HEWLETT-PACKARD

HP-IB Interface

OWNER'S MANUAL

SERIES 80

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

HEWLETT PACKARD

Series 80 HP-IB Interface Owner's Manual

January 1982

82937-90017

Printed in U.S.A.

Contents

Section 1: General Information 5
Introduction 5
Plug-in ROMs 5
Specifications 5

Section 2: Installation 7
Unpacking and Inspection 7
Installing the Interface 7
Connecting Peripherals 8
Removing Peripherals 9
Disconnecting the Interface 9
Interconnecting Cables 10
Accessory Cables 10
Cable Length Restrictions 11
Metric Conversion Kits 11
Changing the Switch Settings 11
Disassembly 11
Switch Settings 12
Parallel Poll Response Lines 18

Section 3: Using Your HP-IB Interface 21
Introduction 21
HP-IB Operations 21
Turn-on and Check-out 22
Controlling the Bus 23
HP-IB Data Transfers 25
Advanced I/O Operations 27
Handling Service Requests 32
Noncontroller Operations 34
Handling Interface Problems 36
HP-IB Limitations 38

Section 4: HP-IB for the Specialist 39
Introduction 39
Controlling with the HP-IB 39
HP-IB I/O Statements 39
Typical HP-IB Output Sequence 42
Typical HP-IB Enter Sequence 42
Mnemonic Conventions 43
HP-IB Control Lines 44
HP-IB Control Responses 46
Polling 47
Serial Poll 47
Parallel Poll 48

Section 5: Control Registers 49
Control Registers 0-3 49
HP-IB Control Registers 49
User-defined End-of-Line Sequence Registers 52
Set I/O for Register Control 53
HP-IB Status Registers 54

Section 6: Maintenance, Service, and Warranty 57
Maintenance 57
Service 57
Warranty and Repair Information 57
Potential for Radio Frequency Interference 58
General Shipping Instructions 59

Appendix A: Functional Description 61
Introduction 61
Hardware Description 62
HP-IB Bus Lines 64
Data Transfer 65
System Controller 67
Bus Functions 69
HP-IB Messages 69
HP-IB Universal Commands 71
Available Bus Addresses and Codes 72
Functional Test 75

Appendix B: A Basic Introduction to the HP-IB 79
General Structure of the HP-IB 79
Data Transfers on the HP-IB 80
Controlling the HP-IB 82
Handling Requests for Service 83

Appendix C: HP-IB Errors 87

Appendix D: HP-IB (I/O ROM) Statement Summary 89

Figures: 8
2-1. Removing the Protective Cover 8
2-2. Connecting the Interface 8
2-3. Disconnecting the Interface 9
2-4. Connector Cable 10
2-5. Adapter 10
2-6. Disassembly 13
2-7. Select Code Switches 14
2-8. Talk/Listen Addresses 16
2-9. System Controller Switch 17
2-10. Removing the Cable Connector 19
2-11. Parallel Poll Jumper Wire Placement 19
4-1. HP-IB Signal Lines 45
5-1. Event Initiated Interrupt Timing 51
5-2. State-Initiated Interrupt Timing 51
5-3. SRQ Interrupt Timing 52
A-1. HP-IB Interface Block Diagram 61
A-2. Data Transfer Timing Diagram 66
A-3. Interface Schematic and Component Layout Diagrams 73

Tables: 38
3-1. Error Handling Methods 38
4-1. Transfer Terminators 41
4-2. HP-IB Control Responses 47
5-1. HP-IB Control Registers 50
5-2. HP-IB End-of-Line Sequence Registers 53
5-3. HP-IB Status Registers 54
A-1. HP-IB Signal Lines 64
A-2. Interface Functions and Allowed Capability 68
A-3. HP-IB Messages 69
A-4. Primary Command Group 71
A-5. Bus Addresses/Codes 72
A-6. HP-IB Interface Replaceable Parts List 74

Notes

Section 1

General Information

Introduction

The HP-IB Interface connects the HP Series 80 Personal Computer to the Hewlett-Packard Interface Bus. This interface conforms to the Institute of Electrical and Electronics Engineers' (IEEE) Standard 488-1978 and supports a wide variety of operations, enabling you to communicate with and control peripheral devices with the HP Series 80 Personal Computer.

The HP-IB interface is used with many types of systems to allow the transfer of data and command messages over short distances. There are two major classes of use for HP-IB: instrumentation I/O (control and measurement) and data entry/output. Examples of the former abound in scientific research, biomedical electronics, and process monitoring. The other category, which we refer to as **peripheral use** is illustrated by HP-IB connections to printers, plotters, disc drives, and other data sources and destinations. The instrumentation category requires an I/O ROM and is discussed at length in the section entitled Using Your HP-IB Interface. The connection to HP-IB compatible printers and other peripherals is much simpler and is explained in the first few paragraphs of that section. This manual provides general information about the interface, theory of operation, and programming details. It is intended to serve as a complete usage manual for both the plug-in interface used with the HP Series 80 Personal Computer as well as the built-in interface provided by the HP-87. Some topics will apply to one of these only and these will be obvious by context or explained as they occur.

Plug-In ROMs

This interface requires one of three available ROMs to perform input-output operations. The type of ROM used determines which bus operations can be performed. The available ROMs include:

- Mass Storage ROM (part no. 00085-15001 for the HP-85/83)
- Plotter/Printer ROM (part no. 00085-15002 for the HP-85/83 and part no. 00087-15002 for the HP-87)
- I/O ROM (part no. 00085-15003 for the HP-85/83 and part no. 00087-15003 for the HP-87)

The Mass Storage ROM supports interfacing of available disc-drive units, and the Plotter/Printer ROM simplifies interfacing plotter and printer devices. The I/O ROM is a general purpose ROM and is used in a wide variety of applications.

These ROMs fit into a ROM drawer that plugs into any of the four plug-in module ports on the HP Series 80 Personal Computer. Together with one of the plug-in ROMs mentioned above, the HP-IB interface enables you to communicate with a wide variety of peripheral devices.

Specifications

Pertinent specifications of the interface are listed below. If complete details of HP-IB electrical, mechanical, and timing requirements are desired, refer to IEEE Standard 488-1978. More detailed specifications are also available in appendix A.

Dimensions	Approximately 16.7 × 12.7 × 1.5 cm (6.59 × 5 × 0.59 in).
Weight	0.5 kg (1.1 lb).
Cable Length	Approximately two meters. (Also, refer to Cable Length Restrictions in section 2.)
Operating Temperature	0° to 55°C (32° to 131°F).
Power Requirements	The mainframe supplies all power for the interface.
Maximum Peripheral Load	The maximum number of peripherals that may be interfaced to the HP Series 80 Personal Computer via the HP-IB interface is 14.

Section 2

Installation

If your HP-IB interface is built into your HP Series 80 Personal Computer, the paragraphs in this section referring to interface installation and removal do not apply. The paragraphs regarding connecting and disconnecting peripherals, however, should be carefully studied.

Unpacking and Inspection

If the shipping carton is damaged, ask the carrier's agent to be present when the interface is unpacked. If the interface is damaged or fails to meet electrical specifications, immediately notify the carrier and the nearest HP sales and service office. Retain the shipping carton for the carrier's inspection. The sales and service office will arrange for the repair or replacement of your interface without waiting for the claim against the carrier to be settled.

Installing the Interface

Make sure you read and understand this entire procedure, especially the following precautions, before you install the interface or connect a peripheral device to it.

Manufacturers of peripheral devices do not all use the same grounding technique. Often, earth ground and logic ground are at different voltage levels. In some instances, this is deliberate in an effort to reduce ground return interference with digital signals.

When the HP 82937A HP-IB Interface is installed on the HP Series 80 Personal Computer, earth ground and logic ground become connected together. Thus, if logic ground on a peripheral is never connected to earth ground or, if it is defective, it may have a voltage level considerably different than logic ground on the interface. This potential may be high enough to be hazardous unless peripherals are connected to the bus in an exacting manner.

If you don't know the grounding technique used on a peripheral, check with the manufacturer of the device. After verifying that suitable grounding techniques have been used in your peripheral, use the following steps in the order given to install the interface and peripherals to the HP-IB interface.

WARNING

To avoid personal injury and equipment damage, read and understand the preceding safety precautions and do not deviate from the order of the following steps to install the interface and peripheral(s).

CAUTION

Always turn off the HP Series 80 Personal Computer and any connected peripherals *before* installing or removing the interface module. If this is not done, the system may be damaged.

1. Turn the power switch, located on the rear panel of the computer, to the OFF position. However, make sure the computer power cord is plugged into a grounded (three-wire) ac outlet.

WARNING

Do not place fingers, tools, or other foreign objects into the HP Series 80 Personal Computer's plug-in ports. Such actions may result in minor electrical shock hazard and interference with pacemaker devices worn by some persons. Damage to the computer's port contact and internal circuitry may also result.

2. Remove the protective cover from any unused port (figure 2-1). The remaining unused ports should be kept covered.

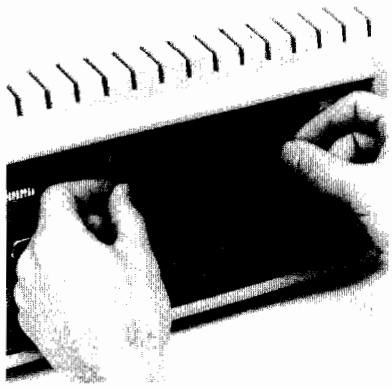


Figure 2-1.
Removing the Protective Cover

CAUTION

Do not force the interface module into the port. If the module is upside-down, forcing it further may result in damage to the computer or to the module itself.

3. Refer to figure 2-2 and position the module as shown. Slide the module into one of the I/O ports until the grips meet the sides of the port. A slight side-to-side motion may be necessary to seat the module in the port. Notice that the tracks are keyed to prevent the module from being inserted upside down.

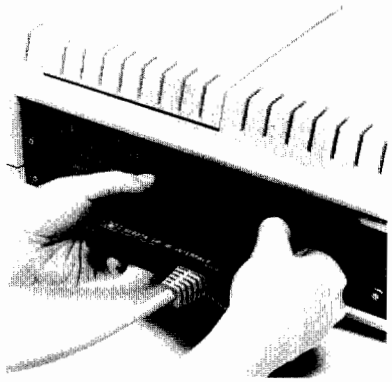


Figure 2-2.
Connecting the Interface

Connecting Peripherals

1. Make sure the power switches on all peripherals to be connected to the HP-IB interface cable are in the OFF position.
2. Check the peripheral device address switches to ensure that they are at the desired settings. Remember that the peripheral address must be a decimal number from 0 through 30; address code 31 is not allowed. Also, do not set peripheral device address switches to the same address as the computer talk/listen address (factory set to 21).

3. Connect the interface cable to the first peripheral. If other peripherals are to be connected at this time, first make sure they all have their power switches in the OFF position. Then, using the accessory cables listen on page 10, install the cable connectors to the remaining peripherals in piggy-back fashion, making sure to observe the cable length restrictions given on page 11.
4. After all the peripherals are connected to the HP-IB interface, turn the power switches on the computer and all peripherals to be operational on the interface to the ON position. A peripheral that is connected to the interface can have its power off without affecting interface operations, as long as more than 50% of the interfacing devices have their own power on. For example, if there are three peripheral devices, two must have their power on; if there are two peripherals, both must have their power on.

Note: Peripheral devices should always be turned on before the computer power is switched on or a system reset should be performed so that the computer will be sure to recognize the existence of peripherals that are connected to the bus.

Removing Peripherals

Use the following steps in the order given to remove peripherals from the HP-IB interface.

> **CAUTION**
>
> Do not remove the interface from the computer with the power switch on. Doing this will cause damage to either the interface, the computer, or both.

1. Turn the power switch on all peripherals connected to the HP-IB interface to the OFF position.
2. Disconnect the HP-IB interface cable from each peripheral you intend to remove from the interface. If all peripherals are not to be removed from the interface, you may return power to the remaining peripherals after the desired peripherals have been disconnected.

Disconnecting the Interface

Use the following steps in order to remove the plug in interface (HP 82937A HP-IB Interface) from the computer.

1. Turn the power switch on the rear panel of the computer to the OFF position. Make sure the interface cable is not connected to a peripheral and proceed with step 2.
2. To remove the interface module from the I/O module port, firmly grasp the module and pull it from the port (figure 2-3). Store the interface module in its original container or where it will be safe from damage.
3. Replace the port cover.

Figure 2-3.
Disconnecting the Interface

Interconnecting Cables

Accessory Cables

WARNING

Before you attempt to use any of the accessory cables, make sure you read and understand the safety precautions on page 7.

The length of the interface cable is 2 meters. The end of the cable has a piggy-back connector that connects to a peripheral device. On some peripherals, a physical interference problem may exist for the cable connector. Any peripheral that will not directly accept the cable connector will require the adapter. Other peripheral devices may be added to the interface by stacking the piggy-back connectors of the standard interface cables listed below.

Length	Accessory Number
½ meter	HP 10833D
1 meter	HP 10833A
2 meters	HP 10833B
4 meters	HP 10833C
Adapter	HP 10834A

There are no restrictions as to how cables may be connected together. However, it is recommended that no more than three or four piggy-back connectors be stacked together on one device. Doing so could exert enough force on the connector mounting to cause mechanical damage.

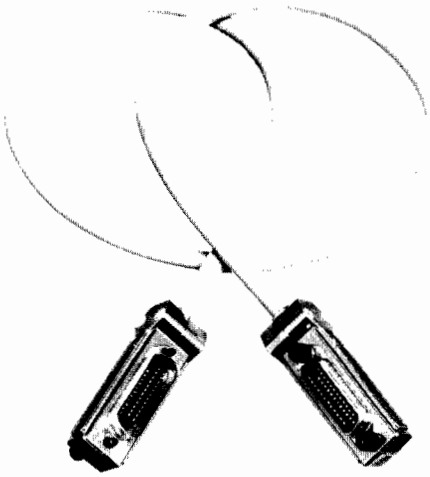


Figure 2-4. Connector Cable

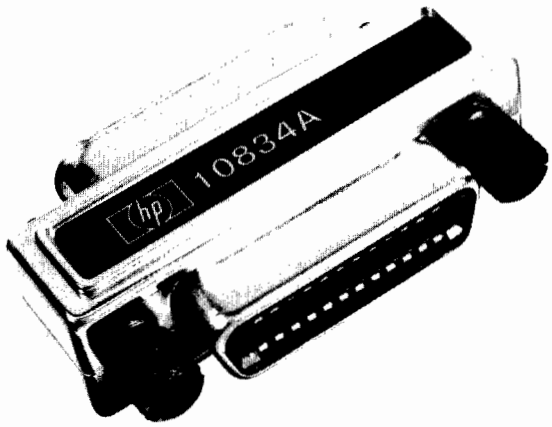


Figure 2-5. Adapter

Cable Length Restrictions

In order to ensure proper operation of the interface, two rules must be observed regarding the total length of bus cables when they are connected together:

- The total length of cable permitted with one bus interface must be less than or equal to 2 meters times the number of devices connected together. (The interface is counted as one device.)
- The total length of cable must not exceed 20 meters.

For example, there may be up to 4 meters of cable between the first two devices (2 devices × 2 meters/device = 4 meters). Additional devices may be added using 2-meter cables up to a total of 10 devices (10 devices × 2 meters/device = 20 meters) using one 4-meter and eight 2-meter cables (4 + (8 × 2) = 20). If more than 10 devices are to be connected together, cables shorter than 2 meters must be used between some of the devices. For example, 15 devices can be connected together using one 2-meter and thirteen 1-meter cables (2 + (13 × 1) = 15). Other combinations may be used as long as both requirements are met.

Metric Conversion Kit

The interface cable is supplied with mounting fasteners having metric threads. Other HP-IB instruments, however, may have either National Coarse (American) threads or metric threads. The American-threaded fasteners are chromium plated, while the metric-threaded fasteners are black.

Since metric and American threads cannot be connected together, a conversion kit is available. Use this kit to replace the mounting fasteners on any bus cable connector. Metric conversion kits can be ordered by specifying HP part number 5060-0138.

Changing the Switch Settings

The select code switches, talk/listen address switches, and system controller switch are preset at the factory as follows:

- Select Code 7
- Talk/Listen Address 21
- System Controller Enabled

For the external, plug-in HP-IB interface, verifying or changing any of the above requires disassembly of the interface housing. If this is necessary, use the following disassembly procedure and disassemble the interface. Then refer to the discussion of the function requiring verification.

The internal, built-in HP-IB interface has two switches on the back panel of the computer which can be easily reconfigured. The following disassembly discussion will not apply.

Disassembly

Refer to figure 3-1 to see how the interface parts fit together. Place the interface on a flat surface with the side having the seven screws facing upward and the cable coming out the left. Then use the following steps to disassemble the interface.

1. Using a Pozidriv® screwdriver, remove the seven screws and set them aside. (Do not use a Phillips screwdriver; the screw heads may be damaged.)

2. Hold the interface parts together and turn the interface over so that the screw holes are facing downward and the cable is still coming out to the left.
3. Hold the cable strain relief in place and remove the top half of the interface housing.

If you have followed the above steps, switch S1 should be oriented as shown in the following figures. If it isn't, orient it as illustrated before identifying the switch segments.

When you reassemble the interface, reverse the above procedure, making sure that the ground contact is in place. The ground contact should be under the printed-circuit assembly when the component side is up.

Switch Settings

As previously noted, the select code, talk/listen address, and system controller functions are preset at the factory. Switch segments 2 through 10 of switch S1 on the interface printed-circuit assembly or back panel of the computer are used for this purpose. Switch segment 1 is not used and may be in either position.

Once the interface is disassembled, any of the factory settings may be verified or changed by referring to the following sections that discuss these individual functions.

The interface may be equipped with either slide or rocker type switches. Both types are illustrated in the factory preset positions.

> Note: If you change any of the factory settings, make sure that you change the proper switch segments. Do not disturb the settings of adjoining switches. The small tip of a pencil or similar object is recommended for this purpose. It is unlikely that the system will damaged by an improperly set switch, but it will not perform as you wish.

Select Code Switches

Switch segments 8, 9, and 10 of switch S1 are preset at the factory for select code 7 as follows:

- Switch segment 8 (SC2) is set to **1**.
- Switch segment 9 (SC1) is set to **0**.
- Switch segment 10 (SC0) is set to **0**.

The **0** and **1** positions are labeled on the printed-circuit assembly or back panel of the computer.

Select codes 3 through 10 may be set with these three switch segments. To change or verify the factory setting, orient the assembly as shown in figure 3-2 and locate switch segments 8, 9, and 10. Next identify the **0** and **1** switch positions on the assembly. You may verify that select code 7 is properly set by comparing the actual positions of switch segments 8, 9, and 10 with those illustrated. They should be the same.

To change the select code, refer to the table and set switch segments 8, 9, and 10 as required for the select code chosen. For example, if select code 3 is to be set, the three switch segments must all be set to **0**. In this case, if the switches are the slide type, slides 8, 9, and 10 must all be set to the **0** position; if they are the rocker type, all three rockers must be pressed down toward the **0** position.

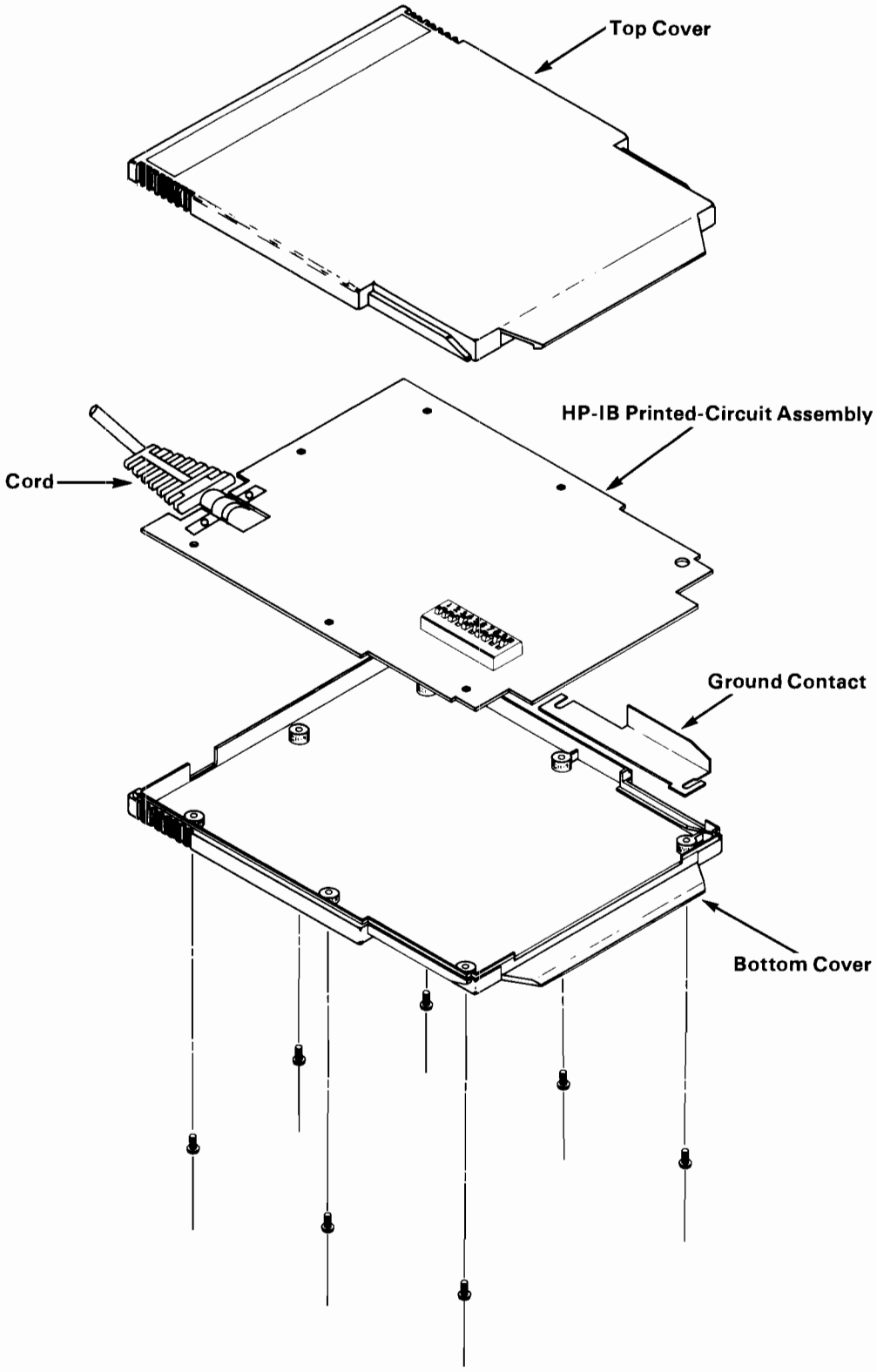
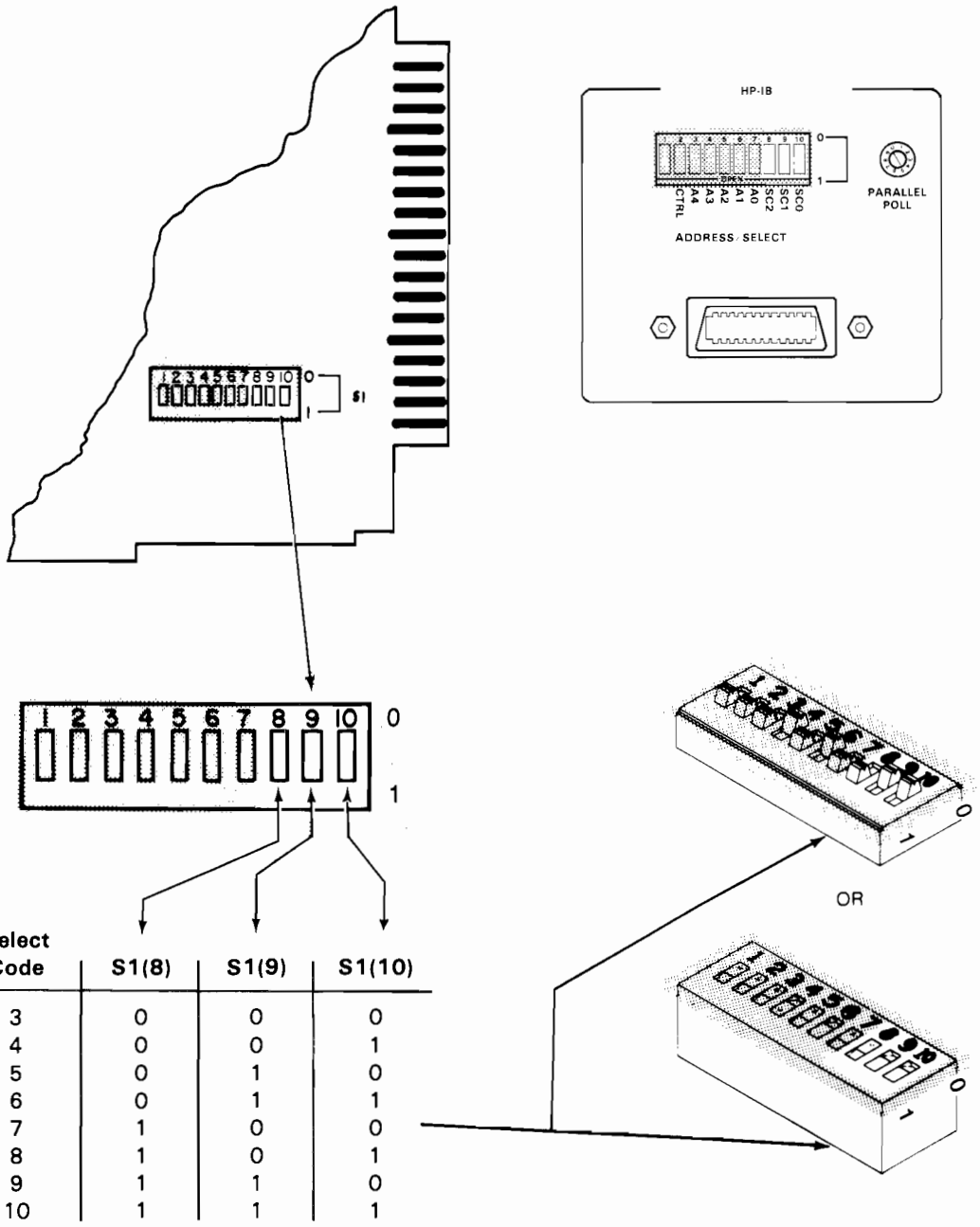


Figure 2-6. Disassembly



Select Code	S1(8)	S1(9)	S1(10)
3	0	0	0
4	0	0	1
5	0	1	0
6	0	1	1
PRESET → 7	1	0	0
8	1	0	1
9	1	1	0
10	1	1	1

Figure 2-7. Select Code Switches

Note: Select codes 1 and 2 are reserved for the CRT and (internal) printer, respectively.

Talk/Listen Address Switches

The HP Series 80 Personal Computer has a device number, just like any other device on the interface bus. It is called the "talk/listen" address so that the computer may specify itself as either the "talker" or the "listener"—to either send information to or receive information from a peripheral device.

The talk/listen address need not be changed unless you have connected two or more HP Series 80 Personal Computers together. Every device on the HP Interface Bus must have a unique device number. Do not set a peripheral address to 21 if your computer talk/listen address is 21.

Switch segments 3 through 7 of switch S1 are preset at the factory for talk/listen address 21 as follows:

- Switch segment 3 is set to **1**.
- Switch segment 4 is set to **0**.
- Switch segment 5 is set to **1**.
- Switch segment 6 is set to **0**.
- Switch segment 7 is set to **1**.

The **0** and **1** positions are labeled on the printed-circuit assembly or the back panel of the computer (for the built-in interface).

Any talk/listen address from 0 through 30 may be set with these five switch segments. To change or verify the factory setting, orient the assembly as shown in figure 2-8 and locate switch segments 3 through 7. **Next identify the 0 and 1 switch positions on the circuit board.** You may verify that talk/listen address 21 is properly set by comparing the actual postions of switch segments 3 through 7 with those illustrated in figure 2-8. They should be the same.

To change the factory setting, refer to the table and set switch segments 3 through 7 as required for the talk/listen address chosen. For example, if talk/listen address 30 is chosen, switch segments 3, 4, 5, and 6 must be set to **1** and switch segment 7 set to **0**. In this case, if the interface is equipped with slide switches, move the slides for segments 3, 4, 5, and 6 to the **1** position and the slide for segment 7 to the **0** position. If they are rocker switches, press the rockers down toward the **1** position for segments 3, 4, 5, and 6, and down toward the **0** position for segment 7.

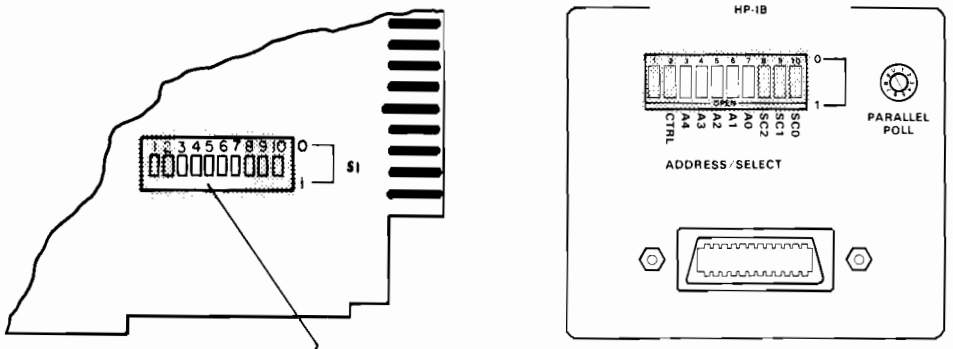
System Controller Switch

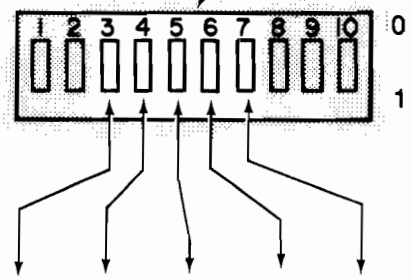
Switch segment 2 of switch S1 is preset to **1** at the factory to enable the HP Series Personal Computer as system controller. The system controller switch need not be changed unless two or more HP Series 80 Personal Computers are connected together. One (and only one) HP-IB bus device can assume system controller status.

If the computer is to be the system controller, leave switch segment 2 in the factory preset position. To verify the factory setting, orient the assembly as shown in figure 2-9 and locate switch segment 2. **Next identify the 0 and 1 switch positions on the printed-circuit assembly or on the back panel of the computer.** Switch segment 2 should be in the **1** position as illustrated.

If an HP Series 80 Personal Computer is *not* to be system controller, set switch segment 2 to the **0** position. In this case, if the interface is equipped with slide switches, move slide segment to the **0** position. If they are rocker switches, press rocker segment 2 down toward the **0** position.

It is important that one (and only one) bus device be assigned as system controller. Failure to comply with this will result in improper bus operation. In operation the system controller may transfer control to another device, but to start operations, one device must be assigned as controller.





Address	S1(3)	S1(4)	S1(5)	S1(6)	S1(7)
0	0	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
4	0	0	1	0	0
5	0	0	1	0	1
6	0	0	1	1	0
7	0	0	1	1	1
8	0	1	0	0	0
9	0	1	0	0	1
10	0	1	0	1	0
11	0	1	0	1	1
12	0	1	1	0	0
13	0	1	1	0	1
14	0	1	1	1	0
15	0	1	1	1	1
16	1	0	0	0	0
17	1	0	0	0	1
18	1	0	0	1	0
19	1	0	0	1	1
20	1	0	1	0	0
PRESET → 21	1	0	1	0	1
22	1	0	1	1	0
23	1	0	1	1	1
24	1	1	0	0	0
25	1	1	0	0	1
26	1	1	0	1	0
27	1	1	0	1	1
28	1	1	1	0	0
29	1	1	1	0	1
30	1	1	1	1	0

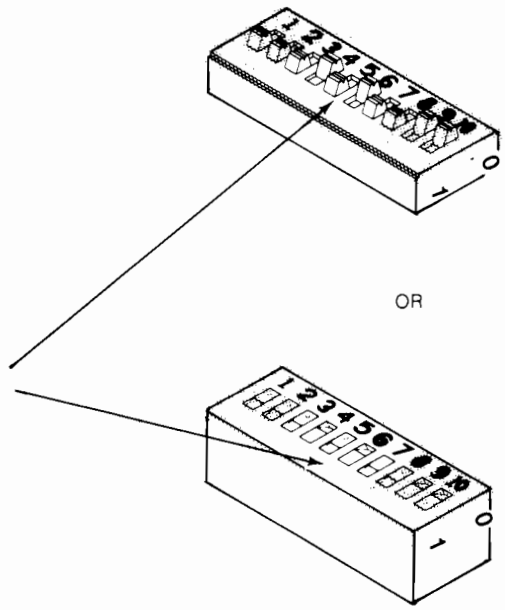
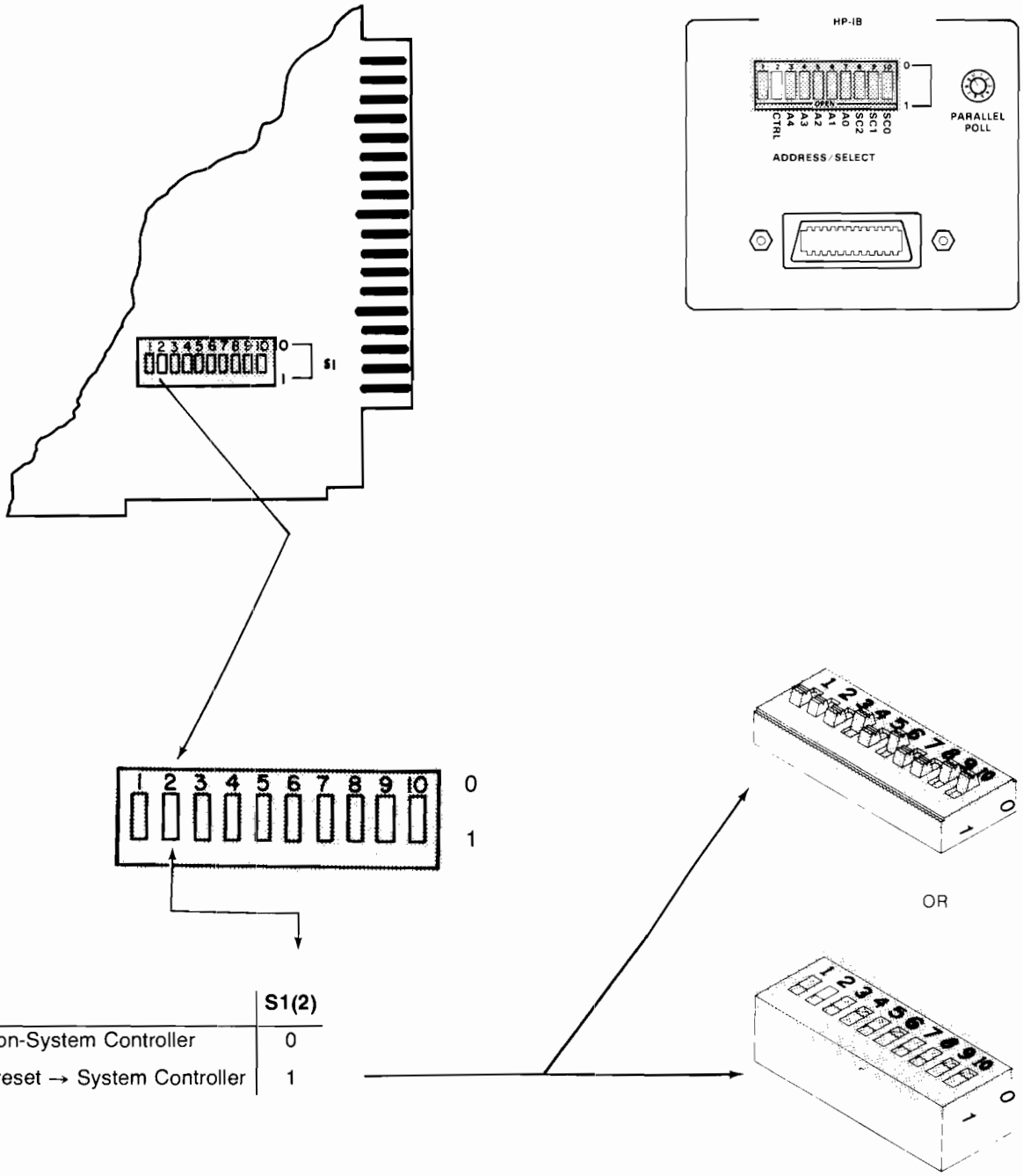


Figure 2-8. Talk/Listen Address



	S1(2)
Non-System Controller	0
Preset → System Controller	1

Figure 2-9. System Controller Switch

Parallel Poll Response Line

External Interface

A jumper wire on the interface circuit board allows the interface to respond to a parallel poll by configuring the interface to one of the DIO lines. The interface is delivered with the jumper installed to assign DIO1 as the parallel poll response line.

The jumper is not necessary when the host HP-85 is the controller directing parallel polls to other HP-IB bus devices. It is necessary when the host HP-85 is being interrogated by another controller for a parallel poll response. In this case, the interface responds to the poll by pulling the DIO line it is configured to if it is requesting service.

The jumper is located underneath the cable wires. If it is necessary to change the DIO response line, first refer to figure 2-10 and remove the cable connector. Remove the cable connector by prying it away from the circuit board connector as shown; do not pull on the cable wires to remove the connector. To change the parallel poll response line, refer to table 2-1, locate the DIO line you want to configure and identify the two holes the jumper must be configured between. Then unsolder and remove the jumper wire using a soldering iron with a 25 watt (or less) rating. Next refer to figure 2-11 and identify the two holes that you want to add the jumper to. The odd-numbered holes are labeled on the component side of the circuit board; the even-numbered holes are labeled on the circuit side. Use the jumper you removed and solder in place between the two appropriate holes using rosin core solder. When you are finished, check both sides of the circuit board to ensure there are no bridges between the circuit board traces and reconnect the cable connector.

Table 2-1. Parallel Poll Jumper Placement

	DIO Line	Place Jumper From	To
PRESET →	DIO1	E1	E2
	DIO2	E3	E4
	DIO3	E5	E6
	DIO4	E7	E8
	DIO5	E9	E10
	DIO6	E11	E12
	DIO7	E13	E14
	DIO8	E15	E16

After all the switches are set and the parallel poll jumper is installed to meet your needs, reassemble the interface before attempting to connect it to the computer.

Internal Interface

For the built-in interface a rotary switch has been provided on the back panel of the computer to allow easy changing of the parallel poll response line setting. The computer has been preset to DIO1. With a small flat blade screwdriver, the switch may be rotated to any position from 1 to 8, corresponding to DIO1 to DIO8.

> **Note:** This switch is used only in configurations in which the I/O ROM is installed and the computer is not the controller. Do not change this setting unless you are familiar with the operation of the HP-IB interface.

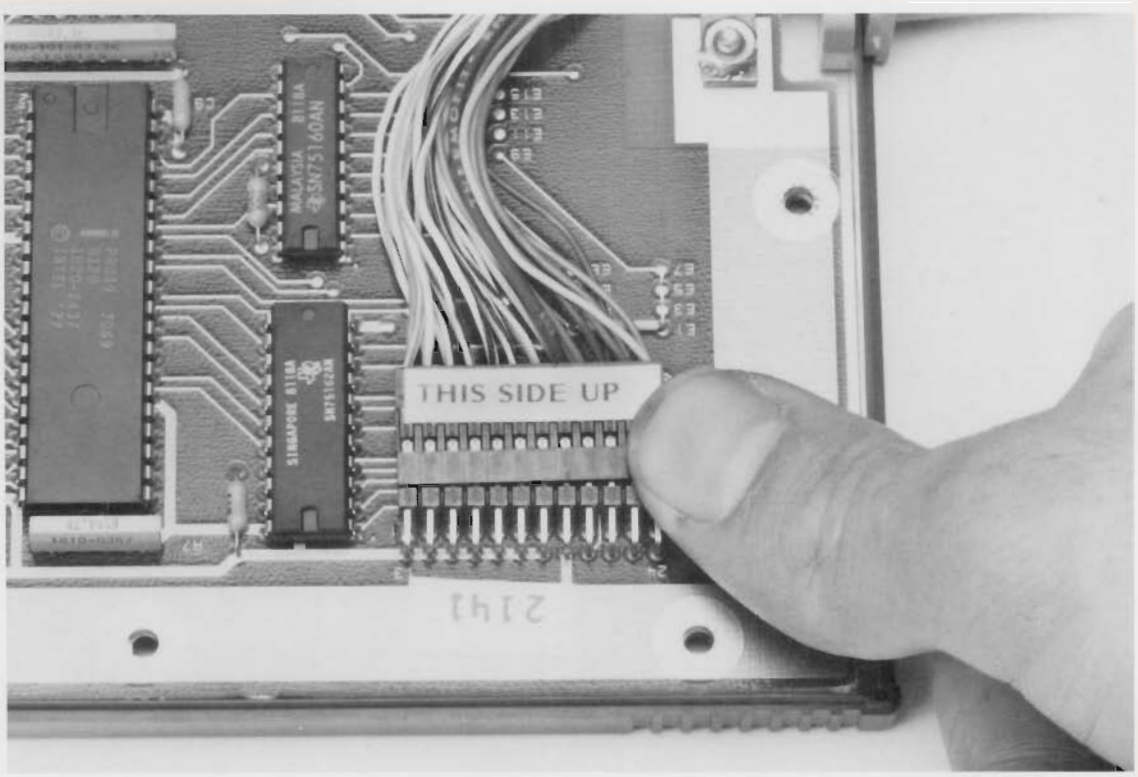


Figure 2-10. Removing Cable Connector

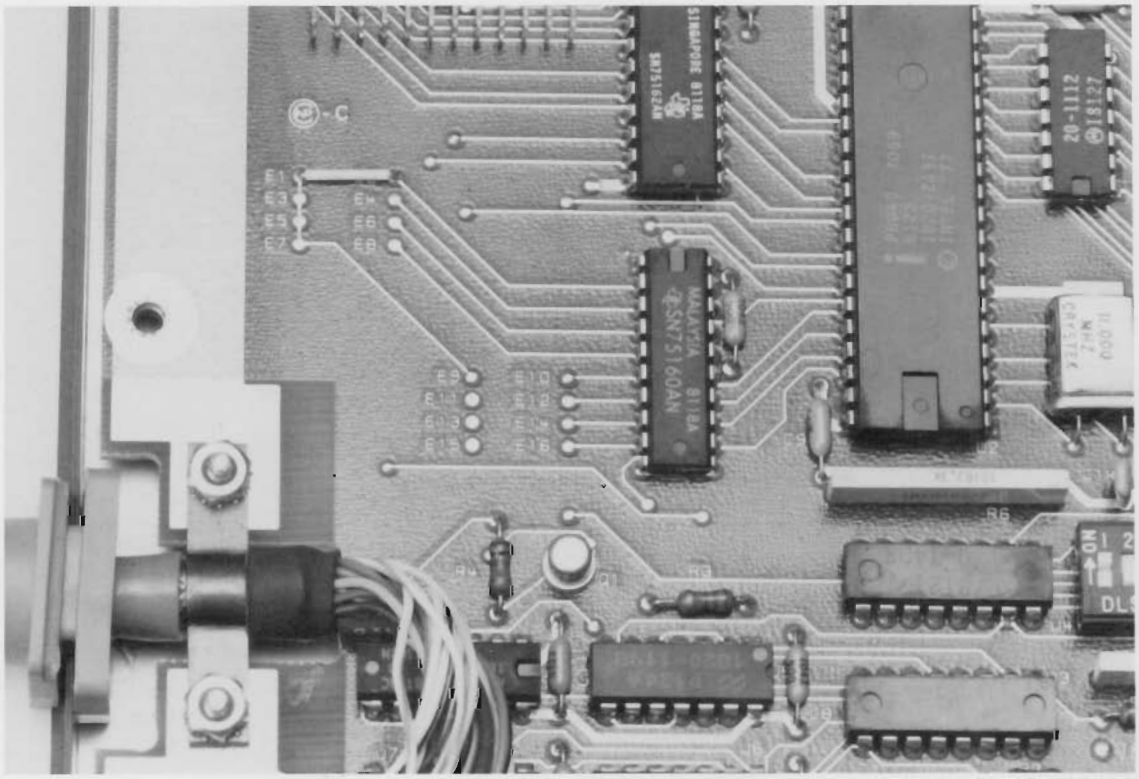


Figure 2-11. Parallel Poll Jumper Wire Placement

Section 3

Using Your HP-IB Interface

Introduction

This section is organized into three parts:

1. A brief summary of HP-IB interfacing for systems that do not contain an I/O ROM.
2. HP-IB operations which require the statements provided by the I/O ROM.
3. HP-IB for the specialist.

It is not necessary for you to read all sections of this manual to use the HP-IB interface. In fact, if you have read the installation section and you have a Printer/Plotter ROM and you just wish to use a printer for listings and output, you need not read beyond the next paragraph. To use your printer you simply need to:

1. Install the interface (if it is of the plug-in type) with the power OFF!
2. Plug the interface cable connector into the connector on your printer.
3. Write down your printer's address (it is usually in your printer manual).
4. Write down your interface select code (factory set to 7).

Assuming that your printer address is 01 and that your interface select code is 7, you turn power on (peripherals first) and execute

```
PRINTER IS 701
```

Now when you PLIST a program, the listing will be printed on your HP-IB printer. Any PRINT statements will also direct output to the HP-IB printer.

This procedure is similar for other types of peripherals such as plotters and disc storage units. Refer to the Printer/Plotter ROM and Mass Storage ROM owner's manuals for examples and capabilities provided by those ROMs. The HP-87 owner's manual also has many examples for use with HP-IB devices.

If you have just had your appetite whetted by the preceding discussion or you need to know more of the capabilities of the HP-IB interface when used with the I/O ROM, read further. The next pages occasionally require a basic understanding of HP-IB concepts. If the HP-IB interface is completely new to you, you might find it helpful to read *A Basic Introduction to the HP-IB* in appendix B.

HP-IB I/O Operations

The HP-IB interface provides both simple operations and complex control functions for systems ranging from a single device plus controller to multiple devices with one or more controllers. To use the majority of these functions, an I/O ROM is required. (The following discussions assume that you have an I/O ROM

in your system). This section is written for the user who understands the terms and structure of the HP-IB, but who needs information on how it is used. This is not an extensive treatment of statements and their syntax, but rather is intended to help the user understand the sequences of operations necessary to make an HP-IB system function. Use this section along with the *I/O ROM Owner's Manual* to design your program, then use the syntax reference in the back of that manual to help code your program.

Turn-on and Check-out

The topic of device installation is covered in detail in section 2, and is not covered here except in general terms.

Once the interface has been installed and the system devices have been connected via the HP-IB cables, a short routine should be run to check the status of the individual components. If you know the bus address of the devices in your system, a program similar to the one below can be used.

```
10 STATUS 7,0 ; A,B,C,D,E,F
20 PRINT "Interface card status = ";A,B,C,D,E,F
30 A=SPOLL(722)
40 B=SPOLL(703)
50 C=SPOLL(713)
60 PRINT "Device #22 status = ";A
70 PRINT "Device #03 status = ";B
80 PRINT "Device #13 status = ";C
90 END
```

In line 10, the status information of the HP-IB interface itself is obtained, and that is printed in line 20. The status value as set at the factory should be 1, 0, 64, n, 53, 160. Lines 30-50 obtain and print the *serial poll response byte* of the devices set to addresses 22, 03, and 13. The value "n" will vary.

If you don't know, or aren't sure what the system device addresses are, the following program segment could be substituted:

```
10 S=7 @ ! VARIABLE S IS SELECT CODE
20 SET TIMEOUT S;500
30 ON TIMEOUT S GOTO 100
40 FOR I=0 TO 31
50 DISP "SPOLL DEVICE # ";I
60 S1=SPOLL(S*100+I)
70 PRINT "DEVICE ";I;" NOT PRESENT"
80 NEXT I
90 STOP
100 ABORTIO 7
110 PRINT "DEVICE ";I;" NOT PRESENT"
120 GOTO 80
130 END
```

Any device present should return some value for its serial poll response, and this will show up on the printout. Please note that the status information returned by each device is normally different. The meanings of the device status information printed by the above routine can be found in the operating

manual for each device, and would typically be termed Serial Poll Response Byte. Different devices may have this information in different areas of the manual, but the key to finding the information is Serial Poll Response.

Now that you have checked out the operation of your system, you can spend some time learning how to control it.

Controlling the Bus

All the operations in this section assume the HP Series 80 Personal Computer and HP-IB interface are set up as system controller (as received from factory). Typically, the first operation necessary for HP-IB systems is to program all devices for **remote** operation via the bus. Most devices are capable of manual, front panel operation or of remote, bus operation. The remote mode of operation for a device is selected by setting the REN, or **Remote Enable** line, and addressing the device to listen. The REN line is set by the computer when power is turned on, the computer is (RESET), or the REMOTE statement is executed. Addressing the device to listen is performed by executing any statement which includes that device's listen address. To place devices 22 and 10 under remote control, the statement

```
10 REMOTE 722,710
```

could be used. To prevent any of the system devices from being returned to local operations from the front panel (by the Return-to-Local switch) a **local lockout**, or LLO message must be sent. The example cited above now looks like this:

```
10 REMOTE 713,722
20 LOCAL LOCKOUT 7
```

Now the system is set up for remote control, with the front panel controls disabled. The next step then is to program each device for the desired mode of operation. This is generally accomplished by means of simple OUTPUT statements directed to the device to be programmed, which is discussed further in the following section. For instance, suppose an HP 3455A Digital Voltmeter is to be set to the 0.1 volt dc range, continuous sample mode. The ASCII sequence of characters necessary to set is to this range is "F1R1T1." The following statement accomplishes this:

```
50 OUTPUT 722 ;"F1R1T1"
```

Output to device #22, select code 7, the ASCII characters F1R1T1.

It is important to note that it is the **ASCII characters** sent by the OUTPUT statement which actually set the HP 3455A Digital Voltmeter to the 0.1 Vdc range. Selecting some other function merely involves changing the characters being sent to the device to the appropriate ones to select that function. The actual characters sent are "device-dependent," that is, their meaning depends upon the interpretation given them by the receiving device.

The example system consists of two devices, so both may need to be programmed, as shown below:

```
10 REMOTE 722,713
20 LOCAL LOCKOUT 7
30 ! Select .1vdc range on device 22 (3455A voltmeter)
40 OUTPUT 722 ; "F1R1T1"
50 ! Select 1K hz resolution on device 13 (freq. counter)
60 OUTPUT 713 ; "PF4G3S1S3S5T"
```

Once the system devices are programmed for operation, it is possible to take readings from them. This is accomplished by means of the ENTER statement, which addresses the specified device and accepts data from it. For example, to take a voltage reading and a frequency reading, the following statements could be used:

```
10 REMOTE 713,722
20 LOCAL LOCKOUT 7
30 ! Select .1vdc range on device 22 (3455A voltmeter)
40 OUTPUT 722 ; "F1R1T1"
50 ! Select 1K hz resolution on device 13 (5328A freq.
   counter)
60 OUTPUT 713 ; "PF4G3S1S3S5T"
70 ! Take reading from voltmeter and place into V
80 ENTER 722 ; V
90 ! Take reading from counter and place into F
100 ENTER 713 ;F
110 PRINT "Voltage =";F
120 PRINT "Frequency =";F
130 END
```

The example as presented so far simply obtains the most recent readings taken by the voltmeter and counter. Suppose your particular application requires that the two readings occur **simultaneously**. An example of such an application is that of a voltage-to-frequency converter accepting a continuously varying voltage and generating an output frequency dependent upon the input voltage. To take both a voltage and a frequency reading simultaneously, the instruments can be triggered.

To accomplish this, it is necessary to change the operating modes of the devices from continual sample to triggered sample. The OUTPUT statements are changed accordingly, and a TRIGGER statement is inserted just before the ENTER statements. The TRIGGER and ENTER statements are put into a loop to provide a more complete picture of the converter's operation.

```
10 REMOTE 713,722
20 LOCAL LOCKOUT 7
30 ! Select .1vdc range on device 22 (3455A voltmeter)
40 OUTPUT 722 ;"F2R2T2T3"
50 ! Select 1K hz resolution on device 13 (5328A freq.
   counter)
60 OUTPUT 713 ;"PF4G3R"
```

```
70 FOR I=1 TO 100
80 TRIGGER 713,722@ RESUME 7 @ ! Drop ATN line.
90 ! Take reading from voltmeter and place into V
100 ENTER 722 ; V
110 ! Take reading from counter and place into F
120 ENTER 713 ; F
130 PRINT "Voltage =";F
140 PRINT "Frequency =";F
150 NEXT I
160 END
```

The ENTER and OUTPUT statements have been presented briefly as a means of programming instruments and taking readings. The next section deals with the more general topic of HP-IB data transfers.

HP-IB Data Transfers

Introduction

The data transfer*, or the data message as it is known by HP-IB specialists, can be either the simplest or the most complex function available to the user. This section will show the simplest transfers first, then proceed to more complex transfers (using the TRANSFER statement provided by the I/O ROM).

Simple Output Operations

To send data from the computer to a device on the HP-IB, the OUTPUT statement is generally used. It is also possible, however, to use the PRINT statement to send data to an HP-IB device. To do this requires executing a PRINTER IS statement to change the system printer over to the specified HP-IB device. The PRINTER IS statement can be inserted into the beginning of a standard BASIC program, then all the PRINT statements in the program will send their output to the specified device until a new PRINTER IS statement is executed. If only one HP-IB device is being sent data, this scheme works fine. For example,

```
10 ! The HP-IB printer has a device address of 01
20 PRINTER IS 701
30 FOR I=0 to 1 STEP .01
40 PRINT SIN(I)
50 NEXT I
60 END
```

This method becomes cumbersome when several devices are to be sent data, such as when programming a DVM (digital voltmeter) and a counter.

*This is the general term for a data exchange—it does not relate specifically to TRANSFER, an advanced I/O capability.

```
10 PRINTER is 722
20 ! Address data output to DVM
30 PRINT "programming sequence"
40 PRINTER IS 713
50 ! Address data output to Counter
60 PRINT "programming sequence"
70 END
```

Obviously, this is not the method of choice. Compare the above program sequence to the following one, which does the same thing.

```
10 OUTPUT 722 ; "Programming sequence"
20 OUTPUT 713 ; "Programming sequence"
```

The OUTPUT statement data can be formatted with the OUTPUT USING statement, as shown in the *I/O ROM Owner's Manual.* The HP-IB version of this statement looks like this:

```
OUTPUT 722 USING "nA":"program sequence"
```

In general, a simple HP-IB output looks very much like any other output, except that additional information in the form of the HP-IB device address must be specified with the select code. If you need more information about using the OUTPUT statement, you can find it in the *I/O ROM Owner's Manual.* More advanced techniques of outputting data are covered in the HP-IB for the Specialist section.

Simple Input Operations

The ENTER statement is the simplest method of inputting data from an HP-IB device to the HP Series 80 Personal Computer. To accomplish a data input from a device such as an HP 5307A Universal Time Interval Counter, the following statement can be used:

```
ENTER 716;A$
```

In this case, the statement is terminated by a line feed sent by the counter as the last character. Other devices may send **EOI** (End or Identify) with the last character to terminate a data input. If this is the case for your device, the ENTER statement could be modified as shown below for device #06:

```
ENTER 706 USING "%,K";A$
```

The "%" image specifier allows the EOI message* sent by the device to terminate the ENTER statement. However, if a line feed character is part of the data being sent, the line feed will prematurely terminate the ENTER statement shown above. The following statement waits for the EOI message before terminating (does not terminate on line feed):

```
ENTER 706 USING "#%,K";A$
```

If you need more information regarding input data formatting, please refer to the *I/O ROM Owner's Manual* where the ENTER statement is covered in greater detail. For information pertaining to more flexible data input transfers, or data input to the computer when it is a non-controller, refer to the following section on Advanced I/O Operations.

Advanced I/O Operations

Introduction

If you are reading this section, it is probably because you have a problem that can't be solved with a simple OUTPUT or ENTER statement. Basically, this section deals with four types of transfers:

1. Data transfers involving a non-controller HP Series 80 Personal Computer.
2. Enters and outputs that include multiple listeners.
3. Data transfers initiated by the computer which do not include the computer.
4. User-specified transfer types, including interrupt and fast-handshake.

Non-controller Adressing

This type of I/O is used when either the computer has passed control to another device and is no longer active controller, or when the HP-IB interface has been set to non-system controller (set by a switch in the interface) and has not received active control from the current active controller.

When the computer (or **any** device, for that matter) is not the active controller, it must wait to be addressed to talk or listen before it may output or input data on the bus. There are three means of accomplishing this waiting to be addressed. This first is automatic and would be the method of choice for a simple system. Simply specify the interface select code with no device numbers in the ENTER or OUTPUT statement, and computer will wait to be addressed before transferring the data. The following statements show the conditions necessary to begin a transfer:

```
      ENTER 7;A$
or TRANSFER 7 TO A$...
```

Wait to be addressed to **listen** by the controller, then read data into A$.

```
      OUTPUT 7;A$
or TRANSFER A$ TO &...
```

Wait to be adressed to **talk** by the controller, then send A$.

A second method of waiting to be addressed is to periodically check the interface status for the proper condition. The computer could be accomplishing some useful computation during the period of time it is

*The EOI message here is the END message, sent by some devices with the last character of a data exchange. It indicates the end of data.

not addressed, then when a status check does finally indicate that the computer has been addressed, it could transfer the requested data. For example, the following program increments and displays a counter, then checks to see if it is addressed. If not, the program loops back to the increment and display statements. Otherwise, the value of the counter is transferred to the HP-IB.

```
10 ! Lines 50,60 check for addressed to talk
20 I=0
30 I=I+1
40 DISP I
50 STATUS 7,5 ; S
60 IF BIT(S,4)=0 THEN GOTO 30 ! Is "TA" bit set?
70 OUTPUT 7 ; I
80 GOTO 20
90 END
```

(See HP-IB for the Specialist section for a complete explanation of the status registers.)

The sequence shown above to read status and check for being addressed to talk is identical for an ENTER operation. The difference is in which status bit is to be checked, and is shown in the following statements:

```
10 STATUS 7,5 ; S
20 IF BIT(S,6)=0 THEN DISP "Not addressed to listen"
```

Bit 6 of Status Register 5 indicates addressed to listen when it is set to a 1. Bit 4 of Status Register 5 indicates addressed to talk when it is set to a 1.

The third method of waiting to be addressed is to enable the interface to **interrupt** the computer when it is addressed. This relieves the programmer of the necessity of designing his program around a periodic status check, and it allows the program to perform useful computation until the computer is addressed to talk or listen. An example serves to demonstrate this third method:

```
10 ! Set up "addressed-to-listen" interrupt and subsequent
   end-of-line branch
20 ON INTR 7 GOSUB 100
30 ENABLE INTR 7;64
40 ! Line 30 enables end-of-line branch service routine
50 I=0
60 I=I+1
70 DISP I
80 GOTO 60
90 ! Service routine for end-of-line branch
100 STATUS 7,1 ; S
110 ENTER 7 ; A$
120 PRINT "Data received was ";A$
130 ! Re-enable "addressed-to-listen" interrupt
140 ENABLE INTR 7;64
150 RETURN
160 END
```

Line 20 sets up an interrupt condition of listener active (or addressed-to-listen). Line 40 directs the program to line 100 when the listener active interrupt occurs. Lines 60-80 represent the computational portion of the program, purposefully kept simple here. Line 100 starts the end-of-line-branch service routine, which is executed when the computer becomes addressed to listen. The ENTER statement specifies only the interface select code (no device numbers!) which means "wait until addressed to listen before inputting data." Line 150 re-enables the interface for an addressed-to-listen interrupt. Output works the same way, except a different value (16) is used for the ENABLE INTR mask, and an OUTPUT 7;<*value*> statement is substituted for the ENTER statement. See the HP-IB for the Specialist section for a complete explanation of the control registers, and interface interrupts as discussed under Status Register 1 of the same section.

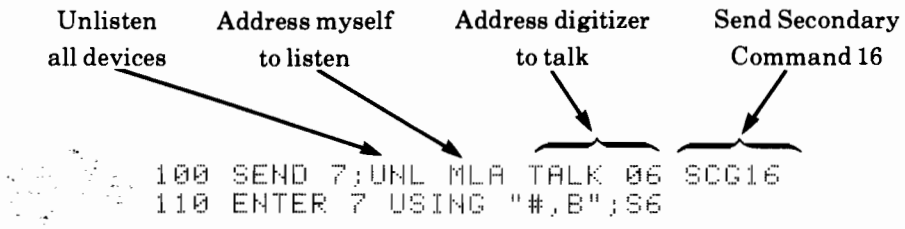
Suppose our non-controller computer is to both send and receive data. The basic sequence of operations remains the same but an additional status check is necessary to determine which operation to perform. The following program illustrates this status check. Line 100 obtains the status value and lines 110 and 120 determine if the cause of an end-of-line branch was either for being addressed to talk or for being addressed to listen.

```
10 ! Enable talker active and listener active interrupts
20 ON INTR 7 GOSUB 100
30 ENABLE INTR 7;(64+16)
40 I=0
50 I=I+1
60 DISP I
70 GOTO 50
80 !      Service routine follows
90 ! Check status to determine the cause of the end-of-
   line branch
100 STATUS 7,1 ; S ! Read interrupt cause
110 IF BIT(S,4)=1 THEN GOTO 150
120 IF BIT(S,6)=1 THEN GOTO 180
130 PRINT "Error in status code"
140 RETURN
150 ! Service for "addressed to talk"
160 OUTPUT 7 ; I
170 GOTO 210
180 ! Service for "addressed to listen"
190 ENTER 7 ; A$
200 PRINT "Received data of ";A$
210 ENABLE INTR 7;(64+16)
220 RETURN
```

Custom Bus Sequences

Occasionally it may become necessary to send a custom bus sequence to a device under development or to one which requires a sequence different from those normally sent by the computer. The SEND statement makes such custom operations possible, and even gives a performance increase in the bargain. However, the price you pay is that you have to specify every character of the sequence yourself: its no longer done automatically.

As one example of a custom bus sequence, the following statements send a Secondary Command to an HP 9874 Digitizer (device 06) on select code 7. The Secondary Command of 16 causes the digitizer to output its status which is read as one byte by the following ENTER statement (notice that ENTER does **not** specify addressing).



Other operations that could be performed include Parallel Poll Configure and Parallel Poll Unconfigure. You can utilize the tables included in HP-IB for the Specialist to determine the necessary codes to accomplish your task.

Multiple Listener Transfers

When more than one device needs to receive data being transferred, these additional devices need to be included in the **listen address group**. This is accomplished in different ways for ENTER and for OUTPUT transfers as described in the following paragraphs.

The basic procedure is to specify all listeners and the talker with a SEND statement. An output or enter operation is then performed but only the select code is specified. Note that all talkers and listeners are required to be on the same select code address. Assume that a string of characters (B$) is to be sent to a printer (device 04) and a non-controller HP Series 80 Personal Computer (device 20). The sequence of program statements would look like this:

```
SEND 7; UNL LISTEN 4,20
OUTPUT 7; B$
```

The TRANSFER statement would work similarly, with the OUTPUT replaced by:

```
TRANSFER B$ TO 7 INTR
```

Specifying other listeners as well as the talker is possible for an input operation also. The bus needs first to be configured for the transfer, then an ENTER statement can be executed with only the interface select code specified so the bus will not be reconfigured. For example, device 07 (a voltmeter) is to be the talker, devices 13 (a printer), our computer (use MLA which means My Listen Address), and 04 (a tape cartridge unit) are to be listeners. The following sequence unlistens all previous listeners, sends the new talk and listen addresses, and finally enters the data:

```
10 SEND 7 ; UNL TALK 7 LISTEN 13,4 MLA
20 ENTER 7 ; V
```

A similar sequence could be used for an input transfer, in this case an interrupt transfer into T$:

```
10 DIM T$[88]
20 IOBUFFER T$
30 SEND 7 ; UNL TALK 7 LISTEN 13,4 MLA
40 TRANSFER 7 TO T$ INTR
```

Another type of multiple listener transfer is one in which the computer is neither sending nor receiving the data: the transfer occurs between other devices on the bus. For example, to send a data file from device 11 to devices 23, 04, and 07, the following statements would be used:

```
10 SEND 7; UNL TALK 11 LISTEN 23,04,07
20 RESUME 7
```

The transfer begins automatically when the ATN (attention) line goes false (by executing RESUME), indicating that all commands have been sent. In this case, since the computer has not addressed itself for the transfer, the operation does not have to wait for the computer to execute an ENTER or OUTPUT statement in order to begin. Obviously, it is not necessary to specify more than one listener—the same statement could be used to start a transfer between one talker and one listener as well.

There is a problem with the above example: how does the computer know then the data transfer is complete? The simplest manner is to address the computer as a listener in line 10, execute an input interrupt TRANSFER to monitor the bus transfer, and wait for an end-of-line branch upon termination of the TRANSFER. The example below shows how this might be done.

```
5 DIM T$[1000]
6 IOBUFFER T$ @ ON EOT 7 GOSUB 1000
10 SEND 7 ; UNL TALK 11 LISTEN 23,4,7 MLA
20 ! For this example, EOI message ends the transfer.
30 TRANSFER 7 TO T$ INTR ; EOI
40 RESUME 7
50 ! Note that RESUME causes ATN to drop which starts the
   transfer.
```

Alternate Transfer Types

As has been shown in examples cited in the previous sections, it is possible to transfer data under interrupt (which allows transfers and program execution to occur at the same time) or by fast handshake (maximum data transfer rate possible). The TRANSFER statement is used to specify which type (INTR

or FHS) is to be used. This statement is discussed in the *I/O ROM Owner's Manual* in the Advanced Transfers section, and details on its use are available there. The following program demonstrates how an HP-IB transfer can be executed:

```
10 ! Transfer data under interrupt from device 04 to
   buffer T$
20 ! A line-feed terminates the transfer.
30 DIM T$[88]
40 I=0
50 IOBUFFER T$
60 ON EOT 7 GOSUB 120
70 ! Now configure the transfer
80 TRANSFER 704 TO T$ INTR ; DELIM 10 ! Line-feed
   terminates.
90 I=I+1
100 DISP "COUNTING",I
110 GOTO 90
120 PRINT "Received data",T$
130 ! Set up another transfer
140 IOBUFFER T$ ! Clear the buffer
150 TRANSFER 704 TO T$ INTR ; DELIM 10 @ RETURN
160 END
```

Handling Service Requests

Introduction

This section deals with asynchronous requests for service. The cause for a service request is device-dependent, that is, different devices have different reasons for requesting service. For instance, a printer may request service because it has just run out of paper, or a digitizer may request service because an operator error has occurred. At any rate, once a request has been received, two actions must be taken:

1. Locating the device which requested service, and
2. Determining the reason for the device's request.

The following sections show you how to do this.

Sensing Service Requests

The HP-IB interface allows you to check the interface to see if a service request (SRQ) has been received. This status check is performed by the following statement:

```
STATUS 7,2;S
```

Bit 5 of variable S now indicates if an SRQ has been received, and the program can make a decision of how to handle an SRQ if one has been received. (See HP-IB for the Specialist section for a complete description of the status registers.) If interrupts are enabled then we can perform the following statment to determine if an SRQ has been received (and an interrupt has occurred):

```
STATUS 7,1;S
```

Bit 3 of S indicates if an SRQ was received. An alternative to the periodic status check is to program for an end-of-line branch condition of SRQ. The following sequence enables an end-of-line service routine, then establishes an interrupt mask for SRQ and enables interrupts from the interface.

```
10 ON INTR 7 GOSUB 200
20 ! The value 8 sets bit 3 of the control register (SRQ)
30 ENABLE INTR 7;8
```

The program will execute a subroutine at line 200 whenever an SRQ is received. What to do at line 200 is the subject of the following sections.

Determining the Problem

It is the purpose of the HP-IB serial poll to provide the active controller with specific, device-dependent information about the device being polled. The bits of the device's serial poll response byte can have any meaning assigned to them and are generally used to indicate some problem or special condition within the device. Bit 6 however, is reserved to indicate that the device is currently requesting service.

Referring to the example below, line 300 is executed when the program has determined that device 15 requested service. The first operation to perform in servicing device 15 is to obtain its status. Assume that the bits are assigned the following meanings:

Bit 0: Out of paper.
Bit 1: Cover off.
Bit 2: Parameter out of range.
Bit 3: Improper escape sequence.
Bit 4: Always 0.
Bit 5: Always 0.
Bit 6: SRQ Active.
Bit 7: Always 0.

Device 15's service routine might look like the following:

```
300 ! Service for device 15
310 S=SPOLL(715)
320 IF BIT(S,6)=1 THEN GOTO 350
330 ! Else SRQ was not this device.
340 RETURN
350 IF BIT(S,0)<>1 THEN GOTO 390
360 PRINT "HP-IB printer out of paper"
370 PAUSE
380 RETURN
390 IF NOT BIT(S,1) THEN GOTO 420
400 ! And so forth
410 ! Re-enable interrupts if necessary.
```

In a larger system, the program would just perform sequential polls to determine which device requested service, and to determine that device's current status, as shown briefly below:

```
100 S=SPOLL(715)
110 IF NOT BIT(S,6) THEN GOTO 200
120 ! Lines 120-190 service device 15
200 S=SPOLL(723)
210 IF NOT BIT(S,6) THEN GOTO 300
    :
```

Non-controller Operations

When the computer is not the active controller, certain constraints are forced on the programmer to avoid violating bus protocols. That is, certain operations can only be performed by the system controller, others only by an active controller. A non-controller is only allowed to talk, listen, and request service.

Passing Control

The computer, as active controller, can pass controller responsibilities to another device by executing the PASS CONTROL statement. This allows the computer to direct its attention to activities other than bus control, such as would be the case when it must direct and coordinate the activities of two or more HP-IB systems, or when it must dedicate all processing power to a high-speed block data transfer to a host computer. The following program segment passes control to device 20, another HP Series 80 Personal Computer, which is currently a non-controller:

```
680 PASS CONTROL 720
```

It may be necessary for our first computer to later assume control of the bus again, in which case some provision must be made to determine that control has been passed. This is discussed in the following section. It should be pointed out that even the system controller can become a non-controller. On a system reset, the device that has been deemed the system controller becomes the active controller again. Only one system controller is allowed (determined by the interface switch settings). All references to non-controllers include both inactive system controllers and inactive non-system controllers. If our system controller has passed control, some provision must be made to determine that control has been passed back. This is discussed in the following section.

Receiving Control

Although a simple check of interface status can tell the program that the computer has received active control, it is more advantageous to use the end-of-line branch facility to do this automatically. The following program segment passes control to device 20, but also makes provision for an end-of-line branch to line 710 when active control is later received:

```
650 ! Pass control subroutine
660 ON INTR 7 GOSUB 710
670 ! ENABLE EOL branch on receiving active control.
```

```
680 ENABLE INTR 7;32
690 PASS CONTROL 720
700 RETURN
710 ! Routine to accept active control goes here.
720 STATUS 7,1 ; S
730 IF BIT(S,5) 1 THEN GOTO 760 ! Bit 5 is active control
740 ! Branch taken but control was not received. Error?
750 RETURN
760 Control has been received. Do something!
 :
```

This end-of-line branch capability can serve to also indicate that the computer has been addressed to talk or to listen by the active controller.

The following program is used to illustrate the above technique in a simple example providing communication between two HP Series 80 Personal Computers.

PROGRAM #1 (ACTIVE CONTROLLER)

```
10 OUTPUT 720 "IT WORKS!!!"
20 PASS CONTROL 720
30 ENTER 7, A$ ! Non-Controller Addressing.
40 DISP A$
45 OUTPUT 720 ;"RETURNED CONTROL ! Controller Addressing.
50 END
```

PROGRAM #2 (LISTENER)

```
10 A$=""
20 ENTER 7 ; A$
30 DISP A$
35 OUTPUT 721 ;"HI THERE"
36 PASS CONTROL 721
37 ENTER 7 ; B$
38 DISP B$
40 END
```

Program #1 is entered into the device which is to be used as the active controller. Program #2 is entered into the device designated as listener with the device address set to 20 and the HP-IB interface set to non-system controller.

Program #2 is run and the listener waits at line 20 for the controller to address it before data may be transferred. The active controller program is then run; both programs continue execution until both END statements are encountered.

Non-controller Responses

In an earlier discussion it was shown how the HP Series 80 Personal Computer sends and receives data as a non-controller—only the interface select code is specified. No device addresses are allowed! The computer can use the end-of-line branch facility to determine when it has been addressed to talk or listen, and this is shown in the following example. This example is an extension of the previous one, in that control is passed and interrupts are enabled for active control, addressed to talk, and addressed to listen.

```
    :
650 ! Pass control subroutine
660 ON INTR 7 GOTO 700
670 PASS CONTROL 720
680 ENABLE INTR 7;(16+32+64) @ RETURN
690 !
700 ! Non-controller service routine
710 STATUS 7,1 ; S
720 IF BIT(S,5)=1 THEN GOTO 810
730 IF BIT(S,4)=1 THEN GOTO 780
740 IF NOT BIT(S,6) THEN PRINT "ERROR" @ STOP
750 ! Bit 4 = 1 so input data
760 ENTER 7 ; A$
770 GOT 860
780 ! Bit 4 = 1 so output data
790 OUTPUT 7 ; X$
800 GOTO 860
810 ! Bit 5 = 1 so assume control
820 ! Do necessary controller things, then pass control
    again
    :
850 PASS CONTROL 720
860 ENABLE INTR 7;(16+32+64) @ RETURN
```

Additional information on HP-IB interrupts can be found under Control Registers and Status Registers in the HP-IB for the Specialist section.

Sending Service Requests

Some condition in the HP Series 80 Personal Computer may require the attention of the active controller, so the computer needs to be able to send a service request to the controller. The REQUEST statement allows the computer to assert SRQ and to send a serial poll response byte to the controller when it finally conducts a serial poll on the computer. Bit 6 of this byte should be set to a 1 to indicate that the computer indeed requested service, but the other bits may indicate anything you (as the system designer) deem important. So, to request service of device 720 (who we just passed control to) and to indicate that the HP Series 80 Personal Computer is ready with data, our example will set bits 0 and 6. Bit 0 will mean ready-with-data in our example system.)

```
900 REQUEST 7;1 + 64
```

Handling Interface Problems

This section describes some of the tools available to you for avoiding and dealing with problems that may arise when using an HP-IB interface. This is not meant to say that you should **expect** problems, but good programming practice anticipates problems and deals with them in advance.

Avoiding Bus Hang-ups

Generally, when an HP-IB device develops a problem, either it holds up the data transfer that it is involved in, or it sends an SRQ (service request) to the controller, or it does both. We have seen how the controller might handle the service request (see the section called Handling Requests for Service in section 1), but suppose the device stops handshaking in the middle of a data transfer and at the same time

it sends an SRQ. This event presents a problem to the computer because it cannot perform an end-of-line branch to service the SRQ. Why this is so becomes apparent when you consider the nature of an end-of-line branch: it does not occur until the current BASIC program line has been executed, and if an ENTER or OUTPUT still has not completed (the device halted the transfer, remember) the computer is "hung". It cannot complete the transfer, and it cannot execute an end-of-line branch until the operation completes. The computer **can** recover from an unsuccessful transfer, however, by using the SET TIMEOUT capability provided for such a situation.

The following example shows the sequence of operations necessary to provide a program with the capability of recovering from a bus hang-up:

```
10 DIM S(6)
20 ! First set up the timeout service routine
30 ON TIMEOUT 7 GOSUB 170
40 ! Then set a handshake time limit (1200 msec here)
50 SET TIMEOUT 7;1200
60 ! Now program the HP3455 for triggered readings.
70 OUTPUT 722 ; "F1R1T2T3"
80 ! Now trigger and read the DVM
90 TRIGGER 722
100 ENTER 722 ; X
110 DISP X;"volts"
120 GOTO 90
130 !
140 ! Begin timeout service routine
150 ! First step is to check the interface status
160 ON TIMEOUT 7 GOTO 270 ! Make provision for bad
    interface.
170 FOR I=0 TO 6
180 STATUS 7,I ; S(I)
190 PRINT "Status byte #";I;" =";S(I)
200 NEXT I
210 ! The next step is to decide on the course of action
220 ! based upon the status info just obtained...
    :
250 GOTO 290 ! End of timeout service.
260 !
270 PRINT "Interface failure"
280 RESET 7 ! Try to help it.
290 ON TIMEOUT 7 GOSUB 170 ! Restore original service
    rountine.
300 RETURN
```

It is up to the programmer to determine what actions the program is to take based upon the results of the status information obtained. In most instances it will be necessary to RESET the interface, avoid transfers with the device causing the hang-up, and signal to the system operator that a malfunction has occured in that device. These actions are described in the next section.

Dealing with Problems

You are probably reading this section either because your HP-IB system is not working or because you need to make provisions in your program to deal with problems. This section first deals with the case of a non-working system, since it is then that you need this information in a hurry.

If the HP-IB system appears to be locked up (everything is running but nothing is happening) you can perform a (RESET) to regain control of the computer. The exception to this is when a fast-handshake TRANSFER is hung. The only possible recourse to such a case is to assert IFC (interface clear) by whatever means*—or to power down the computer and then power it back up.

You may now wish to perform a serial poll of the device in question to determine its condition. This, however, requires that you know how to obtain the correct information from the device. It may be sufficient to simply turn the device off, then back on to put it into operation again.

The following section shows the alternatives (and reasons for each) available to the programmer writing a routine to handle error conditions encountered during HP-IB operations.

Action	Reason/Result

`FOR I = 1 TO 6`
`STATUS 7,I;S(I)`
`NEXT I`
	Obtain the current state of the interface to determine conditions and possible causes.

`S=SPOLL(`*<each device>*`)`	Obtain current status of devices in the system. (Maybe a printer is out of paper.)
`CLEAR`*<selected device>*	Return the desired device to its particular device-dependent "clear" state. (Like a reset.)
`CLEAR 7`	Return all devices to their device-dependent states.
`ABORTIO 7`	Terminates all bus activity and returns control to the system controller. This would normally be used only by an HP Series 80 Personal Computer that is set to system controller (the original factory setting is that of system controller).

It is a good practice to preserve a hard copy record of the interface and device status as the information is obtained. Also printed should be a record of any actions taken to rectify the situation. This information is vital for analyzing the cause of a system failure, and can be used by the system operator to make necessary adjustments or corrections (such as loading paper in the printer!)

HP-IB Limitations

Before leaving this brief overview of the HP-IB, some of the limitations of the HP-IB should be considered. The first limitaion is that a maximum of 15 devices may be connected together by one HP-IB. This limitation arises from electrical specifications for the line driver and receiver circuits, and how much current they can provide or sink. Another limitation is that the total cable length connecting all of the instruments on one bus cannot exceed 20 meters in length. Voltage levels on the various lines do not change instantaneously, but require a certain amount of time proportional to the length of the cable. A limit is placed on the cable length to insure that the bus will operate properly at its rated maximum speed. In general, then, the HP-IB is intended to provide a simple means of interconnecting local instrumentation clusters. Other means of interfacing (such as serial I/O) are better suited to long distance communications.

*A bus analyzer could be used to assert IFC.

HP-IB for the Specialist

Introduction

This section is intended for use by the specialist who is familiar with the IEEE-488-1978 interface standard and who requires detailed information about programming the HP-IB Interface. It is merely a description of the tools available for use by the expert; it is not a discussion of how to use them. More information is available in the IEEE Standard 488-1978 description.

Controlling with the HP-IB

HP-IB I/O Statements

ABORTIO: If the interface is System Controller, it pulses Interface Clear (IFC) and sets Remote Enable (REN) true. If the interface is not System Controller but is Controller Active, it sources its Talk Address which "untalks" any other Talkers on the bus. If the interface is neither System Controller nor Controller Active, it leaves the interface bus in the present state and becomes ready for the next I/O operation (see HALT).

ASSERT: ASSERT does an immediate write to Control Register 2, the HP-IB control lines, regardless of whether the interface is "busy" with I/O. Writing to Register 2 via CONTROL is identical except that CONTROL execution waits until the interface is no longer busy. The user must be aware of the interface logic to correctly write to Register 2 or the interface may become inoperative and have be reset.

CLEAR: Must be Controller Active; if bus addressing is specified, does the addressing and sends the Selected Device Clear command. If no addressing is specified, sends the universal Device Clear command. Upon completion of CLEAR, the ATN uni-line message remains true; it can be set false by doing a RESUME.

CONTROL: Writes to control registers within the interface. Error 111 is generated if writing occurs to a non-existent register. The valid Control Registers are 0-3 and 16-23.

ENABLE INTR: Same as Write Control to Register 1, provides the End-of-Line Interrupt Enable Mask.

ENTER: If addressing is specified, the interface must be Controller Active; if no addressing is specified, the interface begins inputting when it becomes Listener Active. The examples below show use of the ENTER statement with and without addressing.

```
ENTER 305 ; A$
```
Must be Controller Active, performs bus addressing prior to entering into the string variable A$.

```
ENTER 3 ; A$
```
If not Controller Active, the interface waits until it is addressed to Listen. If the interface is Controller Active, it must have already addressed itself to Listen or Error 116 is generated.

HALT:

Causes the interface to "break away" from its current I/O operation and become ready for the next I/O operation. HALT leaves the bus signals unaffected. Note that ABORTIO has the same affect as HALT if the interface is neither System Controller nor Controller Active.

LOCAL:

If no addressing is specified, then the interface must be System Controller and REN is set false. If addressing is specified, the interface must be Controller Active, then addressing is performed and the Go to Local (GTL) command is sent. After GTL is sent, the ATN uni-line message remains true; it can be set false by doing a RESUME.

LOCAL LOCKOUT:

Must be Controller Active, causes the interface to source the Local Lockout (LLO) command. After LLO is sent, the ATN uni-line message remains true; it can be set false by doing a RESUME.

OUTPUT:

If adressing is specified, the interface must be Controller Active; if no addressing is specified, the interface begins outputting when it becomes Talker Active. The examples below show use of the OUTPUT statement with and without addressing.

```
OUTPUT 305,306;A$
```
Must be Controller Active, performs bus addressing prior to outputting A$.

```
OUTPUT 3 ; A$
```
If not Controller Active, the interface waits until it is addressed to Talk. If the interface is Controller Active, it must have already addressed itself to Talk or Error 115 is generated.

PASS CONTROL:

Must be Controller Active. Addressing, if specified, precedes the Take Control command. The active controller can pass control to any device capable of controlling the bus (including itself).

PPOLL:

Must be Controller Active, returns a Parallel Poll response byte from the interface.

REMOTE:

Must be System Controller. Sets REN and sources addresses, if specified, to put bus devices in the remote state. If addresses are specified, ATN remains true; RESUME can be used to set ATN false.

REQUEST:

Must be non-Controller Active, sets SRQ on the interface if bit 6 (based on bits 0-7) is equal to one in the specified byte. This byte is provided to the Controller Active device upon being serial polled (which causes SRQ to be cleared). SRQ can also be cleared by providing a REQUEST byte with bit 6 = 0.

RESET:

Provides a hardware reset to the interface, returning it unconditionally to its power-on state. This causes the interface to perform the self-test. If self-test fails, Error 110 is displayed; if self-test passes, no display occurs.

RESUME: Must be Controller Active, sets ATN = 0. This is useful, for example, after doing a CLEAR, which leaves ATN = 1. Note that normally RESUME is not required since data I/O statements (OUTPUT, ENTER, TRANSFER, etc.) ensure that ATN is set to 0.

SEND: Used to send commands (in which case the interface must be Controller Active) or data (in which case the interface must be Talker Active). While sending data, ATN is false. While sending commands, ATN is true. Upon completion, ATN can be set false by using the RESUME instruction.

SPOLL: Must be Controller Active, used to conduct a serial poll of a device on the bus to obtain its status byte. The status byte indicates whether the device is requesting service and provides additional device-dependent information.

STATUS: Used to read status registers in the interface. STATUS is immediately executed regardless of the interface state (unless IFC is true). The valid Status Registers are 0-6; attempting to read registers outside of this range generates Error 111.

TRANSFER: Used for Interrupt or Fast Handshake I/O. TRANSFER relies on the use of declared buffers as discussed in the *I/O ROM Owner's Manual;* refer to the Specialized Transfer section of that document for information on buffer control; i.e. Fill and Empty pointers, etc. Associated with TRANSFER are three termination specifiers: (1) A delimiter numeric expression (DELIM), (2) a termination count (COUNT) and (3) an EOI enable specifier (EOI). Shown below are the termination specifiers which are permitted for each type of TRANSFER.

Table 4-1. Transfer Terminators

TRANSFER Type	DELIM	COUNT	EOI
TRANSFER(in)FHS	Not applicable	Used if supplied, otherwise Transfer proceeds from Fill Pointer to End of Buffer.	Enables EOI termination.
TRANSFER(out)FHS	Not applicable	Not applicable—transfer proceeds from Empty Pointer to Fill Pointer.	Not applicable—EOL sequence sent.
TRANSFER(in)INTR	Yes	Used if supplied, otherwise input terminates with DELIM, with EOI or when Fill Pointer is at End of Buffer.	Enables EOI termination.
TRANSFER(out)INTR	Not applicable	Not applicable—transfer proceeds from Empty Pointer to the Buffer Fill Pointer.	Not applicable—EOL sequence sent.

TRIGGER: Must be Controller Active, does addressing (if specified) followed by the Group Execute Trigger command. Upon completion, ATN remains true; it can be set false by the RESUME instruction.

*These statements require the use of an HP Series 80 I/O ROM.

Typical HP-IB Output Sequence

The table below illustrates the command bus sequence produced by the following statement (commands are in black, data bytes are in color):

```
10 OUTPUT 705;"HEWLETT-PACKARD INTERFACE BUS"
```

Command	Binary	Decimal	Mnemonics
U	01010101	85	ATN,MTA
?	00111111	63	UNL
%	00100101	37	LAG,ATN
H	01001000	72	(DATA)
E	01000101	69	(DATA)
W	01010111	87	(DATA)
•	•	•	•
•	•	•	•
•	•	•	•
•	•	•	•
•	•	•	•
•	•	•	•
B	01000010	66	(DATA)
U	01010101	85	(DATA)
S	01010011	83	(DATA)
	00001101	13	(CR)
	00001010	10	(LF)

Typical HP-IB Enter Sequence

The table below illustrates the command bus sequence produced by the following statement (commands are in black, data bytes are in color):

```
10 ENTER 705 ;A$
```

Command	Binary	Decimal	Mnemonics
?	00111111	63	ATN • UNL
5	00110101	53	MLA
E	01000101	69	TAG,ATN
H	01001000	72	(DATA)
E	01000101	69	(DATA)
W	01010111	87	(DATA)
•	•	•	•
•	•	•	•
•	•	•	•
B	01000010	66	(DATA)
U	01010101	85	(DATA)
S	01010011	83	(DATA)
	00001101	13	(CR)
	00001010	10	(LF)

Mnemonic Conventions

The following conventions are used to document the HP-IB control sequences that are used to implement bus functions.

<ATN>: Attention (ATN) true
<CA>: Controller Active State
<CR>: Carriage Return
<data>: One or more data bytes.
<DCL>: Device Clear
<GET>: Group Execute Trigger
<GTL>: Go To Local
<LA>: Listener Active State
<LAG>: Listen Address Group (listen addresses of specified devices)
<LF>: Line Feed
<LLO>: Local Lockout
<MLA>: My Listen Address (listen address of computer)
<MTA>: My Talk Address (talk address of computer)
<PPC>: Parallel Poll Configure
<PPU>: Parallel Poll Unconfigure
<SC>: System Controller
<SCG>: Secondary Command Group
<SDC>: Selected Device Clear
<SPD>: Serial Poll Disable
<SPE>: Serial Poll Enable
<TA>: Talker Active State
<TAD>: Talk Address (talk address of specified device)
<TCT>: Take Control
<UNT>: Untalk
<≅6μs>: Time span slightly greater than 6 microseconds

Message Concepts

Devices which communicate along the interface bus are transferring quantities of information. The transfer of information can be from one device to another device, or from one device to more than one device. These quantities of information can easily be thought of as "messages."

In turn, the messages can be classified into twelve types. The HP Series 80 Personal Computers are capable of implementing all twelve types of interface messages. The list below gives the twelve message types for the HP-IB.

1. The Data Message. This is the actual information which is sent from one talker to one or more listeners along the interface bus.
2. The Trigger Message. This message causes the listening device(s) to perform a device-dependent action when addressed.
3. The Clear Message. This message causes either the listening device(s) or all of the devices on the bus to return to their predefined device-dependent states.

4. The Remote Message. This message causes listening devices to switch from local front-panel control to remote program control when addressed to listen.
5. The Local Message. This message clears the Remote Message from the listening device(s) and returns the device(s) to local front-panel control.
6. The Local Lockout Message. This message prevents a device operator from manually returning the device to local (front-panel) control.
7. The Clear Lockout/Local Message. This message causes all devices on the bus to be removed from Local Lockout and revert to Local. This message also clears the Remote Message for all devices on the bus.
8. The Require Service Message. A device can send this message at any time to signify that the device needs some type of interaction with the controller. This message is cleared by sending the device's Status Byte Message if the device no longer requires service.
9. The Status Byte Message. A byte that represents the status of a single device on the bus. Bit 6 indicates whether the device sent a Require Service Message, and the remaining bits indicate operational conditions defined by the device. This byte is sent from a talking device in response to a serial poll operation performed by a controller.
10. The Status Bit Message. A byte that represents the operational conditions of a group of devices on the bus. Each device responds on a particular bit of the byte thus identifying a device-dependent condition. This bit is typically sent by devices in response to a parallel poll operation.

    The Status Bit Message can also be used by a controller to specify the particular bit and logic level that a device will respond with when a parallel poll operation is performed. Thus more than one device can respond on the same bit.
11. The Pass Control Message. This transfers the bus management responsibilities from the active controller to another controller.
12. The Abort Message. The system controller sends this message to unconditionally assume control of the bus from the active controller. This message terminates all bus communications (but does not implement a Clear Message).

    These messages represent the full implementation of all HP-IB system capabilities. Each device in a system may be designed to use only the messages that are applicable to its purpose in the system. It is important for you to be aware of the HP-IB functions implemented on each device in your HP-IB system to ensure the operational compatibility of the system.

HP-IB Control Lines

The figure below shows the meanings given to the eight control lines that make up the HP-IB. Three of these lines are designated as the **handshake** lines and are used to control the timing of data byte exchanges so that the talker does not get ahead of the listener(s). The three handshake lines are:

Data Valid (DAV)
Not Ready for Data (NRFD)
Not Data Accepted (NDAC)

Using these lines, a typical data exchange would proceed as follows. All devices currently designated as active listeners would indicate (via the NRFD line) when they are ready for data. A device not ready

would pull this line low (ground), while a device that is ready would let the line float high. Since a low overrides a passive high, this line will stay low until all active listeners are ready for data. When the talker senses this, it places the next data byte on the data lines and then pulls DAV low. This tells the listeners that the information on the data lines is valid and that they may read it. Each listener (at its own speed) then takes the data and lets the NDAC line go high. Again, only when all listeners have let NDAC go high will the talker sense that all listeners have read the data. It can then remove DAV (let it go high) and start the entire sequence over again for the next byte of data.

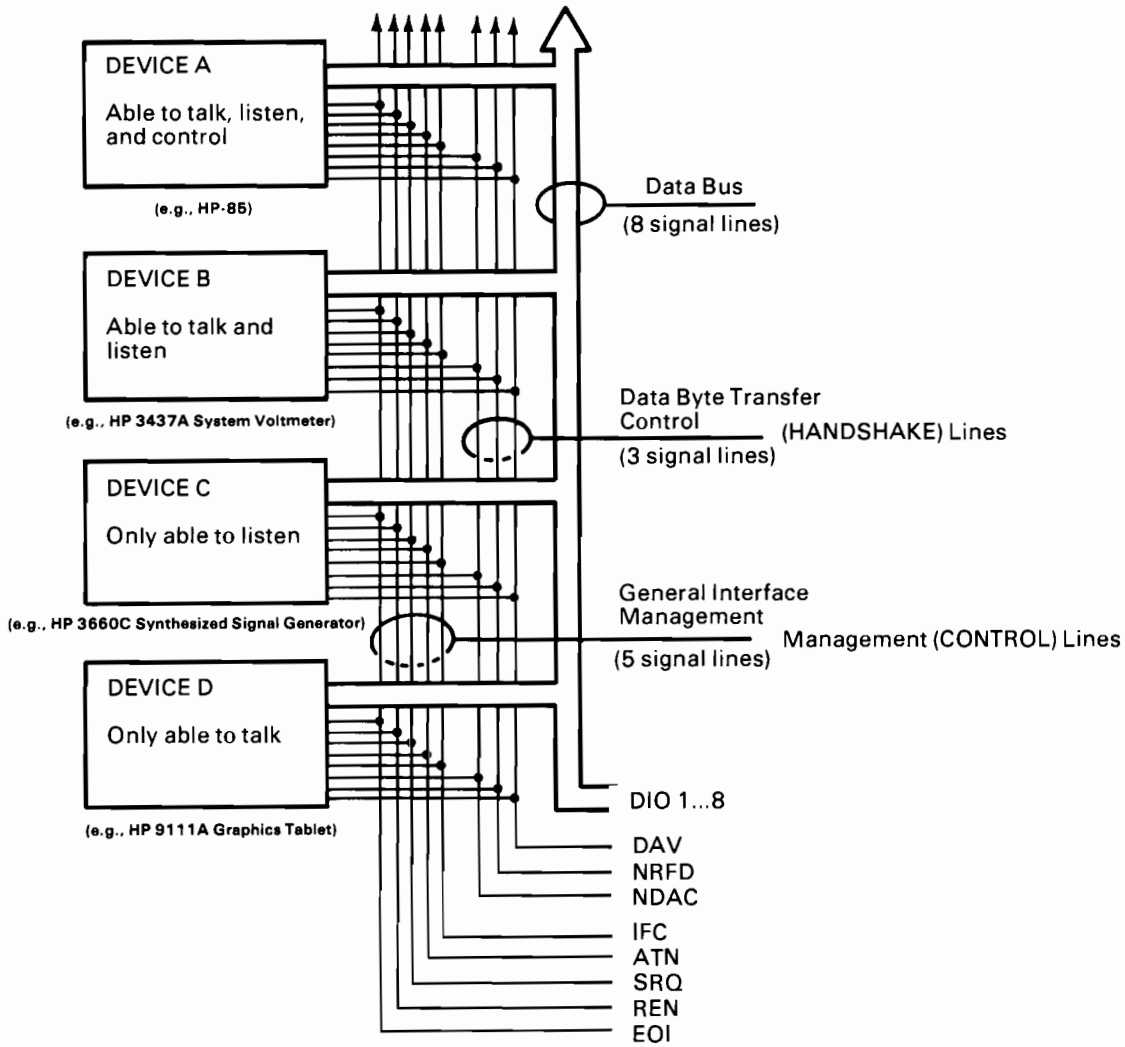


Figure 4-1. HP-IB Signal Lines

Attention (ATN)

Command messages are encoded on the data lines as 7-bit ASCII characters, and are distinguished from normal data characters by the setting of the attention (ATN) line. That is, when the ATN line is false, bytes on the data lines are interpreted as simple data characters. But when the ATN line is true, the data lines become the carriers of command information. The set of 128 ASCII characters that can be placed on the data lines during this ATN-true mode are divided into four classes as shown in an appendix at the back of the *I/O ROM Owner's Manual.*

Interface Clear (IFC)

Only the hardwired system controller can set the IFC line true. By asserting IFC, all bus activity is unconditionally terminated, the system controller regains (if it has been passed to another device) the status of active controller, and any current talker and listeners become unaddressed. Normally, this line is only used to abort an unwanted operation, or to allow the system controller to regain control of a bus where something has gone wrong. It overrides any other activity that is currently taking place on the bus.

Remote Enable (REN)

This line is used to allow instruments on the bus to be programmed remotely by another device on the bus, usually (but not necessarily) the active controller. Any device that is addressed to listen while REN is true is placed in the REMOTE mode of operation.

End or Identify (EOI)

Normally, data messages sent over the HP-IB are sent using the standard ASCII code and are terminated by the ASCII line-feed character (decimal 10). A device (e.g., a disc) may wish to send blocks of information in 8-bit bytes which represent general binary patterns; and no specific 8-bit pattern can be designated as a terminating character since it could occur anywhere in the data stream. In this case, the EOI line is used to mark the end of the data message. When the listeners detect that the EOI line is true, they recognize that the byte on the data line is the last one of the data message.

The EOI line is also used during an identify (parallel poll) sequence.

Service Request (SRQ)

The active controller is always in charge of the order of events on the HP-IB. If a device on the bus has some information of which the controller should be aware, it can use the service request line to ask for the controller's attention. For example, a printer might request service to inform the controller that it is out of paper. A digitizer might assert a service request to tell the controller that its sample button was pressed by the operator and a reading is ready to be taken. This represents a request (NOT a demand), and it is up to the controller when and how it will service that device. However, the device will continue to assert SRQ until it has been satisfied. Exactly what will satisfy a service request depends on each individual device and will be contained in the operating manual for that device.

HP-IB Control Responses

The following table shows the responses of the HP-IB Interface when it receives the various bus control messages.

Table 4-2. HP-IB Control Responses

ATN:	Interprets bus data as commands. The interface can still interact with the computer while receiving commands.
IFC:	Clears all talker and listener states to the power-on state. May release Active Control unless also System Controller.
REN:	Sets bit 1 of SR5.
EOI:	Terminates data input transfer to a buffer. Can terminate ENTER or TRANSFER statement.
SRQ:	Sets the service request bit (bit 3 of SR1) and interrupt if bit 3 of interrupt mask is set.
DCL, SDC:	Can interrupt the computer if bit 2 of CR1 is set.
GTL, LLO:	Set appropriate bits in SR5.
GET:	Can interrupt the computer if bit 1 of CR1 is set.
Serial poll:	Delivers the currently set serial poll response byte (REQUEST) without computer intervention.
Parallel poll:	Responds to a parallel poll using the line and sense set by the switches on the card if asserting SRQ.
PPU, PPC:	Parallel poll response is switch settable and not programmable. No response.
TCT:	Assumes active control of the HP-IB.

Polling

Polling is a special bus activity that permits a controller to determine the operating status of other devices. These two processes, serial poll and parallel poll, are an intricate part of the interupt protocol. However, because software can define whether a serial or parallel poll (or both) is implemented, they are discussed as separate topics to simplify the explanation.

Serial Poll

A serial poll permits the controller active device to obtain a status byte from any bus device that supports the serial poll function. When the controller active device detects that a service request exists (SRQ line is pulled), it may serially poll the bus devices expected to have requested service. When a device is polled, it always returns its status byte. Bit 7 of the status byte will be set to 1 if the device requested service or it will be 0 if it did not. The remainder of the bits in the status byte can be used to indicate the reason for the service request and are totally device dependent.

The programmer has two means of detecting when SRQ is pulled:

- Perform a STATUS read of register SR2 to see if the SRQ bit (bit 5) is set;
- Enable an End-of-Line interrupt when SRQ is pulled.

The bus might have only one bus device capable of requesting service and perhaps for only one purpose. In this case, a serial poll may not be needed. However, if a single device can pull SRQ for various reasons or, if more than one bus device is capable of pulling SRQ, then a serial poll can be used to determine which device(s) requested service and why.

When the HP Series 80 Personal Computer is controller active and initiates a serial poll, the program specifies the select code and the address of the device to be polled. The HP-IB interface then sends the following sequence over the bus:

1. ATN is set true.
2. UNL (Unlisten) command is sent to the device.
3. Interface sends its own listen address and the peripheral talk address commands.
4. SPE (Serial Poll Enable) is sent over the bus, followed by ATN going false. The device being polled then sends its status byte. If it was the only device requesting service, SRQ is removed. If it was not the only device pulling SRQ, or if it didn't request service, SRQ remains, indicating some other device requires service.
5. ATN is set true again followed by the SPD (Serial Poll Disable) command.
6. The UNT (Untalk) command is then sent over the bus causing the peripheral to cease being a talker.

Note in the above sequence that SRQ may or may not be removed when a device is polled. SRQ is removed if the device being polled is the one and only device requesting service. Therefore, the user's program must specify the order in which bus devices are polled and serviced.

If the HP Series 80 Personal Computer is not the controller active device, the user may allow it to request service with the REQUEST statement followed by a program dependent status byte. Bit 7 of this status byte maps directly to SRQ on the bus. The computer provides this status byte when it is serially polled by the controller active device. If it was requesting service, bit 7 will be set to 1; otherwise, it will be 0. The computer will remove SRQ when it sends its status byte if it was the one and only bus device requesting service. However, if SRQ is removed, bit 7 remains set until changed by the program.

Parallel Poll

In a parallel poll each bus device is assigned one of the DIO (data) lines for identification purposes. This is configured within each bus device. The parallel poll jumper on the interface is used to assign one of the DIO lines for identification of the interface if it is to respond to parallel polls.

A parallel poll can be much faster than a serial poll because the controller reads DIO1 through DIO8 all at once to determine which device(s) requested service.

Section 5

Control Registers

The primary purpose of the interface is to enable data to be exchanged between the computer and the peripheral device to which it is connected. Interface modules are extremely versatile, however, and most of them are programmable. This means that they have various optional capabilities that can be set and changed by control instructions from the computer. The HP-IB interface has 12 "write-only" control registers. These registers reside in the interface module and are accessible via SET I/O (in the Plotter/Printer ROM) and CONTROL (in the I/O ROM) statements.

You have the capability to write to Control Registers 0 through 3 and 16 through 23. The registers are not numbered consecutively to ensure compatibility with the serial and GPIO interfaces which both have their EOL (end-of-line) sequence specifiers in Registers 16 through 23. Writing to control registers outside this range generates ERROR 111.

CAUTION

Do not write to Control Registers 0 through 3 unless you have an I/O ROM and you are completely familiar with the function of these registers. In particular, Control Registers 2 and 3 provide direct access to the HP-IB control and data lines. They must be used with care, and used only by persons aware of HP-IB protocols! It is possible to cause a bus malfunction or device damage by improper use of these registers.

HP-IB Control Registers

These registers are set by executing the CONTROL statement. For example, to set the value of control register 0 to a 1, execute CONTROL 7,0;1. Execution of the CONTROL statement is delayed until any operation currently in progress has been completed. The ASSERT statement, however, provides immediate access to CR2, the HP-IB Control Lines Register, regardless of any operations in progress.

These registers may also be set or altered by the SET I/O command from the Printer/Plotter ROM (SET I/O is built in to the HP-87). Since it is possible to cause bus malfunctions (and even device failure) by improper use of control registers, it is recommended that these statements be used with care.

A complete control register table is given, followed by explanations of the individual registers.

Table 5-1. HP-IB Control Registers

Register Number	Bit Number								Default Value	Register Function
	7	6	5	4	3	2	1	0		
CR0	X	X	X	X	Odd	Even	Always One	Always Zero	0	Parity Control
CR1	IFC	LA	CA	TA	SRQ	DCL or SDC	GET	SCG	0	Interrupt Mask
CR2	X	REN	SRQ	ATN	EOI	DAV	NDAC	NRFD	Not Applicable	HP-IB Control Lines
CR3	DI08	DI07	DI06	DI05	DI04	DI03	DI02	DI01	Not Applicable	HP-IB Data Lines

CR0: Parity Control

This register controls the parity mode for input and output **data**: no parity is used for commands. The right-most non-zero bit controls the parity selection, with an all-zero value (CR0 = 0) meaning no parity (default at power-on). For example, `CONTROL 7,0;4` selects **even** parity. `CONTROL 7,0;6` selects **always one** parity (even though even parity has been indicated by bit 2 being set, right-most bit set is bit 1 = always one).

CR1: Interrupt Mask

A bit, when **set**, enables the corresponding interrupt condition to cause an end-of-line branch.

- Bit 0, when set, enables interrupt when a secondary command that was received is stored in SR6, the secondary command register. This is an event-initiated interrupt.
- Bit 1, when set, enables interrupt when a GET (Group Execute Trigger) is received while addressed to listen (LA). This is an event-initiated interrupt.
- Bit 2, when set, enables interrupt either when DCL (Device Clear) is received, or when SDC (Selected Device Clear) is received while addressed to listen (LA). This is an event-initated interrupt.
- Bit 3, when set, enables interrupt when SRQ (Service Request) is true.
- Bit 4, when set, enables interrupt when TA (talker active: addressed to talk) becomes true. If TA is **already** true, then a 0-to-1 transition of bit 4 of CR1 causes an interrupt. This is a state-initiated interrupt.
- Bit 5, when set, enables interrupt when CA (controller active) becomes true (by receiving control). If CA is **already** true, then a 0-to-1 transition of bit 5 of CR1 causes an interrupt. This is a state-initiated interrupt.
- Bit 6, when set, enables interrupt when LA (listener active: addressed-to-listen) becomes true. If LA is **already** true, then a 0-to-1 transition of bit 6 of CR1 causes an interrupt. This is a state-initiated interrupt.
- Bit 7, when set, enables interrupt when an IFC (Interface Clear) occurs. An **externally** caused IFC can cause an interrupt even when the interface is the system controller. This is an event-initiated interrupt.

The following diagrams illustrate the interface response to state-initiated, event-initiated, and SRQ interrupts. The interrupt-cause status registers (SR1) bits are set when an interrupt occurs and cleared when the status of SR1 is read.

Event-Initiated Interrupt
(SCG,GET,SDC/DCL,IFC)

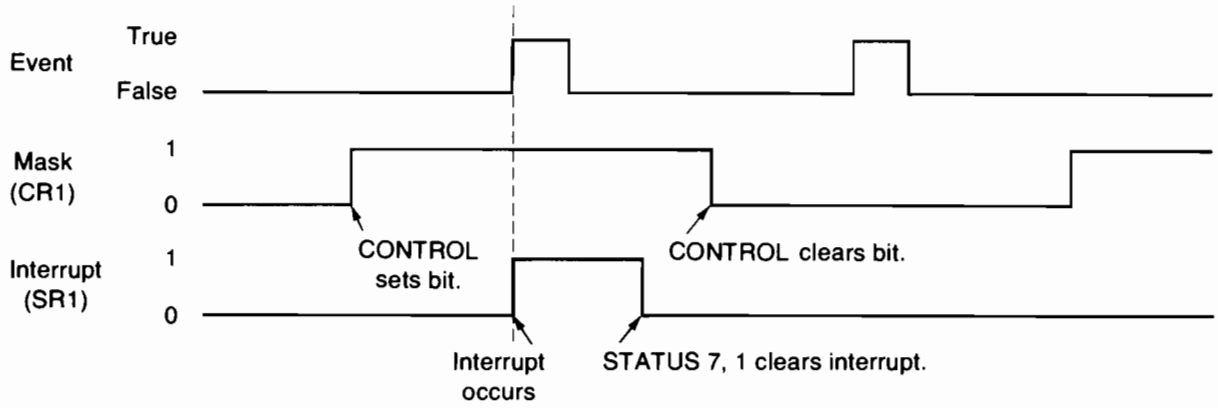


Figure 5-1. Event-Initiated Interrupt Timing

State-Initiated Interrupt
(TA,CA,LA)

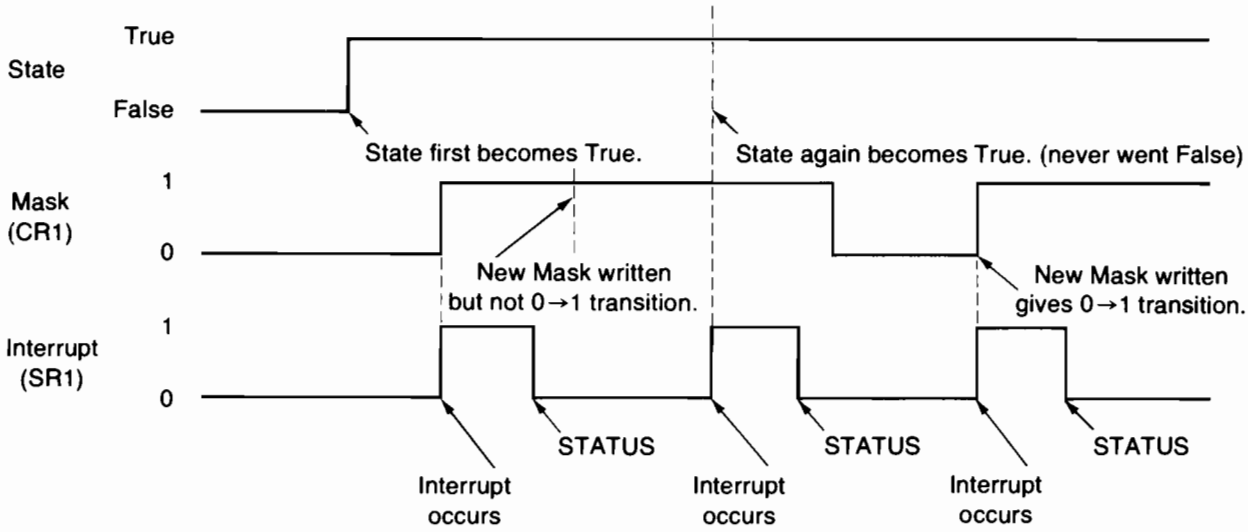


Figure 5-2. State-Initiated Interrupt Timing

SRQ interrupt

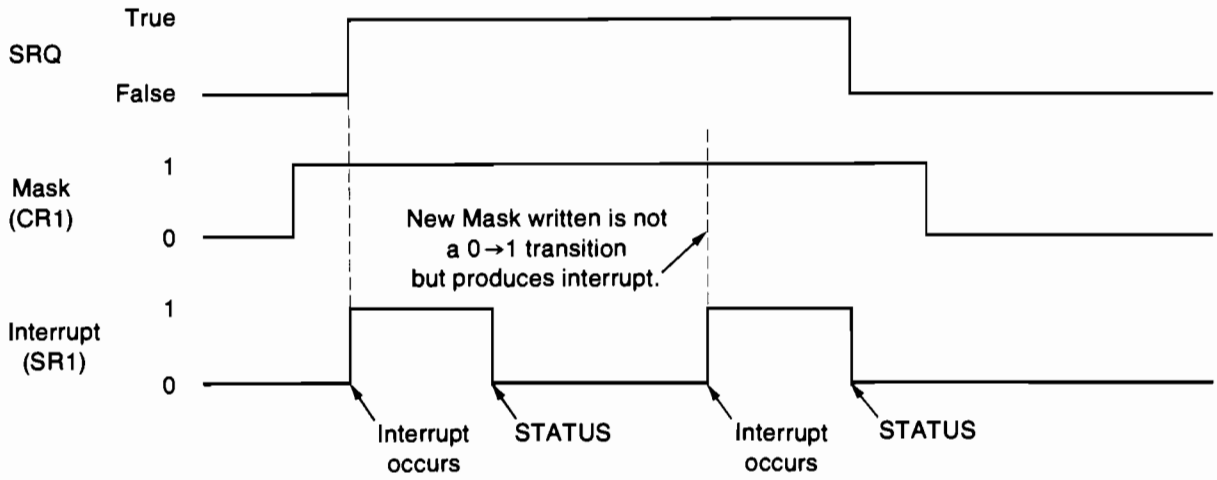


Figure 5-3. SRQ Interrupt Timing

CR2: HP-IB Control Lines

This register gives direct access to the eight HP-IB control lines. A bit, when set, causes the corresponding HP-IB control line to be set true for as long as the CR2 bit is set. The user is cautioned to be aware of bus protocols when using this register. For example, a non-controller may not set the ATN line true, and a non-system controller is not supposed to assert REN line.

CR3: HP-IB Data Lines

This register gives direct access to the eight HP-IB data lines, DI01 through DI08. Setting a bit in this register causes the corresponding HP-IB data line to be set true. (The user should note; however, that HP-IB data lines are numbered 1 through 8 while the control register data lines are numbered 0 through 7.) The user should exercise caution when writing to CR3, the HP-IB data lines register. For example, by writing to CR3 while the interface is addressed to listen (LA state), a hardware conflict will occur in the interface that could cause erratic operation and damage the interface.

User-defined End-of-Line Sequence Registers

Control registers CR16 through CR23 provide the user with the capability of customizing the end-of-line output sequence that is sent at the end of a data transfer and with the "/" image specifier. Also provided is the capability of automatically asserting EOI (End or Identify) with the last character of a data transfer.

The following table shows these registers, and their meanings are explained in the following paragraphs.

Table 5-2. HP-IB End-of-Line Sequence Registers

Control Register Number	Bit Number								Default Value	Register Function
	7	6	5	4	3	2	1	0		
DR16	EOI Enable	X	X	X	X	EOL2	EOL1	EOL0	2	EOL Control
CR17	Default value = 13 (Carriage Return)								13	Character 1
CR18	Default value = 10 (Line Feed)								10	Character 2
CR19									0	Character 3
CR20									0	Character 4
CR21									0	Character 5
CR22									0	Character 6
CR23									0	Character 7

CR16: EOL Control

This register controls the end-of-line character sequence that is normally sent after a line of output and for the "/" image specifier.

- Bits 0 through 2 specify the number of characters sent as the EOL (end-of-line) sequence. The default count is 2, which causes 2 characters (CR and LF) of registers CR17-CR23 to be sent. A count of 0 specifies that no EOL sequence is to be sent.
- Bits 3 through 6 are not used. (X = don't care)
- Bit 7 when set causes the HP-IB control line EOI to be asserted with the last byte of a data transfer. If the EOL count is non-zero, EOI is asserted with the last character of the EOL sequence. If the EOL count is zero, EOI is asserted with the last character of the data list being sent (PRINT, SEND, and TRANSFER only).

CR17-CR23: EOL Sequence

These registers contain the characters sent as the end-of-line sequence. Default values are a carriage return (decimal 13) for CR17 and a line feed (decimal 10) for CR18.

To set up an EOL sequence for double-spaced printing for example, set the EOL count to 3 and the EOL sequence to DR, LF, LF.

```
CONTROL 7,16;3,13,10,10
```

SET I/O for Register Control

When using the Plotter/Printer ROM with an HP-IB interface (without an I/O ROM) the SET I/O statement can be used to define an end-of-line sequence. There is no method for reading the status registers without an I/O ROM and the use of SET I/O requires a full understanding of the workings of the HP-IB interface and the specifics of the devices that you are using. The following sequence of instructions performs the same operation as the previous single CONTROL statement.

```
210 SET I/O 7, 16, 3
220 SET I/O 7, 17, 13
230 SET I/O 7, 18, 10
240 SET I/O 7, 19, 10
```

HP-IB Status Registers

These registers are ready by executing the STATUS statement. For example, to return the value of Status Register 3 in variable S3, execute STATUS 7,3;S3. A complete status register table is given, followed by explanations of the individual registers.

Table 5-3. HP-IB Status Registers

Status Register Number	Bit Number								Default Value	Register Function
	7	6	5	4	3	2	1	0		
SR0	0	0	0	0	0	0	0	1	1	Interface Identification
SR1	IFC	LA	CA	TA	SRQ	DCL or SDC	GET	SCG	0	Interrupt Cause
SR2	0	REN	SRQ	ATN	EOI	DAV	NDAC	NRFD	64	HP-IB Control Lines
SR3	DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1	Not Applicable	HP-IB Data Lines
SR4	0	0	SC	A4	A3	A2	A1	A0	53	HP-IB Address/ System Controller
SR5	SC	LA	CA	TA	SPE	Parity Error	REN	LLO	160	State Register
SR6	0	0	0	SC5	SC4	SC3	SC2	SC1	0	Secondary Commands

SR0: Interface Identification

Always returns a value of 1, meaning an HP-IB interface

SR1: Interrupt Cause (See also HP-IB Control Registers discussion for interrupt timing diagrams)

A bit, when set, indicates the interrupt condition that caused an end-of-line branch. SR1 is reset to 0 when it is read by a STATUS statement.

- Bit 0, when set, indicates that an SCG (secondary command) was received. The value of the secondary command received is available in SR6. This is an event type interrupt: the interrupt, if enabled, will occur when a secondary command is received.
- Bit 1, when set, indicates that a GET (group execute trigger) was received while addressed to listen (LA). This is an event type interrupt: if enabled, the interrupt will occur when trigger is received.
- Bit 2, when set, indicates that either (1) a DCL (device clear) was received or that (2) an SDC (selected device clear) was received while addressed to listen (LA). This is an event type interrupt: if enabled, the interrupt will occur when device clear is received.

- Bit 3, when set, indicates that an SRQ (service request) was received. The interrupt will occur as long as the SRQ line is true. In general, this means that your service routine must clear SRQ, which is usually accomplished by satisfying the requesting device's need for service.
- Bit 4, when set, indicates that either (1) the talker active (TA) bit of CR1 underwent a 0-to-1 transition while the interface was addressed to talk, or (2) the interface became addressed to talk while the talker active (TA) bit of CR1 was set. This is a state-enable interrupt.
- Bit 5, when set, indicates that either (1) the controller active (CA) bit of CR1 underwent a 0-to-1 transition while the interface was active controller, or (2) the interface received control while the controller active (CA) bit of CR1 was set. This is a state-enable interrupt.
- Bit 6, when set, indicates that either (1) the listener active (LA) bit of CR1 underwent a 0-to-1 transition while the interface was addressed to listen, or (2) the interface became addressed to listen while the listener active (LA) bit of CR1 was set. This is a state-enable interrupt.
- Bit 7, when set, indicates that an IFC (interface clear) has occurred on the bus. This is an event type interrupt: the interrupt, if enabled, will occur when interface clear is received.

SR2: HP-IB Control Lines

A bit, when set, indicates that the corresponding HP-IB control line is true.

SR3: HP-IB Data Lines

A bit, when set, indicates that the corresponding HP-IB data line is true.

SR4: HP-IB Address/System Controller Switches

- Bits 0 through 4 indicate the current setting of the HP-IB address switches of the interface. These are factory set to 21.
- Bit 5 indicates the setting of the system controller switch of the interface. A 1 indicates system controller.
- Bits 6 and 7 are always 0.

SR5: HP-IB State Register

This register indicates current HP-IB status of the interface.

- Bit 0, when set, indicates that the interface is in a local lockout state (LLO).
- Bit 1, when set, indicates that the interface is in a remote state (REN).
- Bit 2, when set, indicates that a parity error occurred on input while parity was enabled. It is cleared when R5 is read.
- Bit 3, when set, indicates that SPE (serial poll enable) has been received. It is cleared when SPD (serial poll disable) is received.
- Bit 4, when set, indicates that the interface is addressed to talk (TA or talker active).
- Bit 5, when set, indicates that the interface is active controller (CA or controller active).
- Bit 6, when set, indicates that the interface is addressed to listen (LA or listener active).
- Bit 7 when set indicates that the interface is system controller (same as SR4 bit 5).

SR6: Secondary Command Register

- Bits 0 through 4 indicate the last secondary command received when the secondary command bit (Bit 0) of CR1 is set. The SR6 register contains any secondary command that follows the interface's talk or listen address.
- Bits 5 through 7 are always 0.

Section 6

Maintenance, Service, and Warranty

Maintenance

There are no customer serviceable parts inside the external HP 82937A HP-IB Interface. It should not be necessary to clean the interface module or cable contacts. The action of installing the module in the port or plugging the cable into a peripheral is normally sufficient to clean contamination from the contacts. Again, for a built-in interface there are no customer serviceable parts.

Service

If at any time you suspect that the interface may be malfunctioning, do the following:

1. Turn off the computer and all peripherals. After disconnecting all plug-in devices from the ports, turn on the computer. If the cursor appears and no error message is displayed, the computer is functioning properly.

2. Turn off the computer. After installing the interface module in question in any port, turn on the computer.

   - If `ERROR 110 : I/O CARD` appears, the interface module requires service.

   - If the cursor does not appear, the system is not operating properly. To help determine if the interface module is interfering with proper operation, repeat this step with the module installed in a different port.

3. If you should get a device error upon addressing a peripheral device, for example, `ERROR 126 : PLOTTER`, check to make sure that the power switch of the device is in the ON position and that you have specified the correct address. If you still get a device error, try addressing the device with another HP-IB interface module, if available. If the system does not respond with another device error, the original interface module may require service.

4. If improper operation is indicated in either the interface module or the computer, repair service is required.

Warranty and Repair Service Information

The warranty statement and procedures for obtaining repair service are contained on the Warranty and Service Information sheet shipped with your HP 82937A HP-IB Interface. If you need additional information, please contact your authorized HP dealer or the nearest Hewlett-Packard sales and service facility.

If you have questions concerning the warranty, please contact:

In the U.S.: One of the six Field Repair Centers listed on the Service Information Sheet packaged with your owner's documentation.

In Europe:
Hewlett-Packard S.A.
7, rue du Bois-du-lan
P.O. Box
CH-1217 Meyrin 2
Geneva
Switzerland
Tel. (22) 82 70 00

Other Countries:
Hewlett-Packard International
3495 Deer Creek Rd.
Palo Alto, California 94304
U.S.A.
Tel. (415) 857-1501

Potential for Radio Frequency Interference

The HP-IB Interface generates and uses radio frequency energy and may cause interference to radio and television reception. Your interface complies with the specifications in Subpart J of Part 15 of the Federal Communications Commission rules for a Class B computer device. These specifications provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. If the computer does cause interference to radio or television reception, which can be determined by turning the computer off and on, you can try to eliminate the interference problem by doing one or more of the following:

- Reorient the receiving antenna.
- Change the position of the computer with respect to the receiver.
- Move the computer away from the receiver.
- Change the position of peripherals and interface cables with respect to the receiver.

  Plug the computer into a different outlet so that the computer and the receiver are on different branch circuits.

If necessary, consult an authorized HP dealer or an experienced radio/television technician for additional suggestions. You may find the following booklet, prepared by the Federal Communications Commission, helpful: *How to Identify and Resolve Radio-TV Interference Problems.* This booklet is available from the U.S. Government Printing Office, Washington, D.C. 20402, Stock No. 004-000-00345-4.

If your system malfunctions and repair is required, you can help assure efficient servicing by having the following items with your unit(s) at the time of service:

1. A description of the configuration of the computer, exactly as it was at the time of malfunction, including any plug-in modules, tape cartridges or other accessories.
2. A brief description of the malfunction symptoms for service personnel.
3. Printouts or any other materials that illustrate the problem area.
4. A copy of the sales slip or other proof of purchase to establish the warranty coverage period.

Computer and peripheral design and circuitry are proprietary to Hewlett-Packard and service manuals are not available to customers.

Each computer and peripheral carries an individual serial number. It is recommended that you keep a separate record of this number. Should your unit be stolen or lost, the serial number is often necessary for tracing and recovery, as well as any insurance claims. Hewlett-Packard does not maintain records of individual owner's names and unit serial numbers.

General Shipping Instructions

Should you ever need to ship any portion of your computer system, be sure it is packed in a protective package (use the original case), to avoid in-transit damage. Hewlett-Packard suggests that the customer always insure shipments.

If you happen to be outside of the country where you bought your computer or peripheral, contact the nearest authorized HP-83/85/87 dealer or the local Hewlett-Packard office. All customs and duties are your responsibility.

Notes

Appendix A

Functional Description

Introduction

The HP-IB Interface (see figure A-1) is based on six primary integrated circuits (ICs):

- The microcomputer IC.
- The translator IC.
- The two transceiver ICs.
- The switch buffer IC.
- The control latch IC.

All of these ICs are contained on the interface PCA (printed-circuit assembly). (See figure A-3 for the schematic diagram of the interface.)

The external, plug-in HP-IB interface will be discussed in the following pages. The internal, built-in interface is identical in function, operation and even hardware but is not user accessible. The main difference between these devices is the component numbering system. The reference designations used in this section refer to the external interface.

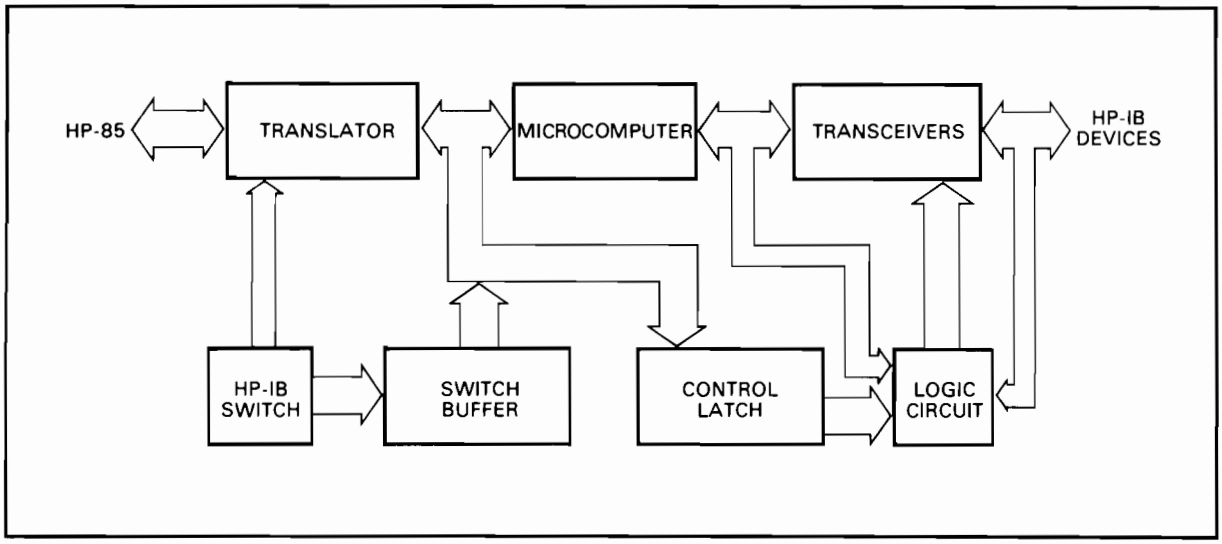


Figure A-1. HP-IB Interface Block Diagram

Hardware Description

Translator

The translator IC (U1) performs the following functions: transferring data and interrupts between the HP Series 80 Personal Computer (CPU) and the microcomputer, (μC), transferring command information from the CPU to the μC, synchronizing the transfer of data in "Fast Handshake" mode, and sensing the select code switch settings (which identify the HP-IB).

Another function performed by the translator is interfacing the different logic levels of the CPU and the microcomputer. Timing is provided by two of the computer timing signals from the I/O port.

Four 8-bit registers in the translator provide information to the computer and the microcomputer. The output buffer and input buffer hold data being sent to and from the microcomputer, respectively. The control register defines data (in the output buffer) being sent to the microcomputer. The status register controls the communication of data to the CPU.

Switch Buffer

The switch buffer is a single integrated circuit that isolates switch lines A0 through A4 and SC from the interface address data bus. These lines were discussed in section 2.

The outputs of U4 connect to the address-data bus and the external control latch (U3). Unless U4 is enabled, these outputs remain in a high impedance state. When it is enabled via inputs DIS1 and DIS2, the switch settings are reflected on the outputs and therefore appear on D0 through D5 of the address-data bus and the inputs to the external control latch. Thus, the switch buffer is enabled only during initialization or when the 8049 μC is reset to prevent it from interfering with other operations. Once the status of the switches is read by the μC it is not necessary to do so again unless the μC is reset.

To read the switch buffer, the μC sends an address to the translator read/write logic via the address-data bus and latches it with the address latch enable (ALE) signal. The translator responds by placing ADR3 low. The μC then places RD low. During the period when both these signals are low, the switch buffer is enabled allowing the switch settings to be reflected on D0 through D5 of the address-data bus to be read by the μC.

External Control Latch

This latch is clocked to accept the information on the address-data bus lines (D0 — D5). This occurs when WR is low and the μC writes an address to the translator that causes ADR3 to go low. This latch is cleared on initialization or when the μC is reset.

The output lines of this latch serve a variety of functions and are discussed separately.

Output	Function
EN NDAC	The IEEE Standard 488-1978 specifies that when ATN goes true, NDAC must go false within 200 ns. This helps ensure that a transfer will proceed in an orderly fashion. The EN NDAC output is gated with IATN to provide a hardware implementation to meet this specification.

EN ATN INT	This line is gated with the IATN line to the interrupt (INT) input of the μC. When EN ATN INT is set and a peripheral device asserts ATN, the μC recognizes an interrupt request.
EN REN INT	This line is gated with the IREN line to the interrupt (INT) input of the μC. When EN REN INT is true and a peripheral device asserts REN, the μC recognizes an interrupt request.
EN IFC INT	This is gated with the IIFC line to the interrupt (INT) input of the μC. When EN IFC INT is set and a peripheral device asserts IFC, the μC recognizes an interrupt request.
TALKER ACTIVE (TA)	When TA and IATN are both true it enables the bi-directional buffers connected to the HP-IB bus (DIO1 through DIO8) in a direction which permits the μC to output information to the bus.
CONTROLLER ACTIVE (CA)*	When this line and IATN are both set the μC is allowed to output information to the HP-IB bus in the same manner described above. Also, the controller can only source commands when ATN is true. The CA line is also used to enabled the buffers for the service request (SRQ) and attention (ATN) lines on the HP-IB bus. In the latter, the controller can only source commands when ATN is true; while another talker can only source data when ATN is false.

Bi-directional Bus Transceivers

Two integrated circuits provide the bi-directional bus transceiver circuitry used to transmit or receive information between the computer and other HP-IB bus devices. The send/receive pairs that connect to bus lines DIO1 through DIO8 are enabled so that all either send or receive bus information in the same direction simultaneously. The handshake and control lines for the bus are enabled as follows:

NRFD and NDAC	Two send/receive pairs are enabled to send or receive the state of these bus lines in the same direction simultaneously.
DAV and EOI	Two send/receive pairs are enabled to send or receive the state of these bus lines in the same direction simultaneously.
ATN	One send/receive pair is enabled by the CA line to permit ATN to be asserted from the controller to other bus devices.
SRQ	One send/receive pair is enabled by the CA line to permit SRQ to be asserted over the HP-IB bus. If CA is high, the CPU monitors SRQ from the bus. If CA is low, the CPU pulls SRQ when it requests service from another controller.
REN and IFC	Two send/receive pairs are enabled to send or receive the state of these bus lines in the same direction simultaneously.

* CONTROLLER ACTIVE (CA) is synonymous with the term controller-in-charge which is referenced in IEEE Standard 488-1978.

8049 Microcomputer

Most of the activities carried out by the 8049 μC have already been discussed in this section. This discussion will summarize these activities.

The μC is the intermediary between the HP Series 80 Personal Computer and the HP-IB bus. It implements interface protocol via its own self-contained ROM. The μC responds to instructions from the host CPU or the bus, depending upon which bus device is assigned controller status. When the host is system controller, the μC can recognize an interrupt from an HP-IB bus device, and, depending upon user-programmed interrupt conditions, it may service the interrupt or it may in turn interrupt the computer.

There are two 8-bit I/O ports that connect the μC to the HP-IB bus. Port 1 connects to the handshake and control lines; port 2 connects to the data lines (DIO0 through DIO8). There are eight other bits (D0 through D7) which go to the address-data bus connecting the μC to the Translator.

Internal to the μC are seven 8-bit status registers and twelve 8-bit control registers. The status registers indicate the state of the μC at a given time. The control registers are written to when the state of the interface is to be changed. Register tables of the status and control registers are given in section 6. If a detailed description of these registers is desired, refer to that section's discussion.

CAUTION

Do not write to Control Registers 0 through 3 unless you have an I/O ROM and you are completely familiar with the function of these registers. In particular, Control Registers 2 and 3 provide direct access to the HP-IB control and data lines. They must be used with care, and used only by persons aware of HP-IB protocols! It is possible to cause a bus malfunction or device damage by improper use of these registers.

HP-IB Bus Lines

The standard HP-IB signal lines are described next. The function of each line is defined by IEEE Standard 488-1978.

Table A-1. HP-IB Signal Lines

DIO1	Data Input/Output 1
•	•
•	•
•	•
DIO8	Data Input/Output 8
DAV	Data Valid
NRFD	Not Ready for Data
NDAC	Not Data Accepted
IFC	Interface Clear
ATN	Attention
SRQ	Service Request
REN	Remote Enable
EOI	End or Identify

Note: IEEE Standard 488-1978 specifies that the HP-IB bus use negative true logic which is often confusing or misunderstood. This explanation of negative true logic, as it applies to the HP-IB bus and interface, is given so that no misconceptions exist:

All HP-IB bus lines are shown on the schematic without bars over them. Thus, if a bus line is true, it will have a low voltage level. Positive true logic is used on the interface. In positive true logic a line whose mnemonic does not have a bar over it is true when it has a high voltage level. As an example, on the schematic bus line ATN is shown as IATN when it appears as an internal signal on the interface. Thus, the voltage level representing a true ATN on the HP-IB bus is the same as a true IATN on the interface.

Data Lines (DIO1 – DIO8)

The data lines are used to communicate all data including input, output, program codes, status, and control information between instruments connected to the bus. One character byte is sent at a time in a byte-serial, bit-parallel fashion. In most instruments, characters are based on the 8-bit ASCII representation. For instance, if the interface is connected to a printer, the 8 bits that represent one character are all sent at once in parallel. Then, the next character is sent and so on. Thus, the 8 bits (one byte) defining a character are all sent at once in parallel (bit-parallel) while each character is sent serially (byte-serial).

SRQ, (Service Request) is pulled low by a device when it wants the attention of the controller. SRQ may be driven low at any time except when IFC is low.

EOI (End or Identify) may be used to indicate the end of a string of characters. For example, if ATN is high to indicate the information on DIO1 through DIO8 is data, the talker may indicate the end of its data transmission by driving EOI low when it places the last byte on the data lines.

REN (Remote Enable) is driven by the system controller and is one of the lines necessary for an instrument to operate under remote control. Only instruments capable of remote operation use REN and they monitor it at all times. Instruments that do not use REN terminate the line into a resistor load. The system controller may change the state of REN at any time.

Transfer Lines (NRFD, NDAC and DAV)

Three transfer (handshake) lines are used to execute the transfer of each byte of information on the data lines. They allow asynchronous data transfer without timing restrictions being placed on any instrument connected to the bus. The transfer speed of each data byte is determined by the speed at which the slowest instrument is capable of sending or receiving data.

NRFD (Not Ready For Data) is high to indicate that all listeners are ready to accept information on the DIO lines. When NRFD is low, at least one listener is not ready for data.

NDAC (Not Data Accepted) is high to indicate that all listeners have accepted the information on the DIO lines. When NDAC is low, at least one listener has not accepted the data.

DAV (Data Valid) is low to indicate the information on the DIO lines is valid for the listener(s). When DAV is high, the information on the DIO lines is not valid.

Data Transfer

Transfer of data on the bus is asynchronous. That is, there are no restrictions placed on the data rates of instruments connected to the bus. The slowest instrument involved in a data transfer on the bus determines the rate of data transfer. As previously mentioned, transfer is under the control of three lines: DAV, NRFD and NDAC. The talker controls the data lines and DAV; the listener(s) controls NRFD and NDAC.

The transfer of a byte of data is initiated by all listeners setting NRFD high, indicating they are ready for data. When the talker recognizes that NRFD is high it places data on the lines and validates the data by placing DAV low. The listener(s) senses that DAV is low, accepts the data, then notes the acceptance by setting NDAC high. Notice that the assertive, or action state, of NRFD and NDAC is high. Because all instruments on the bus have their corresponding lines connected together, all listeners must have NRFD and NDAC high before the respective line is high. This is because a low state overrides a high. Thus, a slow listener on the bus that does not accept data and thereby asserts NRFD and NDAC as readily as other listeners will slow the rate of data transfer.

The timing diagram in figure A-2 shows the relationship of these signal lines during a data byte transfer. At t_1 NRFD is set high to indicate all listeners are ready to accept data. At t_2 the talker places data on the lines and indicates the validity of the data at t_3 by placing DAV low. At some interval between t_3 and t_4 a listener may place NRFD low but it must do so before, or at the same time that NDAC is set high. The talker may return DAV high after it detects NDAC is high, which is shown at t_5. A listener may set NDAC low (shown between t_5 and t_6) but, it must do so before NRFD is set high at t_6. When NRFD goes high at t_6, a new cycle begins equivalent to t_1.

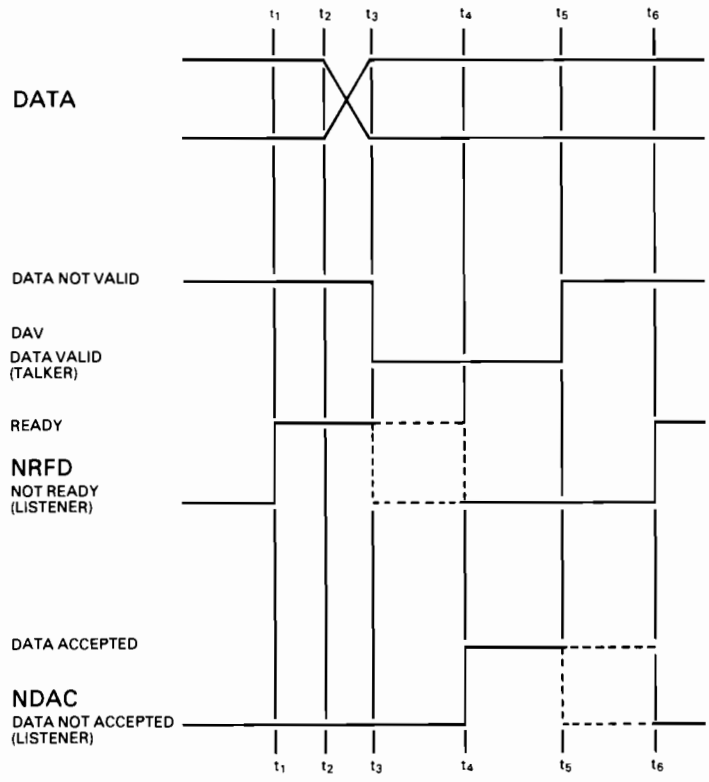


Figure A-2. Data Transfer Timing Diagram

Data Transfer Rate

The HP-IB interface is capable of transferring data at a rate of 25k bytes per second in the fast handshake mode. Normal rates can be considerably slower and are determined by the program and the devices on the bus at a given time. The slowest device determines the data transfer rate.

Line Drivers/Recievers

Each signal line connecting to the HP-IB bus has a driver and receiver circuit with the following characteristics:

	Drivers	Receivers
Type:	Open Collector	
Output Voltage Low State:	≤0.5V	Input Voltage Low State: ≤0.8V
Output Voltage High State:	≥2.5V (HP-IB Bus requires ≥2.4V)	Input Voltage High State: ≥2.0V

System Controller (SC)

There is one and only one special device on the bus known as the system controller. This capability is established by the hardware of the device itself (usually by the setting of a slide switch or a jumper) so that when power is turned on or the bus is reset, the device set to be the system controller will also assume the role of the active controller. At any time, the current active controller may pass control to any other device on the bus that is capable of performing the functions of a controller. (All devices are not required to have this capability.) The role of system controller, however, stays with the device which is physically set for that function and cannot be passed off. At any time when the system controller determines that something has gone wrong with the normal bus operations, it can reset the bus (by asserting IFC) and get back active control.

Bus Functions

Interface functions are those elements which provide the capability for a bus device to send, receive, and process messages if the device has the functional capability to do so.

Table 1-3 lists the HP-IB functions implemented by the Hewlett-Packard Interface. A complete description of these interface functions can be found in IEEE Standard 488-1978. Some of the terms used in the table are defined below.

Handshake	A technique used by devices to synchronize data byte transfers.
Source	An originator of data or commands.
Acceptor	A receiver of data or commands.
Listener	A device with Listen capability becomes Listener Active when it receives its Listen Address from the Controller Active device. As such, it is prepared to receive data bytes sourced by the Talker Active device.
Talker	A device with Talker capability becomes Talker Active when it receives its Talk Address from the Controller Active device. As such, it is prepared to source bytes to one or more Listener Active devices. There can never be more than one device as a Talker at a given time.
System Controller	At power on, one (and only one) device on the HP-IB bus assumes the roll of System Controller. The HP Series 80 Personal Computer can be configured to be the System Controller by a switch on the interface circuit board on the back panel. The System Controller resets the bus at power on and becomes Controller Active. Also, the System Controller may reset the bus and become Controller Active at any time, even if control has been passed to another bus device.

Controller Active — The Controller Active device configures the bus for the exchange of data by sourcing commands that designate one talker and one or more listeners. It can also send commands to cause specific actions to occur within a device, such as trigger, clear, etc. The Controller Active device can pass control to any other bus device capable of receiving control.

Serial Poll — The Controller Active device may serially poll another device to obtain it's status byte. This is usually done in user's software in response to a request for service. A device's status byte denotes the device's present status and whether or not it requested service.

Parallel Poll — The user can program the Controller Active device to conduct a parallel poll to obtain a status bit from devices on the HP-IB bus that are properly configured.

Table A-2. Interface Functions and Allowed Capability

Mnemonic	Function/Allowed Capability
SH1	Source Handshake, complete capability. Provides the interface with the capability to send message bytes.
AH1	Acceptor Handshake, complete capability. Provides the interface with the capability to guarantee the proper reception of message bytes.
T6	Talker. Provides the interface with the capability to send device dependent data over the HP-IB bus to other devices. T6 implements: Basic Talker Serial Poll Unaddress if my listen address (MLA)
L4	Listener. Provides the interface with the capability to receive device dependent data from the HP-IB bus from other devices. L4 implements: Basic Listener Unaddress if my talk address (MTA)
SR1	Service Request, complete capability. Permits the interface to asynchronously request service from the Controller Active device.
RL1	Remote/Local, complete capability. Provides the interface with the capability to allow the computer to select between either keyboard control (local) or program control (remote) if provided for in the user's program. Use of the RL function is totally program dependent.
PP2	Parallel Poll. Implemented by a jumper wire on the interface circuit board. Permits the interface to return a 1-bit status to the controller in the response to a parallel poll.
DC1	Device Clear, complete capability. Gives the user access to the Device Clear state; use is program dependent.
DT1	Device Trigger, complete capability. Gives the user access to the Device Trigger state; use is program dependent.
C1, 2, 3, 4, 5	Controller. This implements: C1—System Controller C2—Send Interface Clear (IFC) and Take Charge C3—Send Remote Enable (REN) C4—Respond to Service Request (SRQ) C5—Send Interface Message, Receive Control, Pass Control, Pass Control to Self, Parallel Poll, Take Control Synchronously

HP-IB Messages

The interface and I/O ROM determine how the preceding functions are implemented for passing messages over the bus. Following is a complete list of single line and multi-line messages. The manner in which some of these messages are responded to is device dependent and/or user defined. The controller is the primary source of messages that causes some specified action to take place on another bus device. Some of these messages are also discussed in more detail in section 4.

Table A-3. HP-IB Messages

Mnemonic	Message Name	Response
		Single Line Messages
ATN	Attention	The Controller Active device places ATN true to source commands on the bus or in conjuncion with EOI, to do a parallel poll. When ATN is false, data may be sent over the bus by a designated talker.
IFC	Interface Clear (Abort)	The System Controller uses this to place talkers and listeners in an unaddressed state. If control has been passed, the System Controller again becomes Controller Active when it sends IFC.
REN	Remote Enable	Removes all devices from Local Lockout mode and causes all devices to revert to manual control.
SRQ	Service Request	Indicates a device's need for interaction with the controller.
EOI	End or Identify	Terminates a flow of data, and can be used with ATN to do a parallel poll.
		Single Line Handshake Messages
DAV	Data Valid	Allows source to validate DIO lines.
NRFD	Not Ready For Data	Used to inform the source that all devices are ready for data.
NDAC	Not Data Accepted	Used by devices to inform the source that data has been accepted.
		Multi-Line Messages
GTL	Go To Local	Causes selected devices to switch to local (front or rear panel) control.
LAG	Listen Address Group	A group of 31 listen addresses, one of which corresponds to the listen address of the interface.
UNL	Unlisten	Device becomes unaddressed to listen.
TAG	Talk Address Group	A group of 31 talk addresses.
UNT	Untalk	Causes a talker to become unaddressed to talk.
LLO	Local Lockout	Prevents local (front or rear panel) control of device functions.
DCL	Device Clear	Causes all devices to be initialized to a predefined or power-up state.
SDC	Selected Device Clear	Causes a device to be initialized to a predefined or power-up state.

Table A-3. HP-IB Messages (cont.)

Mnemonic	Message Name	Response
SPD	Serial Poll Disable	Devices exit serial poll mode and are not allowed to send their status byte.
SPE	Serial Poll Enable	Devices enter serial poll mode and are allowed to send their status byte when addressed to talk.
GET	Group Execute Trigger	Signals one or more devices to simultaneously initiate a set of device dependent actions.
TCT	Take Control	Passes bus controller responsibilities from the current controller to a device which can assume the bus supervisory roll.
SCG	Secondary Command Group	A group of 32 commands which are only recognized if they immediately follow a talk or listen address.
DAB	Data Byte	Transfers device dependent information between a talker and one or more listeners. This may be programming information or data.

Note: Pressing the (RESET) key will reset the interface and return keyboard control to the operator. If the HP Series 80 Personal Computer is System Controller, pressing the (RESET) key will output the IFC message and set REN true.

HP-IB Universal Commands

The table below documents the decimal value of the HP-IB interface messages. Also shown are the numeric ranges for address and Command Groups. These commands are sent when ATN is true.

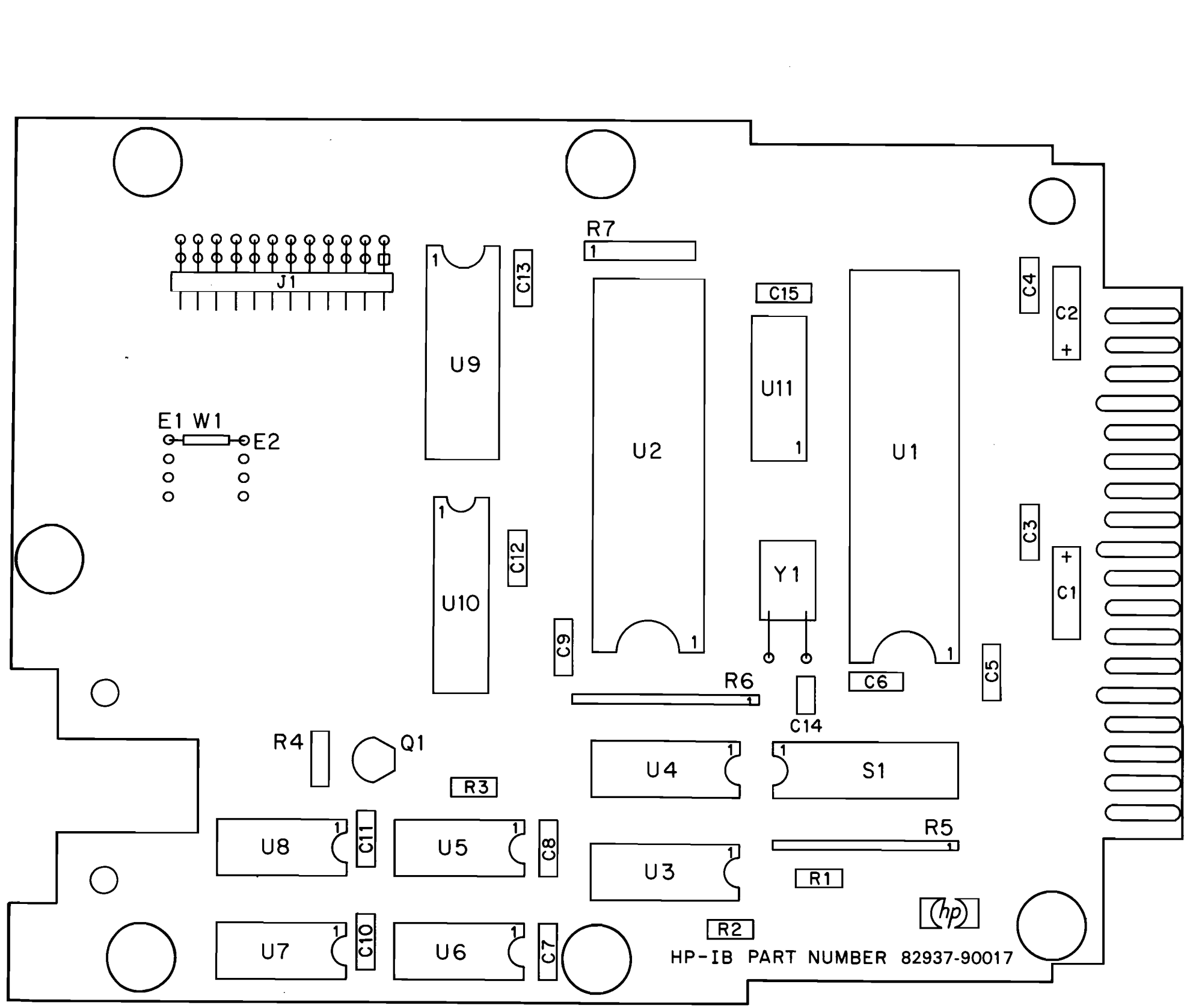
Table A-4. Primary Command Group

Decimal Value	ASCII Character	Interface Message	Description
0	NUL		
1	SOH	GTL	Got To Local
2	STX		
3	ETX		
4	EOT	SDC	Selected Device Clear
5	ENQ	PPC	Parallel Poll Configure
6	ACK		
7	BEL		
8	BS	GET	Group Execute Trigger
9	HT	TCT	Take Control
10	LF		
11	VT		
12	FF		
13	CR		
14	SO		
15	SI		
16	DLE		
17	DC1	LLO	Local Lockout
18	DC2		
19	DC3		
20	DC4	DCL	Device Clear
21	NAK	PPU	Parallel Poll Unconfigure
22	SYN		
23	ETB		
24	CAN	SPE	Serial Poll Enable
25	EM	SPD	Serial Poll Disable
26	SUB		
27	ESC		
28	FS		
29	GS		
30	RS		
31	US		
32-62	SP to > (Numbers, special char)	LAG	Listen Address Group
63	?	UNL	Unlisten
64-94	@ to > (Upper case ASCII)	TAG	Talk Address Group
95	—	UNT	Untalk
96-126	(lowercase ASCII)	SCG	Secondary Command Group
127	DEL		

Available Bus Addresses and Codes

Table A-5. Bus Addresses/Codes

Address Characters		Address Switch Settings					Address Codes		
Listen	Talk	(5)	(4)	(3)	(2)	(1)	Decimal	Octal	
SP	@	0	0	0	0	0	0	0	
!	A	0	0	0	0	1	1	1	
"	B	0	0	0	1	0	2	2	
#	C	0	0	0	1	1	3	3	
$	D	0	0	1	0	0	4	4	
%	E	0	0	1	0	1	5	5	
&	F	0	0	1	1	0	6	6	
'	G	0	0	1	1	1	7	7	
(	H	0	1	0	0	0	8	10	
)	I	0	1	0	0	1	9	11	
*	J	0	1	0	1	0	10	12	
+	K	0	1	0	1	1	11	13	
,	L	0	1	1	0	0	12	14	
-	M	0	1	1	0	1	13	15	
.	N	0	1	1	1	0	14	16	
/	O	0	1	1	1	1	15	17	
0	P	1	0	0	0	0	16	20	
1	Q	1	0	0	0	1	17	21	
2	R	1	0	0	1	0	18	22	
3	S	1	0	0	1	1	19	23	
4	T	1	0	1	0	0	20	24	
5	U	1	0	1	0	1	21	5	←Preset
6	V	1	0	1	1	0	22	26	
7	W	1	0	1	1	1	23	27	
8	X	1	1	0	0	0	24	30	
9	Y	1	1	0	0	1	25	31	
:	Z	1	1	0	1	0	26	32	
;	[	1	1	0	1	1	27	33	
<	/	1	1	1	0	0	28	34	
=	]	1	1	1	0	1	29	35	
>	^	1	1	1	1	0	30	36	



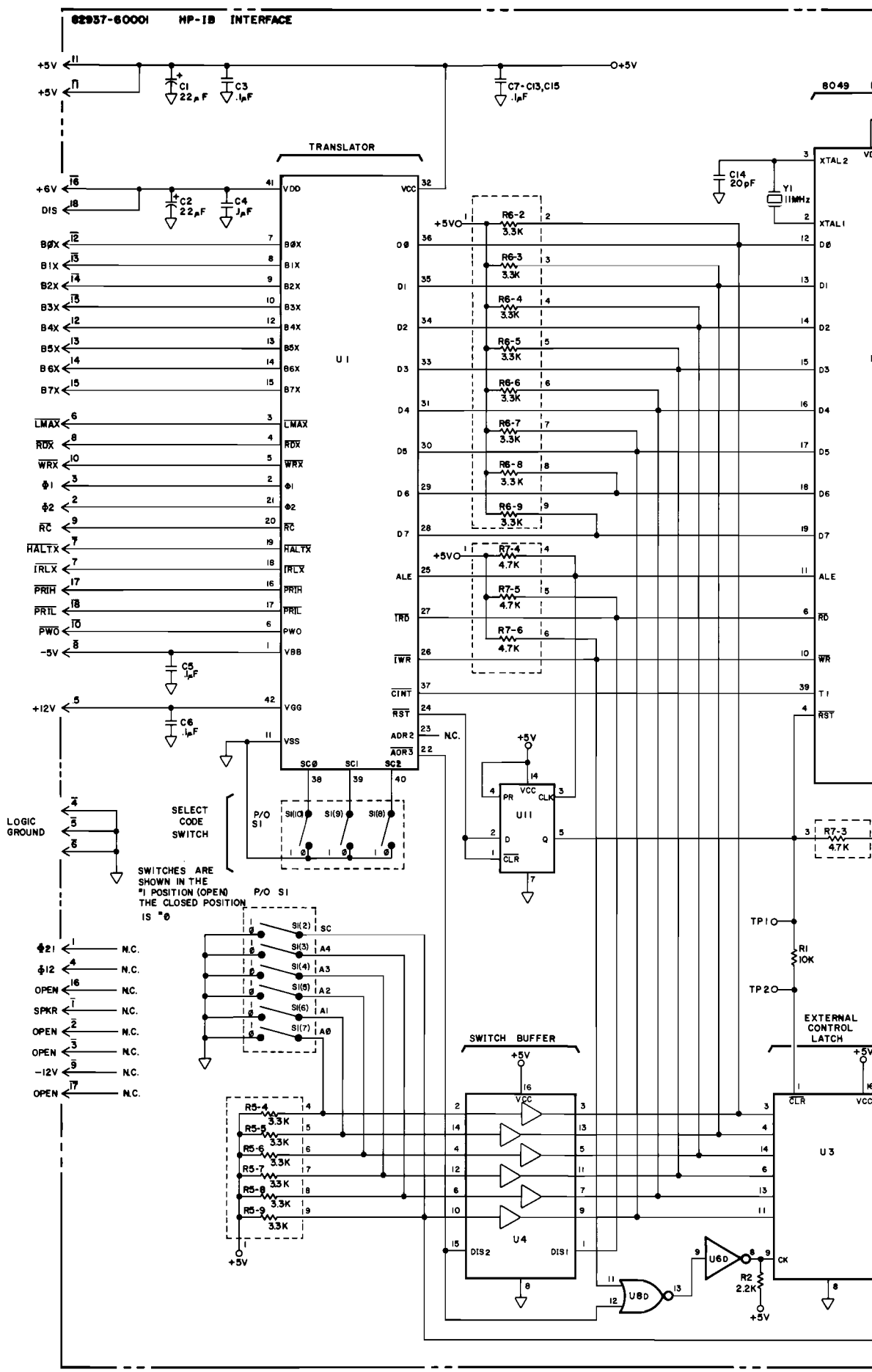


Figure A-3. HP 82937A HP-IB Interface Schematic and Component Layout Diagrams

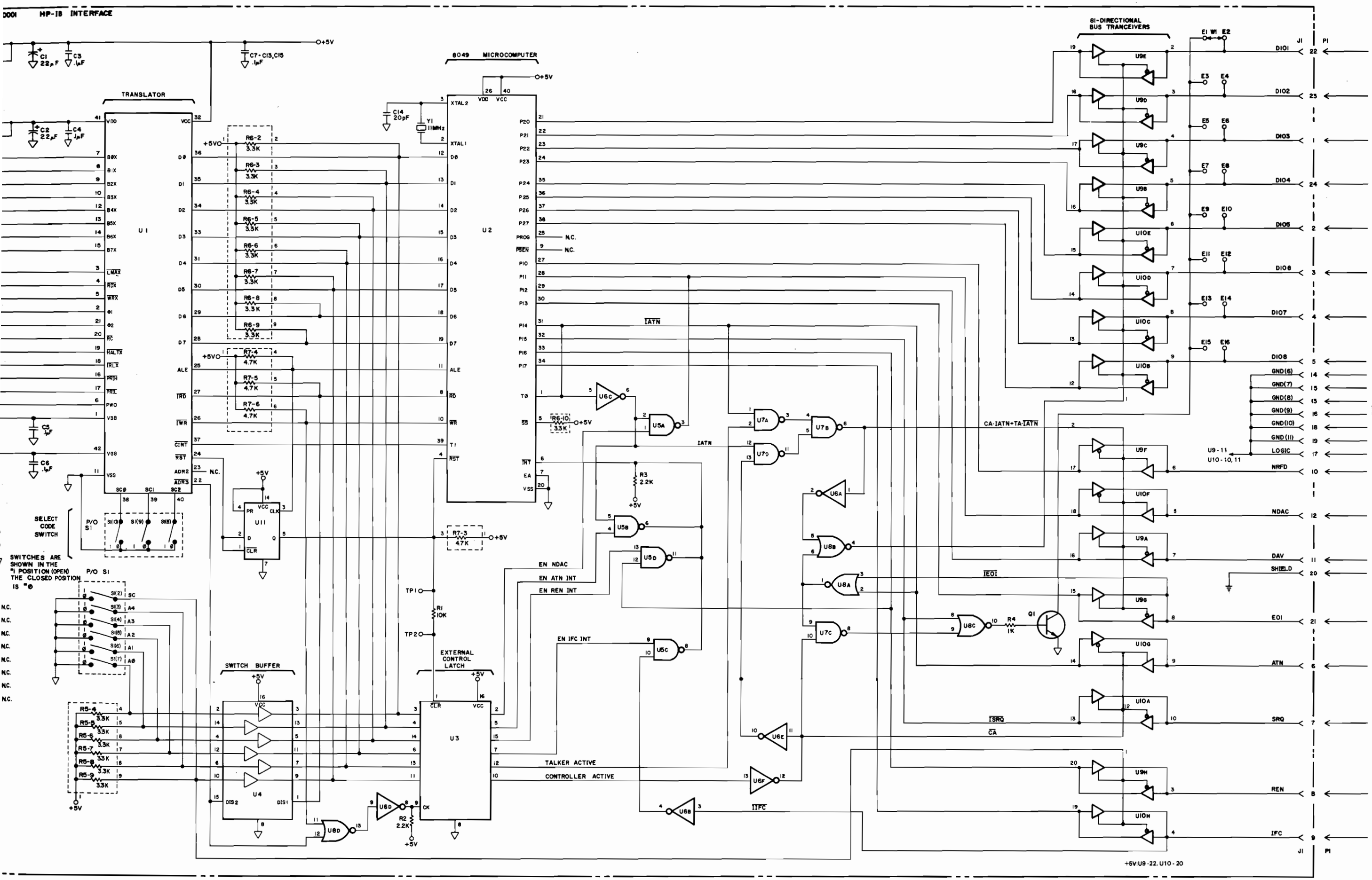

0001 HP-IB INTERFACE
TRANSLATOR
U1
8049 MICROCOMPUTER
U2
8I-DIRECTIONAL BUS TRANCEIVERS
U3
U4
U11
SELECT CODE SWITCH
P/O SI
SWITCHES ARE SHOWN IN THE "1" POSITION (OPEN) THE CLOSED POSITION IS "0"
SWITCH BUFFER
EXTERNAL CONTROL LATCH
EN NDAC
EN ATN INT
EN REN INT
EN IFC INT
TALKER ACTIVE
CONTROLLER ACTIVE
IATN
IEOI
ISRQ
IIFC
CA·IATN+TA·IATN
CA
TA
LOGIC
ground on interface card
+5V:U9-22, U10-20
GROUND
SHIELD
ATN
SRQ
IFC
NDAC
NRFD
DAV
EOI
REN
DIO8
DIO7
DIO6
DIO5
DIO4
DIO3
DIO2
DIO1
B 51000

Table A-6. HP 82937A HP-IB Interface Replaceable Parts List

Reference Designator	HP Part No.	TQ	Description
A1	82937-60901	1	PC Assembly
C1, C2	0180-0228	2	C-F: 22μF, 15V
C3-C13, C15	0160-4571	12	C-F: 0.1μF, 50V
C14	0160-4767	1	C-F: 20pF, 200V
J1	1251-5266	1	Connector: 24 Pin
MP1	0340-0883	1	Transistor Insulator
Q1	1854-0019	1	Transistor: NPN, Si
R1	0683-1035	1	R-F: 10kΩ, 5%, .25W
R2, R3	0683-2225	2	R-F: 2.2kΩ, 5%, .25W
R4	0683-1025	1	R-F: 1kΩ, 5%, .25W
R5, R6	1810-0278		Resistor Network: 3.3kΩ, 2%, 1.25W
R7	1810-0367	1	Resistor Network: 4.7kΩ, 2%, 1.25W
S1	3101-2532	1	Switch: 10 Segment SPDT
U1	1MB5-0101	1	IC: Translator
U2	1820-2437	1	IC: 8049 Microcomputer
U3	1820-1466	1	IC: 74C174, Hex D Flip-Flop
U4	1820-1402	1	IC: 80C95, Hex Buffer
U5	1820-1198		IC: 74LS03, Quad 2-Input NAND
U6	1820-1416	1	IC: 74LS14, Hex Schmitt Trigger Inverter
U7	1820-1197	1	IC: 74LS00, Quad 2-Input NAND
U8	1820-1144	1	IC: 74LS02, Quad 2-Input NOR
U9	1820-2547	1	IC: SN75162N
U10	1820-2485	1	IC: SN75160N
U11	1820-1112	1	IC: 74LS74, Quad D Flip-Flop
W1	815-3773	1	Wire Jumper
Y1	0410-1222	1	11 MHz Crystal
	82937-60902	1	Case
	82937-60004	1	Interface Cable
	0590-0199	2	Hex Nut with Lock Washer
	1400-1063	1	Top Cable Clamp
	1400-1064	1	Bottom Cable Clamp
	2200-0143	6	Screws: 4-40 Machine
	0363-0174	1	Ground Contact

Functional Test

The following program listing may be entered and run to functionally test the interface. In most instances, running this test will indicate whether or not the interface is operational.

> **Note:** Before you run the test, make sure the interface has the System Controller switch segment set to "1" and the I/O ROM is installed.

Use the following steps to enter and run the test from the HP Series 80 Personal Computer keyboard:

1. Make sure the interface and I/O ROM are installed. If necessary, refer to the installation section and install the interface and I/O ROM.
2. Peripherals may or may not be connected to the interface.
3. Turn the computer power switch to the ON position.
4. Press the following keys: (A)(U)(T)(O), then press (ENDLINE). This will cause the program lines to be numbered by ten in ascending order.
5. Refer to the following program listing and enter the first line by pressing these keys:

   (C)(L)(E)(A)(R) then press (ENDLINE)

   Continue this procedure until the entire program is entered, making sure to press (ENDLINE) as each numbered program line is entered.

```
10 CLEAR
20 DISP "HP-IB EXERCISER";"ENTER SELECT CODE"
30 INPUT S
40 DISP "ENTER # OF TIMES TO RUN TEST..."
50 INPUT N
60 RESET S
70 GOSUB 130
80 GOSUB 370
90 N=N-1
100 IF N>0 THEN GOTO 70
110 DISP "TEST COMPLETE"
120 END
130 DISP "CHECKING SWITCH SETTINGS AND";"HANDSHAKE"
140 A2=0
150 A1=53
160 WIO S,0;2
170 WIO S,1;230
180 IF RIO (S,0)<128 THEN 220
190 A2=A2+1 @ IF A2>2 THEN GOSUB 460
200 IF A2=3 THEN 220
210 GOTO 180
220 WIO S,0;0
230 WIO S,1;3
240 IF RIO (S,0)<128 THEN 280
250 A2=A2+1 @ IF A2>4 THEN GOSUB 460
260 IF A2=4 THEN 280
270 GOTO 240
280 WIO S,0;2
```

```
290 WIO S,1;120
300 WIO S,0;0
310 IF RIO (S,0)<1 THEN 310
320 A1=RIO (S,1)
330 IF RIO (S,0)#0 THEN GOSUB 480
340 IF A1>192 THEN A1=A1-192
350 IF A1#53 THEN GOSUB 50
360 RETURN
370 DISP "STATUS TEST"
380 STATUS S,0 ; B1,B2,B3,B4,B5,B6
390 IF B1#1 THEN GOSUB 540
400 IF B2#0 THEN GOSUB 540
410 IF B3#64 THEN GOSUB 540
420 IF B4#0 THEN GOSUB 540
430 IF B5#A1 THEN GOSUB 540
440 IF B6#160 THEN GOSUB 540
450 RETURN
460 DISP "HANDSHAKE ERROR-PROBABLE";"PROCESSOR OR XLATOR
    FAILURE"
470 RETURN
480 DISP "XLATOR IB NOT CLEAR WHEN";"EXPECTED-PROBABLE
    LATOR OR";"PROCESSOR FAILURE"
490 RETURN
500 DISP "SWITCHES DIDN'T READ AS EXPECTED DEFAULT"
510 A$=DTB$(A1)
520 DISP "SWITCHES 2-7 READ AS '"&A$[11,16]&"-EXPECTED
    110101"
530 RETURN
540 DISP "STATUS ERROR"
550 DISP "STS BYTES=";B1;B2;B3;B4;B5;B6
560 RETURN
```

6. After the program is entered press (RUN).

   The screen should display:

```
HP-IB EXERCISER
ENTER SELECT CODE
?
```

7. Respond by entering the select code the interface is set to. For example, if select code 7 is set, press (7) and then (ENDLINE). If the wrong select code is entered, one or more error messages will appear on the screen after step 9 is performed. If this happens, press (RUN) and enter the select code again.

8. The screen should then display:

```
ENTER # OF TIMES TO RUN TEST
?
```

9. Enter the number of times you wish to run the short test. To run the test once, press [1] and then [ENDLINE]; to run the test ten times, press [1] [0] and then [ENDLINE], etc. Running the test several times may be useful if you suspect an intermittent failure.
10. The screen should then display:

```
CHECKING SWITCH SETTINGS AND
HANDSHAKE
```

If this test passes, shortly thereafter the screen will display:

```
STATUS TEST
```

The test names are displayed on the screen the number of times you have specified for the test to run unless there is a failure. If a test fails, an error message is displayed for that test. If the first test fails, the screen should display:

```
HANDSHAKE ERROR-PROBABLE
XLATOR OR PROCESSOR FAILURE
```

OR

```
XLATOR IB NOT CLEAR WHEN
EXPECTED-PROBABLE XLATOR OR
PROCESSOR FAILURE
```

If the factory settings of switch segments 2 through 7 have been changed, you can always expect the screen to display:

```
SWITCHES DIDN'T READ AS EXPECTED
DEFAULT

SWITCHES 2-7 READ AS
```

(This portion of the display will be a series of 0's and/or 1's depending on how the computer interprets the switch settings. The first 0 or 1 is for switch segment 2, the second is for switch segment 3, etc. Switch segment 1 and 8 through 10 are not read.)

```
-EXPECTED 110101
```

(These are the factory settings.)

If the above is displayed, it does not necessarily mean that the switches are being read improperly. Except for switch segment 2, which must always be 1 to enable the computer as System Controller, the switches may be set to any combination. This feature permits you to set the swtiches for any talk/listen address given in figure 2-8 and then verify the switches are being read properly for that address.

If the status test fails, the screen will display:

```
STATUS ERROR
STS BYTES
```

(This portion of the display will be a series of numbers indicating which bits are set in status registers SR0 through SR5, SR6 is not read. the first number is for SR0, the second for SR1, etc. These status registers are shown in the discussion in the section entitled Controlling with the HP-IB Interface.

An example of a proper status byte would be:

```
1   0   64   0   53   160
```

The only number that may vary in a proper status byte is the one indicating the bits set in SR4. This is shown as 53 in the above status byte and is discussed in more detail in the following discussion.

If a status error is displayed, you can identify which bits of a register are set by decoding the decimal number printed into its binary equivalent. Then refer to the status registers on page 54. The status register bits are shown with bit 0 being the least significant and bit 7 the most significant. The decimal values of the bits are as follows:

SR Bit	Decimal Equivalent
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128

The numbers of the proper status byte can therefore be interpreted as follows:

- The first number, 1, indicates that bit 0 in SR0 is set to 1 and all others are 0.
- The second number, 0, indicates that all bits of SR1 are 0.
- The third number, 64, indicates that bit 6 of SR2 is set to 1 and all others are 0.
- The fourth number, 0, indicates that all bits of SR3 are 0.
- The fifth number, 53, indicates that bits 5, 4, 2, and 1 of SR4 are set to 1 and all others are 0.
- The sixth number, 160, indicates that bits 7 and 5 of SR5 are set to 1 and all others are 0.

Now, let us examine the fifth number, 53, which is the only number that may vary in a proper status byte. This number reflects the contents of SR4, the address register. This register should always coincide with how switch segments 2 through 7 are set, which was discussed earlier in this section. The number 53 reflects that the system controller switch and the talk/listen address switches are all in the factory preset positions. The smallest number that should ever appear in this position while running the test is 32, corresponding to the system controller switch being set to 1 and the talk/listen address switches set for address 0. The largest number that should ever appear in the fifth position is 62, corresponding to the system controller switch being set to 1 and the talk/listen address switches set for address 30.

11. After all the tests are run, the screen should display:

```
TEST COMPLETE
```

Notes

Appendix B

A Basic Introduction to the HP-IB

General Structure of the HP-IB

The letters HP-IB stand for **H**ewlett-**P**ackard **I**nterface **B**us. It is Hewlett-Packard's implementation of the IEEE-488-1978* interface standard. The purpose of the HP-IB is to provide for mechanical, electrical, timing, and data compatibilities between all devices adhering to the standard†. In essence, interfacing computers to other devices has been simplified by the HP-IB. Instead of worrying about **how** to hook up your devices so they can communicate, you merely have to consider **what** they are going to communicate. Given that all the details of compatibility have been worked out, we can move on to understanding the general structure of a **system** tied together with the HP-IB, commonly called **the bus**.

Like most things one learns about, the HP-IB has a structure—a precise organization—that prevents chaos from becoming the general rule. 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 describe the manner in which business is to be conducted. For the HP-IB, these rules of order are the IEEE-488-1978 standard.

One member, designated the **committee chairman**, is set apart for the purpose of conducting the meetings and organizing the agenda. The chairman is responsible for the actions and results of the committee, and generally uses the rules of order to ensure the proper conduct of business. If for some reason the committee chairman cannot attend a meeting, he designates some other member to be acting chairman.

For the HP-IB, the committee chairman is the **system controller**. This is generally established by a switch setting on the interface, and cannot be changed under program control. However, it is possible to designate an "acting chairman" on the HP-IB: this device is called the **active controller**, and may be any device capable of directing HP-IB activities, such as a computer. When the system controller is first turned on, or reset, it assumes the role of active controller. These responsibilities may be subsequently passed to another device while the system controller tends to other business: more than one computer can be connected to the HP-IB at the same time.

On a committee, only one person at a time may speak (this alleviates confusion somewhat) and the chairman is responsible for **recognizing** appropriate members to speak, one at a time. For the HP-IB, the device designated to speak is called the **active talker**, and there may be only one active talker at any time. The act of recognizing or giving the floor to that device is called **addressing to talk**, and is performed by the active controller. The **message** delivered by the active talker is called a **data message**, and is the primary function of the HP-IB.

On a committee, those members present usually listen, but this is not the case for the HP-IB. The active controller selects which devices are going to listen, and commands all other devices to ignore the present data message. A particular device is told to listen by being **addressed to listen** by the active controller. It is then an **active listener**. Devices are told to ignore a data message by being **unaddressed** with an **unlisten command** from the active controller.

*Institute of Electrical and Electronics Engineers 488-1978 standard for a general-purpose interfacing bus, commonly termed the GPIB.
†You should be aware that some devices claiming "IEEE-488 compatible" really are not fully compatible.

The concept of **unlisten** seems at first confusing, and one might wonder why it is done. Imagine a slow note-taker on our committee, and a fast note-taker. Suppose also that the speaker is allowed to talk no faster than the slowest note-taker. This would guarantee that everybody gets the full set of notes and misses no important information. However, requiring all presentations to go at that slow pace certainly imposes a restriction on our committee. Now, if the chairman knows which presentations are not important for the slow note-taker, he can direct that person to put away his notes for those presentations. That way, the speaker and the fast note-taker can cover more items in less time. On the HP-IB, a similar situation may exist. A printer and a flexible disc can both be listeners, but if all the data transfers must be slow enough for the printer to keep up, saving a program on the disc would take as long as listing the program on the printer. That would not be a very effective use of the speed of the disc drive, certainly. Instead, by unlistening the printer, the computer can save a program as fast as the disc can accept it.

Now that you have seen the general structure of the HP-IB, you are ready to learn about the primary function of the bus—that of data transfers.

Data Transfers on the HP-IB

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, and this transfer occurs 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 designated to listen.

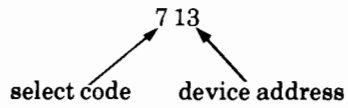
The technical term for the mechanism used to control the transfer rate is called **handshake**. Imagine, on the committee, that the speaker must pause every time a note-taker raises his hand. When the speaker presents one item, a slow note-taker can raise his hand to halt the speaker, while he (the note-taker) finishes jotting down that item. Then, when he is ready, the note-taker lowers his hand and the speaker continues with the next item.

A similar mechanism is used for the HP-IB. As you might expect, the HP-IB handshake is more complex in order to deal with many different devices and their different modes of operation. It is not necessary, however, to understand the details of this handshake in order to use the HP-IB.

How does one **address** a device? Remember, this is necessary in order to designate a device to be either a talker or a listener for a data transfer. The **select code** of the interface, 7 in the case of an HP-IB interface, is the first part of a **device selector**. This corresponds to the street number of an apartment house, say, 315 Exray Drive. The second part of a device selector is the **device address** (13, for our example) on the bus. This corresponds to the apartment number of the person in the apartment house. To address a letter to this person, it is necessary to give both the street address and the apartment number, like so:

315 Exray Drive]←street address
Apartment #27]←apartment number

The complete device selector looks like this:



Every device on the bus must have a different device address, or a great deal of confusion will result (as you might intuitively expect).

For a data transfer to occur, the active controller must:

1. Unlisten all devices (to ensure there are no "eavesdroppers").
2. Designate the talker (by **addressing** a device to talk).
3. Designate the listener(s) (by **addressing** the device(s) to listen).
4. Indicate to all devices involved that they may begin the data transfer.

Fortunately, this is normally done automatically by the computer. There are exceptions to this, but exceptions are topics for more advanced discussion and are not presented here.

The computer, as the active controller, is normally involved in any data transfers that occur. Take the case of the computer sending data to a printer. Suppose the message to be printed is "Now is the time for all good men". The statement

```
OUTPUT 701;"Now is the time for all good men"
```

does the following:

1. Unlistens all devices.
2. Designates the computer as talker (**My Talk Address** is sent).
3. Designates the printer (device #01) as listener.
4. Begins the data transfer.

The computer receives data as simply as it sends it. Assume that the computer is to take a reading from a voltmeter, which is reading + 1.075 volts. The device address of the voltmeter is 03, so the appropriate statement

```
ENTER 703;V
```

does the following:

1. Unlistens all devices.
2. Designates the computer as listener (**My Listen Address** is sent).
3. Designates the voltmeter (device #03) as talker.
4. Begins the data transfer, and stores the value in the variable 'V'.

Notice in both examples the computer only needs to be told what the address of the other device is—it already knows it own address.

If transferring data were all the HP-IB did, the simplicity of hooking up devices to the computer would be the only strong point of the bus. Its power goes far beyond that, however, as you will see in the next two sections.

Controlling the HP-IB

Just as the committee chairman must have certain measures of control over the committee, so must the system controller have methods of controlling the bus.

When a committee meeting has gotten out of hand for some reason (two or more members talking at once, or a member refusing to give up the floor) the chairman must call the meeting to order, usually by pounding a gavel. The possibility of a bus out of control exists, therefore, the system controller (and **only** the system controller) has the power to assert **Interface Clear**. This terminates all bus activity, **unaddresses** all talkers and listeners, and passes active controller responsibilities back to the system controller (if active control had been passed previously to some other device). Interface clear is not normally used in most bus operations, but has been provided to deal with any situations that may occur.

Individual devices on the bus may be reset back to their power-on state by sending the **device clear** message. The active controller can send this message to any one device or all devices at once. **Device Clear** can be a quick means of re-programming a device back to its default (or power-on) parameters, and that device generally acts as if it had just been turned on.

Passing active control has been mentioned twice, and it refers to the ability of delegating active controller responsibilities to another device.

The analogy of designating an acting chairman in the absence of the committee chairman is appropriate. The system controller may need to direct the activities of another HP-IB system, or to perform calculations on data that has been gathered, or some other function that demands attention away from the bus. The technical term for passing active control is **Take Control**. It is a short message sent via the bus from the current active controller to the device receiving control which then assumes active control responsibilities. Obviously it doesn't make sense to pass control to a device incapable of controlling bus activities, such as a printer.

The controller also has certain functions available that affect selected devices on the bus. It is sometimes desireable to have two or more instruments start their operations at the same time. Suppose two voltage readings have to be taken simultaneously at different points in a circuit under analysis. The two voltmeters involved have to be **triggered** at the same time, so the controller can send the **Group Execute Trigger** message to accomplish this.

Another type of control necessary in HP-IB systems is that of remote control of instruments. Normally, an instrument such as a voltmeter has switches, dials, and buttons on the front panel which the operator adjusts to set range, polarity, and function. Requiring the presence of an operator to control these settings, however, would impose a severe restriction to the ability to automate a system. The **Remote Enable** message places an instrument under remote control so it can be set up by the active bus controller.

There is still a problem with the system described above. It is possible that a casual passer-by may come along and put an instrument back under local control of the front panel switches. This can be done with a switch generally provided on instruments called **return-to-local**. If this happens, the computer could begin receiving erroneous readings, so a bus message is provided to **lock out** the return-to-local switch. The message is appropriately called Local Lockout. Its primary purpose it to prevent manual operation of bus instruments when they are under remote control.

Naturally, there needs to be some method of returning the instruments back to manual control. This is done by sending the **Go-to-Local** message. When a device receives the Go-to-Local message, any local lockout and remote messages in effect on that device are cancelled. The device then returns to local, front-panel control.

This completes the introduction to controller-directed activities, but one more aspect needs to be covered before this discussion of the HP-IB is complete. Devices on the bus need some mechanism of getting the attention of the controller when unusual circumstances develop. The next section deals with this mechanism.

Handling Requests for Service

When a device encounters a problem, or is ready with some other information for the controller, it needs to be able to signal that such a situation exists. It does this by a **Service Request**. An example of a need for a service request is that of a printer out of paper. The printer has to stop printing, but it needs to indicate to the controller that it has stopped, so that data will not be lost. The printer sends a service request to the controller, which must then find out

1. **Who** sent the request (if there are several devices on the bus)?
2. **What** is the problem?

The controller can determine who sent the service request by performing a **Parallel Poll***. This is like the committee chairman asking for those members with a problem to raise their hand. Once he determines which members have a problem, he can ask each in turn what their problems are. On the HP-IB, this process of determining the problem is called a **Serial Poll** and, like the name implies, involves querying each device in turn as to its status. The information returned by each device is dependent upon the nature of the device: a printer may indicate out-of-paper, cover-off, ribbon low, or whatever. A voltmeter may indicate overrange, illegal command, and so on. On one hand, the functions available with the HP-IB interface make possible a very sophisticated automated system. On the other hand, a very simple system is possible, because all the details of electrical, timing, mechanical, and data compatabilities have been already taken care of for you. Section 3 shows how an HP-IB system might be set up, and the sequence of operations typically used to operate it.

* Note that response to Parallel Poll is device-dependent, so it may not be possible to use your device in this manner.

Notes

Appendix C

HP-IB Errors

The HP-IB interface generates the error messages listed below. Mainframe errors are numbered 1 through 92.

Error Number	Message	Error Condition
101	TRANSFER	Program was paused with an I/O TRANSFER still active.
110	I/O CARD	Interface failed a self-test, indicating a possible hardware problem.
111	I/O OPER	The I/O operation attempted is not valid with the HP-IB interface.
112		The plug-in ROM has failed a check-sum test, indicating a possible hardware problem, not likely the interface.
113		Not system controller. The statement executed requires that the computer be the system controller.
114		Not active controller. The statement executed requires that the computer be the current active controller.
115		Not active talker. The statement executed requires that the computer be addressed to talk. A possible cause for this error is that a PRINT statement was executed after a statement specifying only the select code was executed. The peripheral address code must be specified with HP-IB devices to establish the current active talker and listener.
116		Not active listener. The statement executed requires that the computer be addressed to listen. A possible cause for this error is that the computer has not been addressed to listen (but has been previously addressed to talk with a PRINTER IS statement) when a PLOTTER IS statement that only specifies a select code is executed. The peripheral address code must be specified with HP-IB devices to specify the current active talker and listener.
117		Interface is active controller. The statement executed requires that the computer be non-controller.
123	NO ";"	Syntax error. A semicolon delimiter was expected in the statement.
124	ISC	No device currently set to the specified select code.
125	ADDR	Improper primary address specified. No device at that address.
126	BUFFER	Buffer error. IOBUFFER not declared for the variable specified, buffer is full at execution of an ENTER, or buffer is empty at execution of a TRANSFER or OUTPUT.
127	NUMBER	If encountered in an input operation, error indicates that the incoming character sequence does not constitute a number. If encountered in an output operation, the number has exeeded the range of the specified "e" format.
128	EARLY TERM	Buffer was emptied before COUNT or ENTER fields were satisfied.

Error Number	Message	Error Condition
129	VAR TYPE	The type variable in an ENTER list does not match the image specified for that variable.
130	NO TERM	No required terminator received from interface or buffer during an ENTER. Default requirement for a line-feed statement terminator is still in effect.

Examples:

SET I/O 7,10,0	ERROR 111 because Control Register 10 does not exist for the HP-IB interface module.
REMOTE 7	ERROR 113 if not system controller.
OUTPUT 702;A$	ERROR 114 if not active controller.
OUTPUT 7;A$	Error 115 if computer is not active talker (but is active controller).
ENTER 7;A$	Error 116 if computer is not active listener but is active controller.
REQUEST 7;*byte*	Error 117 if active controller (only non-controllers may request service).

HP-IB (I/O ROM) Statement Summary

The following table summarizes the interface conditions that must be met for a given interface state and program statement, and the resultant actions that are performed. Note that these statements are provided by the I/O ROM and are not inherent in the interface itself.

Statement	Given current state(s) of:	Additional required conditions are:	Actions Performed (Bus sequences are in color)
ABORTIO 7	SC	none	IFC, assumes active control. [IFC]
	SC & CA	none	MTA, leaves ATN true. [ATN • MTA]
	SC & CA	none	Terminates I/O operation. [No sequence generated]
ASSERT 7;X	any	none	Immediate write to CR2. IFC is not asserted.
CLEAR 701	CA	none	Addressing performed, then send SDC to device 01. [ATN • UNL, MTA, LAG, SDC]
CONTROL 7,*n*;X	any	none	Writes X to CRn when interface becomes non-busy.
ENABLE INTR 7;X	any	none	Writes X to CR1 when interface becomes non-busy.
ENTER 705;X	CA	none	Device 05 is addressed to talk, the computer is addressed to listen, data is input to X. [See ENTER sequence table]
ENTER 7;X	CA	LA	Inputs data to X. [No sequence generated]
	CA	wait for LA	Waits until addressed to listen then inputs data to X. [No sequence generated]
HALT 7	any	none	Terminates I/O operation. [No sequence generated]
LOCAL 7	SC	none	REN is set false. [REN]
LOCAL 701	CA	none	Addressing is performed, then GTL is sent. Note: ATN remains true. User may use RESUME 7 to set ATN false. [ATN • UNL, MTA, LAG, GTL]
LOCAL LOCKOUT 7	CA	none	LLO is sent. [ATN • LLO]

Statement	Given current state(s) of:	Additional required conditions are:	Actions Performed (Bus sequences are in color)
OUTPUT 705;X	CA	none	HP Series 80 Personal Computer is addressed to talk, device 05 is addressed to listen, data X is sent. [See OUTPUT sequence table]
OUTPUT 7;X	CA	TA	Outputs data X. [No sequence generated]
	CA	wait for TA	Waits until addressed to talk, then output data X. [No sequence generated]
PASS CONTROL 715	CA	none	Device 15 is addressed to talk then TCT is sent. [ATN ● UNL, MLA, TAG, UNL, TCT, ATN]
PASS CONTROL 7	CA	none	No addressing. Send TCT. [ATN ● UNL, TCT, ATN]
PPOLL (7)	CA	none	Sends IDY (Identify). [ATN ● EOI, (>6μs), ATN ● EOI]
REMOTE 7	SC	none	REN is set true. [REN]
REMOTE 701	SC	none	REN is set true, then device 01 is addressed. Note: ATN is left true. [REN, ATN ● UNL, MTA, LAG]
REQUEST 7;X	CA	none	If bit 6 of X is 1, then SRQ is set true. The computer then sends X in response to a serial poll and drops SRQ if set.
RESET 7	any	none	Sets the HP-IB interface to its power-on state. If system controller, then IFC is asserted and REN is turned off then on again.
RESUME 7	CA	none	Sets ATN false. [ATN]
SEND 7;*commands*	CA	none	Sends specified commands with ATN true. ATN is left true.
SEND 7;*data*	any	TA	Sends specified data with ATN false. ATN is left false.
SPOLL (7)	CA	LA	Conducts a serial poll. [ATN ● SPE, ATN, <data>, ATN ● SPD, UNT]
SPOLL (724)	CA	none	Addresses device 724 to talk then conducts serial poll. [ATN ● UNL, MLA, TAG, SPE, ATN, <data>, ATN ● SPD, UNT]
STATUS 7,*n*;X	any	none	Sets X to the value of SRn.
TRIGGER 7	CA	none	Sends GET. [ATN ● GET]
TRIGGER 70	CA	none	Addresses device 01, sends GET. Note ATN is left true. [ATN ● UNL, MTA, LAG, GET]

HEWLETT PACKARD

Personal Computer Division

1010 N.E. Circle Blvd., Corvallis, OR 97330

Reorder No.
82937-90017

Printed in U.S.A.
82937-90026