq) OR (Right_freq > Stop_freq) THEN
61    DISP "Invalid measurement setup."
62    BEEP
63    STOP
64  END IF
65  !
66  DISP "3 dB Bandwidth = ";Right_freq-Left_freq;" Hz"
67  !
68  ! Restart measurement, turn off limits
69  !
70  OUTPUT @Hp3589a;"ARM:SOUR IMM"
71  OUTPUT @Hp3589a;"DISP1:LIM:LINE OFF"
72  OUTPUT @Hp3589a;"DISP1:LIM:STAT OFF"
73  GOTO Main_menu
74  !
75  Mkr_3db:!
76  GOSUB Blank_keys
77  DISABLE
78  !
79  ! Stop measurement for marker search
80  !
81  OUTPUT @Hp3589a;"ARM:SOUR MAN"
82  OUTPUT @Hp3589a;"REST;ARM;*WAI"
83  !
84  OUTPUT @Hp3589a;"FREQ:SPAN?"
85  ENTER @Hp3589a;Meas_span
86  Freq_per_bin=Meas_span/400.
87  OUTPUT @Hp3589a;"MARK:MAX:GLOB"
88  OUTPUT @Hp3589a;"MARK:FREQ?"
89  ENTER @Hp3589a;Peak_freq
90  !
91  OUTPUT @Hp3589a;"MARK:AMPL?"
92  ENTER @Hp3589a;Target_ampl
93  !
94  Target_ampl=Target_ampl-3.   ! (peak - 3 dB)
95  DISP "Calculating -3.0 dB Bandwidth"
96  !
97  !   Search for left target point
98  !
99  Right_freq=Peak_freq
100  Left_freq=Right_freq-(Meas_span/2.)
101  Slope_up=1
102  GOSUB Bin_search
```

```
103  OUTPUT @Hp3589a;"MARK:OFFS ON; OFFS:DELT:X 0; Y 0"
104  !
105  ! Search for right target point
106  !
107  Target_ampl=0.  ! Relative to OFFSET MARKER
108  Right_freq=Peak_freq+(Meas_span/2.)
109  Left_freq=Peak_freq
110  Slope_up=0
111  GOSUB Bin_search
112  !
113  OUTPUT @Hp3589a;"MARK:OFFS:DELT:X?"
114  ENTER @Hp3589a;Three_db_bw
115  DISP "3 dB Bandwidth = ";Three_db_bw;" Hz"
116  !
117  !
118  ! Restart measurement, turn off offset marker
119  !
120  OUTPUT @Hp3589a;"ARM:SOUR IMM"
121  OUTPUT @Hp3589a;"MARK:OFFS OFF"
122  GOTO Main_menu
123  !
124  !
125  Bin_search: !
126  !
127  ! Look for Target_ampl +- 0.001 dB
128  !
129  ! Left_freq and Right_freq bound area of search
130  !
131  Search_freq=Left_freq+(Meas_span/4.)
132  !
133  LOOP
134  OUTPUT @Hp3589a;"MARK:X ";Search_freq
135  IF Slope_up = 0 THEN
136    OUTPUT @Hp3589a;"MARK:OFFS:DELT:Y?"  ! Searching for right target
137  ELSE
138    OUTPUT @Hp3589a;"MARK:Y?"
139  END IF
140  ENTER @Hp3589a;Search_ampl
141  EXIT IF (ABS(Target_ampl-Search_ampl)<=.001) OR
(ABS(Left_freq-Right_freq)<=Freq_per_bin)
142  !
143  IF Slope_up THEN
144    IF (Search_ampl>Target_ampl) THEN
145      Right_freq=Search_freq
146    ELSE
147      Left_freq=Search_freq
148    END IF
149  ELSE
150    IF (Search_ampl>Target_ampl) THEN
151      Left_freq=Search_freq
152    ELSE
153      Right_freq=Search_freq
154    END IF
```

```
155  END IF
156  !
157  Search_freq=Left_freq+(Right_freq-Left_freq)/2.
158  END LOOP
159  RETURN
160  !
161  Pause_prog:!
162  DISP "PROGRAM PAUSED"
163  BEEP
164  PAUSE
165  GOTO Start
166  !
167  Blank_keys:!
168  OFF KEY 0
169  OFF KEY 1
170  OFF KEY 1
171  OFF KEY 2
172  OFF KEY 3
173  OFF KEY 4
174  OFF KEY 5
175  OFF KEY 6
176  OFF KEY 7
177  OFF KEY 8
178  OFF KEY 9
179  RETURN
180  END
181  !
182  !#######################################################################
183  !
184  SUB Read_failures(@Hp3589a, Left_freq, Right_freq, Command$)
185  !----------------------------------------------------------------------
186  ! MODULE DESCRIPTION:
187  !   This module dumps a block of limit failures from the instrument and
188  !   finds the frequecies for the first and last failure points.
189  !
190  ! INPUTS:   @Hp3589a    : Device to dump data from.
191  !           Command$    : HP-IB mnemonic used to prompt the instrument
192  !                         for the limit failures.
193  ! OUTPUTS:  Left_freq   : Frequency of first failure point.
194  !           Right_freq  : Frequency of lasst failure point.
195  !
196  !----------------------------------------------------------------------
197   REAL Failures(1:1200)
198   DIM A$[10]
199   INTEGER Dig_cnt
200   OUTPUT @Hp3589a;"*OPC?"  ! Make sure input queue is empty
201   ENTER @Hp3589a;a$
202   CLEAR @Hp3589a
203   OUTPUT @Hp3589a;"FORM:DATA REAL,64"
204   OUTPUT @Hp3589a;Command$
205   ASSIGN @Hp3589a;FORMAT ON
206   Dig_cnt=-1
207   ENTER @Hp3589a USING "%,A,D";A$,Dig_cnt
```

```
208   IF (A$<>"#") OR (Dig_cnt<=0) THEN
209     PRINT "NOT CORRECT BLOCK MODE"
210     CLEAR @Hp3589a
211   ELSE
212     ENTER @Hp3589a USING "%,"&VAL$(Dig_cnt)&"D";Num_pts
213     IF (Num_pts MOD 8=0) THEN
214       Num_pts=Num_pts DIV 8
215       ASSIGN @Hp3589a;FORMAT OFF
216       FOR I=1 TO Num_pts
217         ENTER @Hp3589a;Failures(I)
218       NEXT I
219       ASSIGN @Hp3589a;FORMAT ON
220       ENTER @Hp3589a;A$             ! Read CR/LF
221       !
222       Left_freq=Failures(1)
223       Right_freq=Failures(Num_pts-2)
224     ELSE
225       PRINT Data_read;" not float size (divisible by 8)"
226       CLEAR @Hp3589a
227     END IF
228   END IF
229  SUBEND
230  !##########################################################################
```

TRC_LOAD

```
1   ! HP Instrument BASIC example program: TRC_LOAD
2   ! -----------------------------------------------------------
3   ! Program to demonstrate dumping and uploading a trace on the
4   ! HP 3589 with HP Instrument BASIC.
5   !
6   ! -----------------------------------------------------------
7   !
8   DIM Dump_data(1:401), Load_data(1:401)
9   INTEGER Num_pts
10  !
11  ASSIGN @Hp3589a TO 800
12  !
13  CLS
14  DISP "Dumping trace..."
15  Dump_trace(@Hp3589a,Dump_data(*),Num_pts,"TRAC:DATA?")
16  OUTPUT @Hp3589a;"ARM:SOUR MAN"
17  !
18  ! Flip-flop data points before uploading
19  !
20  DISP "Reversing data points..."
21  FOR I=1 TO Num_pts
22    Load_data(Num_pts-I+1)=Dump_data(I)
23  NEXT I
24  !
25  DISP "Uploading trace..."
26  Upload_trace(@Hp3589a,Load_data(*),"TRAC:DATA" )
27  DISP
28  OUTPUT @Hp3589a;"ARM:SOUR IMM"
29  !
30  END
31
!##########################################################################
32  !
33  SUB Dump_trace(@Hp3589a,REAL Trace_data(*),INTEGER Num_pts,Command$)
34  !-----------------------------------------------------------------------
35  ! MODULE DESCRIPTION:
36  !   This module dumps a trace of data from the instrument.
37  !
38  ! INPUTS:    @Hp3589a    : Device to dump data from.
39  !            Command$    : HP-IB mnemonic used to prompt the instrument
40  !                          for the trace of data.
41  ! OUTPUTS:   Trace_data  : Array of data received from instrument.
42  !            Num_pts     : Number of points dumped into Trace_data.
43  !
44  !-----------------------------------------------------------------------
45   DIM A$[10]
46   INTEGER Dig_cnt
47   CLEAR @Hp3589a
48   OUTPUT @Hp3589a;Command$
49   ASSIGN @Hp3589a;FORMAT ON
```

```
50   Dig_cnt=-1
51   ENTER @Hp3589a USING "%,A,D";A$,Dig_cnt
52   IF (A$<>"#") OR (Dig_cnt<=0) THEN
53     PRINT "NOT CORRECT BLOCK MODE"
54     CLEAR @Hp3589a
55   ELSE
56     ENTER @Hp3589a USING "%,"&VAL$(Dig_cnt)&"D";Num_pts
57     IF (Num_pts MOD 8=0) THEN
58       Num_pts=Num_pts DIV 8
59       ASSIGN @Hp3589a;FORMAT OFF
60       ENTER @Hp3589a;Trace_data(*)
61       ASSIGN @Hp3589a;FORMAT ON
62       ENTER @Hp3589a;A$             ! Read CR/LF
63     ELSE
64       PRINT Data_read;" not float size (divisible by 8)"
65       CLEAR @Hp3589a
66     END IF
67   END IF
68  SUBEND
69  !#########################################################################
70  !
71  SUB Upload_trace(@Hp3589a, REAL Trace_data(*), Command$ )
72  !-------------------------------------------------------------------------
73  ! MODULE DESCRIPTION:
74  !   This module uploads a trace of data.
75  !
76  ! INPUTS:    @Hp3589a   : Device to dump data to.
77  !            Command$   : HP-IB mnemonic used to load the trace of
78  !                         data into the instrument
79  ! OUTPUTS:   Trace_data : Array of data to load into instrument.
80  !
81  !-------------------------------------------------------------------------
82   CLEAR @Hp3589a
83   OUTPUT @Hp3589a;"FORM:DATA REAL"
84   OUTPUT @Hp3589a;Command$;" #0";
85   ASSIGN @Hp3589a;FORMAT OFF
86   OUTPUT @Hp3589a;Trace_data(*),END
87   ASSIGN @Hp3589a;FORMAT ON
88  SUBEND
```

TRC_UTILS

```
1   ! HP Instrument BASIC example program:  TRC_UTILS
2   ! ---------------------------------------------------------
3   ! Program to demonstrate dumping transform-coordinated
4   ! display trace data and then plotting it on the HP 3589
5   !
6   ! -----------------------------------------------------------
7   !
8   DIM Dump_data(1:401)
9   INTEGER Num_pts
10  !
11  ASSIGN @Hp3589a TO 800
12  !
13  CLS
14  DISP "Dumping trace..."
15  Dump_trace(@Hp3589a,Dump_data(*),Num_pts,"CALC:DATA?")
16  DISP Num_pts;" points dumped."
17  OUTPUT @Hp3589a;"ARM:SOUR MAN"
18  OUTPUT @Hp3589a;"DISP:PART FULL"
19  DISP "Plotting trace..."
20  Plot_trace(Dump_data(*),Num_pts)
21  DISP
22  !
23  END
24
!################################################################################
25  !
26  SUB Dump_trace(@Hp3589a,REAL Trace_data(*),INTEGER Num_pts,Command$)
27  !-----------------------------------------------------------------------
28  ! MODULE DESCRIPTION:
29  !   This module dumps a trace of data from the instrument.
30  !
31  ! INPUTS:    Command$    : HP-IB mnemonic used to prompt the instrument
32  !                          for the trace of data.
33  ! OUTPUTS:   Trace_data  : Array of data received from instrument.
34  !            Num_pts     : Number of points dumped into Trace_data.
35  !
36  !-----------------------------------------------------------------------
37   DIM A$[10]
38   INTEGER Dig_cnt
39   CLEAR @Hp3589a
40   OUTPUT @Hp3589a;"FORM:DATA REAL,64"
41   OUTPUT @Hp3589a;Command$
42   ASSIGN @Hp3589a;FORMAT ON
43   Dig_cnt=-1
44   ENTER @Hp3589a USING "%,A,D";A$,Dig_cnt
45   IF (A$<>"#") OR (Dig_cnt<=0) THEN
46     PRINT "NOT CORRECT BLOCK MODE"
47     CLEAR @Hp3589a
48   ELSE
49     ENTER @Hp3589a USING "%,"&VAL$(Dig_cnt)&"D";Num_pts
```

```
50    IF (Num_pts MOD 8=0) THEN
51      Num_pts=Num_pts DIV 8
52      ASSIGN @Hp3589a;FORMAT OFF
53      ENTER @Hp3589a;Trace_data(*)
54      ASSIGN @Hp3589a;FORMAT ON
55      ENTER @Hp3589a;A$             ! Read CR/LF
56    ELSE
57      PRINT Data_read;" not float size (divisible by 8)"
58      CLEAR @Hp3589a
59    END IF
60  END IF
61 SUBEND
62 !##############################################################
63 !
64 SUB Plot_trace(REAL Trace_data(*),INTEGER Num_pts)
65 !-------------------------------------------------------------
66 ! MODULE DESCRIPTION:
67 !   This module plots a trace of data.
68 !
69 ! INPUTS:   Trace_data : Array of data to plot.
70 !           Num_pts    : Number of points in Trace_data to plot.
71 !           Title$     : Title to put on plot.
72 !           X_annot$   : X axis annotation for plot.
73 !           Y_annot$   : Y axis annotation for plot.
74 ! OUTPUTS:  Min_bin    : Index location of smallest value in Trace_data.
75 !           Max_bin    : Index location of largest value in Trace_data.
76 !
77 !-------------------------------------------------------------
78  REAL Y_scaler          ! 1/(Delta per vertical pixel)
79  REAL Max_val,Min_val   ! Maximum and minimum values in data trace
80  REAL Height,Width      ! Display height and width
81  INTEGER Orig_x,Orig_y  ! Origin (in pixels)
82  !
83  Height=340
84  Width=401
85  Orig_x=10
86  Orig_y=15
87  !
88  GCLEAR
89  Max_val=MAX(Trace_data(*))
90  Min_val=MIN(Trace_data(*))
91  !
92  ! Perform display auto-scale on data
93  !
94  Y_scaler=Height/(ABS(Max_val-Min_val))
95  Top_pixel=(Height+Orig_y)
96  MOVE Orig_x,Top_pixel-(ABS(Max_val-Trace_data(1))*Y_scaler)
97  !
98  FOR X=1 TO Num_pts
99    DRAW Orig_x+X-1,Top_pixel-(ABS(Max_val-Trace_data(X))*Y_scaler)
100  NEXT X
101 SUBEND
```

TWO_CTLR

```
10  !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 3589A and run it two
60  !times.  After each run, two BASIC program variables will
70  !will be read from the HP 3589A and displayed.
80   !
90  Scode=7                               !Select code for interface
100 Address=19                            !Address for HP 3589A
110 Hp3589a=Scode*100+Address
120 !
130 CLEAR Hp3589a
140 OUTPUT Hp3589a;"PROG:DEL:ALL"         !Scratch the program space
150 !
160 DISP "Downloading the program..."
170 ASSIGN @Prog TO Hp3589a;EOL CHR$(10)     !Change EOL character
180 OUTPUT @Prog;"PROG:DEF #0";              !Send program
190 OUTPUT @Prog;"10 COM INTEGER Times_run,Test$[10]"
200 OUTPUT @Prog;"20 Times_run=Times_run +1"
210 OUTPUT @Prog;"30 IF Times_run=1 THEN Test$=""PASS"""
220 OUTPUT @Prog;"40 IF Times_run=2 THEN Test$=""FAIL"""
230 OUTPUT @Prog;"50 BEEP"
240 OUTPUT @Prog;"60 END"
250 OUTPUT @Prog;CHR$(10) END                !Terminate the data block
260 !
270 !Set up registers for interrupt on PROGRAM_RUNNING going false
280 OUTPUT Hp3589a;"*CLS"             !Clear the STATUS BYTE register
290 OUTPUT Hp3589a;"STAT:OPER:NTR 16384"  !Program NTR reg for
PROGRAM_RUNNING bit
300 OUTPUT Hp3589a;"STAT:OPER:ENAB 16384" !Set OPERATION ENABLE reg bit 14
310 OUTPUT Hp3589a;"*SRE 128"      !Allow SRQ on OPERATIONAL_STATUS_EVENT bit 7
320 !
330 DISP "Running the program..."
340 OUTPUT Hp3589a;"PROG:STAT RUN"       !Run Program
350 Display_res(Hp3589a,Scode)           !Read and display variables
360 OUTPUT Hp3589a;"PROG:STAT RUN"       !Run Program  again
370 Display_res(Hp3589a,Scode)           !Read and display variables
380 !
390 END                                !End of this program
400  !
410 SUB Display_res(Hp3589a,Scode)
420 ! This subprogram waits for an SRQ interrupt to signal that a
430 ! BASIC program has finished.  It then clears the HP-IB registers
440 ! by reading them.  Once that is done, the values of two IBASIC
450 ! variables are read and displayed.
460 !
470   ON INTR Scode GOTO Read_results    !Set up interrupt branching
480   ENABLE INTR Scode;2                !Allow interrupt on SRQ
490 Idle:GOTO Idle
```

```
500    !
510 Read_results:                    !Program has finished
520    A=SPOLL(Hp3589a)                 !Read and clear the SRQ
530    OUTPUT Hp3589a;"STAT:OPER?"      !Read and clear OPERATION STATUS reg.
540    ENTER Hp3589a;Event
543    WAIT .5
550     !
551    OUTPUT Hp3589a;"FORM:DATA ASCII,3"
560    OUTPUT Hp3589a;"PROG:NUMB? 'Times_run'"      !Read the first variable
570    ENTER Hp3589a;Times_run
590     !
600    OUTPUT Hp3589a;"PROG:STR? 'Test$'"           !Read the second variable
610    ENTER Hp3589a;Test$
630     !
640    PRINT "Times_run: ";Times_run,"Test$: ";Test$
650  SUBEND
```

WAI_SYNC

```
10    ! HP Instrument BASIC program: WAI_SYNC -  Measurement synchronization
20    ! ------------------------------------------------------------------
30    ! This program demonstrates how to use the *WAI command to
40    ! prevent execution of an HP-IB command until all previous
50    ! commands have finished.  In this example, the trace display
60    ! will not change to the UPPER/LOWER FORMAT until after the
70    ! measurement has finished.
80    !
90    ! The *WAI command does not affect program operation. The
100   ! program will run to completion, sending all of the commands to
110   ! to the HP3589A without waiting for them to be executed.
120   ! -------------------------------------------------------------------
130   Scode=8                            !Interface select code
140   Address=00
150   Hp3589a=Scode*100+Address
160   !
161   DISP "Sending HP-IB commands..."
170   OUTPUT Hp3589a;"SCR:CONT TRACE;FORM SING" !Set format to single
180   OUTPUT Hp3589a;"SWE:TIME 8 S"      !Set record length to 8 seconds
190   OUTPUT Hp3589a;"ABORT; INIT"       !Start the measurement
200   OUTPUT Hp3589a;"*WAI"              !Tell analyzer to wait here until
210                                      !all HP-IB commands have finished
220   OUTPUT Hp3589a;"SCR:CONT TRACE;FORM ULOW" !Go to upper/lower after
waiting
230   BEEP
240   DISP "Finished. Display will go to UPPER/LOWER when meas. done"
250   END
```

Chapter 11

Instrument-Specific HP Instrument BASIC Features

11

Instrument-Specific HP Instrument BASIC Features

Introduction

The *HP Instrument BASIC Users Handbook* that accompanies this manual includes *HP Instrument BASIC Programming Techniques*, *HP Instrument BASIC Interfacing Techniques*, and *HP Instrument BASIC Language Reference*. These references are 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 interpret the *HP Instrument BASIC Users Handbook* for the HP 3589A by listing exceptions and additions. Global differences apply throughout the manual set and discuss differences by category. Specific differences are listed by command.

Global Exceptions

In general, the differences can be categorized as

- HP-IB, GPIO and RS-232 interfaces.
- CRT and keyboard interfaces
- Disk I/O
- Miscellaneous command differences
- Commands not implemented

Each of these categories is explained in detail in this section.

HP-IB, GPIO and RS-232 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 3589A supports only the HP-IB interface, and not the GPIO or RS-232 interfaces.

Also, HP Instrument BASIC in the HP 3589A supports only two interface select codes: 7 for the external bus, and 8 for the internal bus. This effects the ABORT, ENABLE INTR, DISABLE INTR , ON INTR, OFF INTR, ON TIMEOUT, and OFF TIMEOUT commands, where select codes are limited to 7 and 8. Furthermore, the LOCAL LOCKOUT and REMOTE commands do not apply to the internal HP-IB interface (select code 8).

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

Finally, the internal default select codes for CRT (1) and keyboard (2) apply, but the select codes for GPIO (12), and RS-232 (9) do not apply in the HP 3589A. The CRT function returns 1 and the KBD function returns 2 and, when select code 2 is used with ENTER (as in ENTER 2) the same menu as is used by the INPUT command appears to allow text entry from the front panel. "ENTER CRT " is not supported.

CRT and Keyboard Interfaces

CRT Differences

References to the CRT in the *HP Instrument BASIC Users Handbook* generally assume a standard 80 column terminal. The HP 3589A has a 58 column display for text. This effects references to the width of the default PRINTER IS device (the CRT) in the LIST, PRINT and PRINTER IS commands.

Also, since the instrument shares the display with HP Instrument BASIC, you must allocate a display partition in order to view any output to the CRT. This effects both the text commands just listed and also the graphics commands MOVE and DRAW. Three different display partitions may be allocated, UPPER, LOWER or FULL. The text width for all three is the same so 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,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.

Finally, the PEN command is implemented in the HP 3589A 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. Note that 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 3589A has neither a keyboard nor a command line execution mode. However, it does have front-panel hardkeys that are mapped to alpha keys, and softkeys that emulate many of the keywords that would be executed from the command line on an HP BASIC workstation (such as RUN, CONTINUE, SCRATCH, etc.). The following HP Instrument BASIC keywords have corresponding softkeys and hardkeys in the HP 3589A:

CONTINUE

[**BASIC**] [CONTINUE]

[**BASIC**] [DEBUG] [CONTINUE]

DEL

[**BASIC**] [EDIT] [DELETE LINE]

EDIT

[**BASIC**] [EDIT]

LIST

[**BASIC**] [PRINT PROGRAM]

PAUSE

[**BASIC**]

RUN

[**BASIC**] [RUN]

[**BASIC**] [DEBUG] [RUN]

SCRATCH

[**BASIC**] [UTILITIES] [SCRATCH]

SECURE

[**BASIC**] [UTILITIES] [SECURE]

STOP

[**Local/HP-IB**]

REN

[**BASIC**] [UTILITIES] [RENUMBER]

Note that 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 3589A front-panel operations. **These 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.

The following keywords have descriptions in the *HP Instrument BASIC Language Reference* that use standard keyboard keys and need clarification for use in the HP 3589A:

EDIT

Ignore all documentation in the *HP Instrument BASIC Language Reference* on the EDIT command. See chapter 6 for information on using the editor in the HP 3589A.

ON KEY and OFF KEY

There are 10 softkeys available for use in the HP 3589A. These appear on the right of the display in place of normal instrument softkeys. Key selector values may range from 0 through 9.

INPUT

When an INPUT statement is encountered in an HP Instrument BASIC program, an alpha entry menu appears to allow the user to use the front panel alpha keys and the symbol softkeys to enter a response.

To enter an input response press the [ENTER] softkey in the alpha entry menu. Disregard all keys mentioned in the *HP Instrument BASIC Language Reference* description of this key. You have only two options for terminating an INPUT command: Press the [ENTER] or the [PAUSE] softkey. (You cannot press the [**BASIC**] key to pause or the [**Local/HP-IB**] hardkey to stop since these are redefined as alpha keys.) If you press the [PAUSE] softkey, when the [**BASIC**] [CONTINUE] or [**BASIC**] [DEBUG] [SINGLE STEP] softkeys are pressed to continue the program, the INPUT statement will be re-executed.

ENTER

For a description of using ENTER from the keyboard (ENTER KBD) see INPUT.

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 3589A, since no commands are keyboard executable.

Disk I/O

Disk Format

HP Instrument BASIC in the HP 3589A recognizes only LIF (Logical Interface Format) disk and files. The *HP Instrument BASIC Users Handbook* addresses LIF, HP-UX, and MS-DOS file formats. HP-UX and MS-DOS are hierarchical file systems (contain directories) and are referred to by the term HFS (Hierarchical File System). In general, disregard all references to HP-UX, MS-DOS or HFS throughout the *HP Instrument BASIC Users Handbook.* Sections with references to HFS appear in commands ASSIGN, CAT, COPY, CREATE ASCII, CREATE BDAT, GET, INITIALIZE, MASS STORAGE IS, PURGE, PRINTER IS, RENAME, RE-SAVE and SAVE, and should be ignored. Several of these commands also contain references to HP-UX files, which should be disregarded.

Another exception is the way the LIF protect codes are handled. LIF protect codes are ignored by HP Instrument BASIC in the HP 3589A and are neither generated nor observed. They will not generate an error if encountered.

Volume Specifiers

HP Instrument BASIC in the HP 3589A does not support external disk drives. There are only three mass storage devices available in the HP 3589A, the internal disk drive, volatile RAM disk (memory unit 0) and non-volatile RAM disk (memory unit 1). This effects the volume specifier parameter in the commands ASSIGN, CAT, COPY, CREATE ASCII, CREATE BDAT, GET, INITIALIZE, MASS STORAGE IS, PURGE, PRINTER IS, RENAME, RE-SAVE and SAVE.

Throughout the *HP Instrument BASIC Language Reference*, the MASS STORAGE IS command is referenced for a description of valid volume specifiers. Disregard the volume specifiers in the MASS STORAGE IS command description. Valid volume specifiers for each mass storage device are shown below.

Table 11-1. Mass Storage Volume Specifiers

Disk	Volume Specifier
INTERNAL	:INTERNAL,4,0
	:,4,0
	:,4
VOLATILE RAM	:MEMORY,0,0
	:MEMORY,0
	:,0
NON-VOLATILE RAM	:MEMORY,0,1
	:,0,1

Initializing Disks

The description of the INITIALIZE command in the *HP Instrument BASIC Language Reference* is incorrect for the HP 3589A in two ways. First, like all other disk commands, it must use valid HP 3589A volume specifiers as previously described. Second, the RAM unit size is limited to values from 0 through 5. The INITIALIZE command takes the following form for RAM:

```
INITIALIZE <volume specifier>, <unit specifier>
```

For example, to INITIALIZE the volatile RAM disk to 622080 bytes, use a unit specifier of 1 as follows:

```
INITIALIZE ":,0,0",1
```

The following table shows valid RAM unit size parameters and the resulting RAM disk sizes for both volatile and non-volatile RAM disks:

Table 11-2. RAM Unit Disk Sizes

UNIT SPECIFIER	SIZE OF INITIALIZED DISK (IN BYTES)	
	VOLATILE RAM	NON-VOLATILE RAM
(none)	64256	63488
0	64256	63488
1	622080	63488
2	699904	63488
3	777728	63488
4	262656	63488
5	622080	63488

Miscellaneous Command Differences

COS

The range of the COS command is all absolute values less than 1.7083127722 E+10 degrees.

SYSTEM$

The SYSTEM$ command accepts only one topic specifier, SYSTEM PRIORITY.

Commands Not Implemented

The following commands appear in the *HP Instrument BASIC Language Reference* but are not implemented in HP Instrument BASIC in the HP 3589A:

CREATE

This creates an HP-UX file, which is not supported.

CREATE DIR

This creates a directory on an HFS volume, which is not supported.

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 3589A. Table 11-3 in the following section lists all HP Instrument BASIC keywords supported by HP Instrument BASIC in the HP 3589A.

Specific Exceptions

Table 11-3 presents a summary of the HP Instrument BASIC keywords used in the HP 3589A and 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 3589A. 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 3589A and should be ignored.

Table 11-3. HP 3589A 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	No directory path (no HFS) LIF protection ignored One HP-IB device per ASSIGN statement
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 (Not associated with [**Save/Recall**] functions)	No HFS Catalogs See "Volume Specifiers" in "Disk I/O"
CHR$	None	None
CLEAR	None	None
CLEAR SCREEN	None	None
COM	None	None
CONT	[**BASIC**] [CONTINUE] or [**BASIC**] [DEBUG] [CONTINUE]	No line number or label support

Table 11-3. HP 3588A Keyword Implementation (Continued)

Command	Front Panel Support	Exceptions
COPY	None (Not associated with [**Save/Recall**] functions)	No directory path (no HFS) LIF protection ignored See "Volume Specifiers" in "Disk I/O"
COS	None	Absolute range values less than 1.7083127722 E +10
CREATE ASCII	None	No directory path (no HFS) LIF protect code not allowed See "Volume Specifiers" in "Disk I/O"
CREATE BDAT	None	No directory path (no HFS) LIF protection ignored See "Volume Specifiers" in "Disk I/O"
CRT	None	ENTER CRT not supported
DATA	None	None
DEF FN	None	None
DEG	None	None
DIM	None	None
DISABLE	None	None
DISABLE INTR	None	None
DISP	None	None
DIV	None	None
DEL	[**BASIC**] [EDIT] [DELETE LINE]	Deletes only the current line
DRAW	None	Maximum x,y coordinates: Full partition (475,355) Upper partition (475,173) Lower partition (475,173)
DROUND	None	None
DVAL	None	None
DVAL$	None	None
EDIT	[**BASIC**] [EDIT]	Editing functions described in chapter 6.
ENABLE	None	None
ENABLE INTR	None	Interface Select Code = 7 or 8 Must not precede an ON INTR statement.
END	None	None
ENTER	None	Files as source do not include HP-UX
ERRL	None	None
ERRLN	None	None

Table 11-3. HP 3588A Keyword Implementation(Continued)

Command	Front Panel Support	Exceptions
ERRM$	None	None
ERRN	None	None
EXOR	None	None
EXP	None	None
FN	None	None
FOR...NEXT	None	None
FRACT	None	None
GCLEAR	None	Clears the graphics buffer
GET	None	See "Volume Specifiers" in "Disk I/O"
GOSUB	None	None
GOTO	None	None
IF...THEN	None	None
IMAGE	None	None
INITIALIZE	None (Not associated with [**Save/Recall**] functions)	RAM unit size = 0 thru 5 HFS volumes not supported See "Volume Specifiers" in "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	Keyboard = front-panel alpha keys
LEN	None	None
LET	None	None
LGT	None	None
LIST	[**BASIC**][PRINT PROGRAM]	Default width = 58 (see PRINTER IS)
LOCAL	None	None
LOCAL LOCKOUT	None	Interface Select code = 7 only
LOG	None	None
LOOP	None	None
LWC$	None	None
MASS STORAGE IS	None	See "Volume Specifiers" in "Disk I/O"

Table 11-3. HP 3588A Keyword Implementation(Continued)

Command	Front Panel Support	Exceptions
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 (475,355) Upper partition (475,173) Lower partition (475,173)
NOT	None	None
NUM	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 or 8
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 or 8
OR	None	None
OUTPUT	None	No HFS or HP-UX file support
PEN	None	0 = erase 1 = draw
PASS CONTROL	None	Select code 8 pass control of external bus to analyzer
PAUSE	None	None
POS	None	None
PRINT	None	PRINTER IS default width = 58
PRINTER IS	None	default width = 58 No directory path (no NFS) LIF protection ignored See "Volume Specifiers" in "Disk I/O"
PROUND	None	None
PRT	None	None

Table 11-3. HP 3588A Keyword Implementation(Continued)

Command	Front Panel Support	Exceptions
PURGE	None	No directory path (no HFS) LIF protection ignored 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	Not supported on select code 8
REN	[**BASIC**] [UTILITIES] [RENUMBER]	No line label support
RENAME	None	No directory path (no HFS) LIF protection ignored See "Volume Specifiers" in "Disk I/O"
REPEAT...UNTIL	None	None
RE-SAVE	None (Not associated with [**Save/Recall**] functions)	See "Volume Specifiers" in "Disk I/O"
RESTORE	None	None
RETURN	None	None
REV$	None	None
RND	None	None
ROTATE	None	None
RPT$	None	None
RUN	[**BASIC**] [RUN] or [**BASIC**] [DEBUG] [RUN]	
SAVE	None (Not associated with [**Save/Recall**] functions)	See "Volume Specifiers" in "Disk I/O"
SCRATCH	[**BASIC**] [UTILITIES] [SCRATCH]	No HFS
SECURE	[**BASIC**] [UTILITIES] [SECURE]	None
SELECT...CASE	None	None
SGN	None	None
SHIFT	None	None
SIN	None	None

Table 11-3. HP 3588A Keyword Implementation(Continued)

Command	Front Panel Support	Exceptions
SIZE	None	None
SPOLL	None	None
SQRT	None	None
STOP	None	None
SUB	None	None
SYSTEM PRIORITY	None	None
SYSTEM$	None	Only supports " SYSTEM PRIORITY "
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

Index