

HP 82401A

HP-IL Interface Owner's Manual

For the HP-71



Notice

Hewlett-Packard Company makes no express or implied warranty with regard to the keystroke procedures and program material offered or their merchantability or their fitness for any particular purpose. The keystroke procedures and program material are made available solely on an "as is" basis, and the entire risk as to their quality and performance is with the user. Should the keystroke procedures or program material prove defective, the user (and not Hewlett-Packard Company nor any other party) shall bear the entire cost of all necessary correction and all incidental or consequential damages. Hewlett-Packard Company shall not be liable for any incidental or consequential damages in connection with or arising out of the furnishing, use, or performance of the keystroke procedures or program material.



HP 82401A
HP-IL Interface
Owner's Manual

For the HP-71

November 1983

82401-90001

Introducing the HP-IL Interface

The HP 82401A HP-IL Interface greatly expands the capabilities of the HP-71. It makes the HP-71 part of an HP-IL system—enabling the HP-71 to interact with a wide variety of HP-IL peripheral devices and perform numerous input/output operations.

The HP-IL interface provides the HP-71 with the following features:

- Convenient assignments for printer and display output.
- Mass storage statements that are extensions of those of the HP-71.
- Six methods for specifying HP-IL devices.
- Simple addressing (for convenience) or extended addressing (for more than 30 devices).
- Operation as an HP-IL controller or operation as an HP-IL device.

These features are combined into one product that lets you easily perform these types of input/output operations:

- Printer and display output operations.
- Mass storage operations.
- General-purpose I/O operations.

With the HP-IL interface, the HP-71 becomes a truly powerful component in your HP-IL system.

Contents

How To Use This Manual	9
Learning About the HP-IL Interface	9
Learning About HP-IL Devices	10

Part I: Overview

Section 1: Getting Started	12
Connecting the Interface	12
Installing the Interface	12
Connecting Peripheral Devices	13
Disconnecting the Interface Loop	13
Using the HP-IL System	14
Turning the System On and Off	15
Resetting the HP-IL System	16
Specifying HP-IL Devices	17
Section 2: Printer and Display Operations	22
Assigning the Printer Device	22
Assigning the Display Device	23
Using Printers and Display Devices	24
Choosing a Device	25
Startup Assignments	26
Section 3: Mass Storage Operations	28
Preparing the Medium	28
Working With Files	29
Using Data Files	30
Creating a Data File	30
Accessing a Data File	31
Packing the Medium	31
Section 4: General I/O Operations	34
Introduction	34
Operating As a Controller	35
Sending Data	35

Fetching Data	38
Using HP-IL Addresses	39
Controlling the Loop	41
Fetching Device Information	44
Setting the HP-IL Timeout	46
Using HP-IL Interrupts	47
Sending HP-IL Messages	50
Passing Control of HP-IL	51
Giving Up Control	52
Acquiring Control	53
Operating As a Device	55
Taking an HP-IL Address	55
Sending Data	56
Receiving Data	58
Receiving BASIC Commands	59
Sending Device Information	62
Requesting Service	63
Using HP-IL Interrupts	64
Sending HP-IL Messages	67
Using Binary Functions	68

Part II: Keyword Dictionary

Organization	72
Reading Syntax Diagrams	73
Definitions of Standard Terms	74
Keyword Entries	77

Appendixes

Appendix A: Care, Warranty, and Service Information	204
Care of the Interface	204
Verifying Proper Operation	205
Limited One-Year Warranty	205
Service	207
Potential for Radio/Television Interference (for U.S.A. Only)	210
When You Need Help	211

Appendix B: Operating Information	212
Reset Conditions	212
System Flag Summary	214
System Memory Requirements	214
HP-71 Responses As a Device	215
Appendix C: HP-71 Character Set	220
Appendix D: Mass Storage Compatibility	224
Compatibility With Other Hewlett-Packard Computers	224
Mass Storage Format	226
Appendix E: Error Messages	232
Alphabetical Message Listing	232
Numerical Message Listing and Descriptions	234
Appendix F: Glossary	238
Subject Index	246
Keyword Index and Summary	Inside Back Cover

How To Use This Manual

Learning About the HP-IL Interface

This manual describes the capabilities provided to the HP-71 by the HP 82401A HP-IL Interface. It also provides information that enables you to use these capabilities effectively. The manual consists of three divisions that contain different types of information:

- **Part I: Overview.** This gives general information about connecting and using your HP-IL system. It provides an overview of the types of operations you can perform using the HP-IL interface and introduces you to the keywords you'll use.
- **Part II: Keyword Dictionary.** This gives detailed information that helps you learn about each keyword. It also serves as a source of reference information about individual keywords.
- **Appendixes and Indexes.** This provides supplementary information that may be useful in certain situations.

To learn how to use the HP-IL interface most effectively, take advantage of the manual's organization by using it this way:

1. Learn how to connect your HP-IL system: Read section 1.
2. Learn about basic HP-IL concepts and operations: Read section 1.
3. Get an overview of the general types of HP-IL operations that you want to perform, and find out which keywords perform them:
 - For printer operations, read section 2.
 - For display operations, read section 2.
 - For mass storage operations, read section 3.
 - For other general-purpose operations, read section 4 (as far as needed).
4. Find out how information is presented in the Keyword Dictionary: Read the Keyword Dictionary "Organization."
5. Learn how to use the keywords that perform the operations you want to perform: Look them up in the Keyword Dictionary.

For example, if you have an HP 82161A Digital Cassette Drive, you'll want to read sections 1 and 3, then use the Keyword Dictionary to learn about mass storage keywords like `CAT` and `COPY`.

By following these guidelines, you'll learn how to perform the operations you want—without spending time learning about other operations. Later, you can explore other capabilities provided by the HP-IL interface.

Learning About HP-IL Devices

The HP-IL interface enables the HP-71 to become part of an HP-IL system. The HP-IL peripheral devices that you connect to the interface also become part of your system.

Each HP-IL device has unique operating features. These features are described in the owner's manual for each device. The device's manual should be your primary source of information about how that device operates. It's an important reference, especially if you want to perform special operations.

Of course, certain classes of devices, such as printers or mass storage devices, have some features in common. You can use your basic knowledge of these common features to anticipate how your device will probably respond to HP-IL operations.

The examples in this HP-IL interface manual illustrate operations using typical HP-IL devices. The examples are primarily intended to show how to perform the operations. If your HP-IL device is similar to the ones in the examples, you'll probably gain even more insight into how you can use your system.

Part I Overview

Section 1

Getting Started

This section explains how to connect and start using your HP 82401A HP-IL Interface. It introduces the HP-IL (Hewlett-Packard Interface Loop) concept and explains some basic ideas about using your HP-IL system.

Connecting the Interface

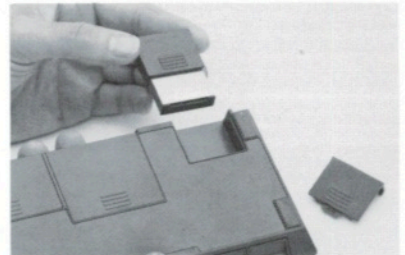
Your HP-IL system consists of your HP-71, the HP 82401A HP-IL Interface, and one or more peripheral devices. These should be connected according to the instructions below.

CAUTION

Be sure the HP-71 is turned off before connecting or disconnecting the interface and cable connectors. If this is not done, the system's operation may be disrupted.

Installing the Interface

The HP 82401A HP-IL Interface plugs into the HP-IL port at the back of the HP-71. First slide the cover off the port, then insert the interface. Be sure the slot in the end of the interface is towards the bottom of the computer. Push in the interface until it snaps into place.



Connecting Peripheral Devices

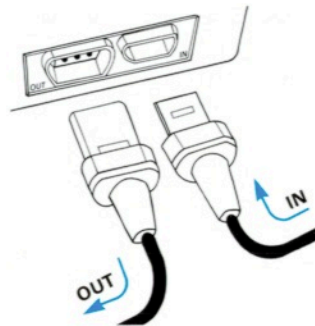
The peripheral devices in the interface loop may be connected to the interface in any order—but all of the interface cables must form a continuous loop. The connectors are designed to ensure proper orientation.



To connect a peripheral device:

1. Turn off the HP-71.
2. Disconnect the loop in one place and connect the new device into the loop at that place.
3. To ensure proper operation, make sure all peripheral devices are turned on.
4. Turn on the HP-71.

The connectors indicate the direction of information transfer as shown below:



If *any* HP-IL device is turned off (and it can't be turned on by the HP-71), it will prevent HP-IL operation.

Disconnecting the Interface Loop

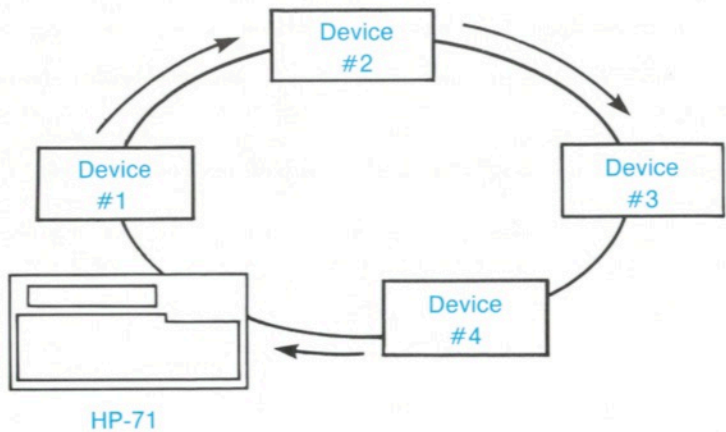
To remove any device from the interface loop, first turn off the HP-71. Then unplug the device from the loop and reconnect the loop where the device was removed.

To remove the HP-IL interface from the HP-71, first turn off the HP-71. Then pull the interface out of the port and install the original cover on the port.

Using the HP-IL System

The Hewlett-Packard Interface Loop is easy to use and understand. By using the HP 82401A HP-IL Interface, your HP-71 can interact with HP-IL peripheral devices, including printers and mass storage devices.

The HP-71 and all devices included in the HP-IL system are connected together in series. They communicate using HP-IL *messages*, which convey instructions and data. Each HP-IL message sent on the loop is passed sequentially from one device to the next. If the message doesn't affect a particular device, the device passes the message on to the next device in the loop. When the message reaches the intended device, the device responds as directed and (usually) passes the message on to the next device. Eventually, the message returns to the sender. All messages travel in one direction only.



If any device isn't turned on, it will probably disrupt the operation of the HP-IL system because it can't pass messages around the loop.

All components of an HP-IL system have different roles at different times, depending upon the operation being performed. Four roles are defined for HP-IL components:

- **Controller:** The component that's responsible for issuing instructions to other components in the HP-IL system. Only one component can be the controller—remaining components are often referred to as *devices*. The role of controller can be transferred to another device, but there can be only one controller at any time.
- **Talker:** A component that has been assigned by the controller to send data to other devices. Only one component at a time can be a talker. The controller can be a talker.
- **Listener:** A component that has been assigned by the controller to receive data from the talker. Several components can be listeners at one time. The controller can be a listener.
- **Idle:** A component is idle if it's not a talker or listener—that is, it's not involved in a data transfer. Although a component may be idle, it still responds to instructions issued by the controller and still passes all messages to the next device.

For example, consider the HP-IL system illustrated above. If the HP-71 is in charge of the HP-IL system (which it usually is), then it's the *controller*. Suppose the HP-71 wants device #3 to send data to the HP-71. The HP-71 assigns device #3 the role of *talker*, and the HP-71 takes the role of *listener*. All other devices remain *idle*. The HP-71 then instructs the talker to send its data, which is received by only the listener. When the data has been transferred, the HP-71 cancels the talker and listener roles. Of course, the HP-71 *automatically* takes care of these details for all operations—you merely tell the HP-71 what operation you want it to perform.

The following topics present important information about how you use the HP-71 to perform HP-IL operations. Be sure you understand these basic concepts.

Turning the System On and Off

If each HP-IL device is either turned on or able to be turned on by the controller, the HP-IL system is activated whenever you turn on your HP-71. Normally, the HP-71 automatically assigns addresses to all devices, so that your system is immediately ready to perform the operations you specify.

Note: If you want to use your HP-71 *without* connecting or turning on HP-IL devices, you may have to wait 2 or 3 seconds after you press **ON** before the prompt turns on (only if the HP-IL interface is installed).

In some cases you may want to control whether the HP-71 is allowed to perform HP-IL operations. The HP-71 provides two statements that allow you to suspend HP-IL activity and to restore HP-IL operation:

- **OFF IO** suspends operation of the loop. Output normally sent to “printer” or “display” devices is sent to the HP-71 display; all other HP-IL activity causes an error message.
- **RESTORE IO** restores HP-IL activity and assigns new addresses to all devices.

Because `RESTORE IO` assigns new addresses to all devices, you can use it after inserting or removing a device in the system—it ensures that all devices have proper addresses. However, this action normally isn't needed if you turn off the HP-71 before changing devices—the HP-71 automatically assigns new addresses whenever you turn it on (except as described next).

Under normal conditions when you turn off the HP-71, it turns off all devices that have that capability. When you turn it on, it turns on such devices and assigns new addresses to all devices. This can reduce power consumption and ensures proper addresses. However, the HP-71 provides an option that bypasses this process by using internal flag `-21`:

- If flag `-21` is clear, the HP-71 turns devices off and on—those that have that capability. If flag `-21` is clear when you turn off the HP-71, it turns off the devices too and remembers that devices may have invalid addresses. When you perform the next I/O operation, the HP-71 turns on the devices and assigns addresses to each device. (Using a “display” device is an I/O operation.)
- If flag `-21` is set, all devices remain turned on when the HP-71 turns off and on. If flag `-21` is set when you turn off the HP-71, it doesn't affect devices. When you turn on the HP-71, it assumes that devices are in the same condition as when the HP-71 was turned off. (If you turned any devices off and on while the HP-71 was off, you should execute `RESTORE IO` to ensure that all devices have valid addresses.)

The normal condition of flag `-21` is clear. Use the `CFLAG` and `SFLAG` statements to clear and set flag `-21` (or any other flag).

Resetting the HP-IL System

Under certain circumstances, the HP-71 may “get stuck” waiting for a response from the HP-IL system when no response is coming. If an operation takes more time than seems reasonable, the best approach is to wait a short while until the HP-71 “gives up” and cancels the operation. (The `STANDBY` statement can change how long the HP-71 waits.) You can then continue using the system—although you may want to check that all devices are connected and turned on.

A less convenient alternative involves interrupting the operation by pressing `[ATTN]` `[ATTN]`. This causes the message `Aborted`. It may then be necessary to execute `RESTORE IO` to activate the HP-IL system.

Under some conditions when you press `[ATTN]` `[ATTN]`, the HP-IL interface may be left in a state that prevents further use of the HP-IL system. In this case, you'll need to execute `RESET HPIL`, then `RESTORE IO` to reset the interface and system.

The following list summarizes an order of remedies that you can use for recovering from a “stuck” condition:

1. First remedy, wait for the HP-71 to cancel the operation by itself, then continue.
2. Second remedy, press `ATTN` `ATTN`, execute `RESTORE IO`, then continue.
3. Third remedy, press `ATTN` `ATTN`, execute `RESET HPIL`, then `RESTORE IO`, then continue.

Two other statements are useful for setting HP-IL devices to known conditions: `CONTROL ON` and `CLEAR`.

The `CONTROL ON` statement, as part of its purpose, ensures that the HP-71 is the controller of the HP-IL system.

The `CLEAR` statement enables you to reset one device or all devices to its known startup conditions. (These conditions may be defined in the device owner's manual as the response to the Device Clear message.)

Specifying HP-IL Devices

For many HP-IL operations that you perform using your HP-71, you'll need to specify the peripheral device that you want to use. The HP-71 provides six ways for you to identify HP-IL devices—six types of *device specifiers*:

- Address.
- Assign code.
- Device word.
- Device ID.
- Accessory ID.
- Volume label.

The first two specifiers—*address* and *assign code*—send information to a known location in the loop. These specifiers provide fastest program execution.

The next three specifiers—*device word*, *device ID*, and *accessory ID*—check each device in the loop until the desired one is found. These specifiers provide convenient ways to identify devices—and programs that use these specifiers aren't noticeably slower than programs that use the methods mentioned above.

The last specifier—*volume label*—is intended primarily for mass storage operations. Because this specifier relates to the medium in the device, the HP-71 must read data from the medium, making this specifier somewhat slower than the other methods—although it ensures that the proper medium is being used.

Each type of device specifier is discussed below. Each type is illustrated by an example that uses the `PRINTER IS` statement—this statement tells the HP-71 where it should send all printed output from `PRINT` and `PLIST` statements. (These statements are described in greater detail in section 2, "Printer and Display Operations.")

If you try the examples shown below, you may get the error message `Device Not Found`. This indicates that your system doesn't contain the device that was specified. You might try changing the device specifier in the example to match the setup of *your* system.

Address. You can specify a device by its physical position in the loop. The HP-71 automatically assigns each device an *address* according to its position in the loop. Addresses are sequential: the first device (connected to the HP-71 OUT receptacle) is given the address #1. The last device (connected to the HP-71 IN receptacle) is given the highest address. (If you have more than 30 devices in the loop or if you have a device with a fixed address, refer to “Using HP-IL Addresses” on page 39.)

Input/Result

```
PRINTER IS :2 [END LINE]
```

Assigns the second device (address 2) as the “printer” device.

```
A=4 [END LINE]
```

Sets variable *A* to a value of 4.

```
PRINTER IS :A [END LINE]
```

Uses the value of variable *A* as the address to assign the “printer” device.

If you don't know a device's address, you can find the address using the `DEVADDR` function.

Assign Code. You can refer to a device by an *assign code*—a one- or two-character code that you assign to each device using the `ASSIGN IO` statement. `ASSIGN IO` causes the HP-71 to associate your assign codes with device addresses. Assignments are made in order: the first assign code is associated with address 1, the second with address 2, and so on.

The first character of an assign code must be a letter. If a second character is used, it can be a letter or digit. Because an assign code may resemble a variable name (which would indicate an address), it's best to enclose an assign code in quotes to remove any ambiguity.

Input/Result

```
ASSIGN IO ":TV,:P1,:MS,:P2"
```

```
[END LINE]
```

Associates codes with the first four devices in the loop.

```
PRINTER IS ":P2" [END LINE]
```

Assigns device with assign code *P2* as the “printer” device.

Assign codes are tied to physical addresses. If you change the order of devices in the loop, you should execute `ASSIGN IO` again to match the assign codes to the new addresses.

Use the LIST IO statement to display a list of all current assign codes.

Input/Result

LIST IO END LINE

Displays all assign codes.

```
4 Device(s) assigned
Device # 1=':TV'
Device # 2=':P1'
Device # 3=':MS'
Device # 4=':P2'
```

Device Word. You can specify a device by describing its “class”—the general type of the device. This is specified by a *device word*. Not all classes of devices have a device word. The table below lists device classes (and subclasses) that are used by the HP-71, the corresponding device words, and the range of accessory ID's used by the HP-71 to define each device class. (Accessory ID is also a type of device specifier, which is discussed later.)

Device Words and Accessory IDs

Class or Subclass	Device Word	Range of Accessory ID
Mass Storage Devices	MASSMEM	16–31
Cassette Drive	TAPE	16
Printers	PRINTER	32–47
Displays	DISPLAY	48–63
Interface Devices	INTRFCE	64–79
HP-IL/GPIO Interface	GPIO	64
Acoustic Coupler	MODEM	65
HP-IL/RS-232 Interface	RS232	66
HP-IL/HP-IB Interface	HPIB	67
Instruments	INSTRMT	80–95
Graphics Devices	GRAPHIC	96–111

Input/Result

PRINTER IS :PRINTER END LINE

Assigns the first printer class device as the “printer” device.

If your HP-IL system contains more than one device of the same class, you can identify each one of those devices by using its *sequence number*. Put its sequence number in parentheses immediately after the device word. (If no sequence number is included, the first device of that class is used.)

Input/Result

```
PRINTER IS :PRINTER(2) END LINE
```

Assigns the second printer class device as the “printer” device.

Refer to the owner’s manual for a device to determine its accessory ID; you can determine its class from its accessory ID. (A device’s accessory ID is defined as its response to the Send Accessory ID message.)

Device ID. You can specify some devices with a *device ID*, which is a string that usually consists of the device’s model number (including the suffix letter). However, not all devices have a device ID.

Input/Result

```
PRINTER IS :HP82905B END LINE
```

Assigns the device with device ID of “HP82905B” as “printer” device.

Use the `DEVID$` function to determine the device ID of a device. `DEVID$` returns a null string if the device has no device ID.

If your HP-IL system contains more than one device with the same device ID, you can identify each one of those devices by using its *sequence number*. Put its sequence number in parentheses immediately after the device ID. (If no sequence number is included, the first device with that device ID is used.)

If you specify fewer characters than contained in a device’s device ID, the HP-71 will look for a match using only that many characters from each device’s device ID.

Refer to the owner’s manual for a device to determine its device ID. (A device’s device ID is defined as its response to the Send Device ID message.)

Accessory ID. You can refer to most HP-IL devices by an *accessory ID*. The accessory ID is a number that identifies both the device and the class to which it belongs. (Refer to the table on page 19.)

To indicate an accessory ID, you must precede the number by `%`. This distinguishes an accessory ID from an address.

Input/Result

```
PRINTER IS %32 END LINE
```

Assigns the device with an accessory ID of 32 as the “printer” device.

Use the `DEVAID` function to determine the accessory ID of a device. A `DEVAID` value of `-1` means that the device has no accessory ID.

If your HP-IL system contains more than one device with the same accessory ID, you can identify each one of those devices by using its *sequence number*. Put its sequence number in parentheses immediately after the accessory ID. (If no sequence number is included, the first device with that accessory ID is used.)

Refer to the owner's manual for a device to determine its accessory ID. (A device's accessory ID is defined as its response to the Send Accessory ID message.)

Volume Label. You can specify a mass storage device by using the *volume label* of the medium installed in the device. Essentially, this enables the HP-71 to access a particular *medium*, rather than a particular type of device. However, not every medium is given a volume label—a volume label is optional. (Refer to “Preparing the Medium” in section 3.)

Note in the following example that a volume label specifier starts with a . (period).

Input/Result

```
CAT .MAIL END LINE
```

Accesses the catalog of directory entries on the medium with volume label MAIL.

The six types of device specifiers can be used in any HP-IL operation (although volume labels will give usable results for mass storage operations only). Refer to “Definitions of Standard Terms” (page 74) in the Keyword Dictionary for information about restrictions on device specifiers.

Printer and Display Operations

This section is for those of you who have printers or display devices in your HP-IL systems. In this section you'll learn how to send output to your printer or display device, and how to duplicate what you see in the HP-71 display.

Assigning the Printer Device

You can assign one device in your HP-IL system to be the designated “printer” device—the device that will be used for all output from the `PRINT` and `PLIST` statements. (If you tried these statements *before* installing the HP-IL interface, you know that the output went to the HP-71 display; this is because no “printer” device was available.)

The `PRINTER IS` statement assigns one device as the “printer” device. This device is called the `PRINTER IS` device.

The `PRINTER IS` statement has three options that direct where your printed output is sent:

- `PRINTER IS device` assigns this device as the `PRINTER IS` device. You can specify the device using any of the first five methods described in section 1, such as an address or a device word.
- `PRINTER IS *` directs all printed output to your “display” device, which can be an HP-IL device or the HP-71 display. (The “display” device is discussed under the next heading.)
- `PRINTER IS NULL` suppresses all printed output—the output is lost.

Many printers can be connected, but only one can be designated by the `PRINTER IS` statement. You can change the `PRINTER IS` device at any time. Simply execute another `PRINTER IS` statement.

When you assign a `PRINTER IS` device, you don't have to designate a printer. You can designate *any* device, such as a video monitor or an acoustic coupler (modem).

Examples: The following statements show some ways you can designate the `PRINTER IS` device.

Input/Result

```
PRINTER IS :PRINTER END LINE
```

Assigns the first printer (device word `PRINTER`) in the system as the `PRINTER IS` device.

```
X=1 END LINE
```

```
PRINTER IS X END LINE
```

Assigns the first device in the system (address defined by `X`) as the `PRINTER IS` device.

```
PRINTER IS NULL END LINE
```

Assigns no device as `PRINTER IS` device—all output is thrown away.

You can also use a printer to get a paper copy of *displayed* output. Refer to the next topic.

Assigning the Display Device

You can assign one device in your HP-IL system to be the “display” device—the device that duplicates the HP-71 display, including all keyboard input and all displayed output. Many HP-71 statements send their output to the “display” device, including `DISP`, `LIST`, and `CAT`.

If you have a standard display device connected in your system, you’ve probably noticed that it’s *already* acting as a “display” device. You didn’t need to assign it. The reason for this is discussed under “Startup Assignments” on page 26.

The `DISPLAY IS` statement assigns one device as the “display” device. This device is called the `DISPLAY IS` device.

The `DISPLAY IS` statement has two options that direct where displayed information is sent:

- `DISPLAY IS device` assigns this device as the `DISPLAY IS` device. You can specify the device using any of the first five methods described in section 1, such as an address or a device word.
- `DISPLAY IS *` or `DISPLAY IS NULL` directs all display information to only the HP-71 display. The HP-71 display is always active.

Many display devices can be connected, but only one can be designated by the `DISPLAY IS` statement. You can change the `DISPLAY IS` device at any time. Simply execute another `DISPLAY IS` statement.

When you assign a `DISPLAY IS` device, you don’t have to designate a display type device. You can designate *any* device, such as a printer (to get a paper copy of keyboard input and displayed output).

Examples: The following statements show some ways you can designate the `DISPLAY IS` device.

Input/Result

```
DISPLAY IS :DISPLAY END LINE
```

Assigns the first display device (device word `DISPLAY`) in the system as the `DISPLAY IS` device.

```
DISPLAY IS * END LINE
```

Assigns no device as the `DISPLAY IS` device—the HP-71 display is used.

```
DISPLAY IS :PRINTER END LINE
```

Assigns the first printer (device word `PRINTER`) as the `DISPLAY IS` device.

DISPLAY IS %48 END LINE

Assigns the first device with accessory ID of 48 (such as HP 82163 Video Interface) as DISPLAY IS device.

The HP-71 recognizes three types of devices as DISPLAY IS devices. Refer to DISPLAY IS in the Keyword Dictionary for detailed information.

Using Printers and Display Devices

After you've assigned a PRINTER IS or DISPLAY IS device, the HP-71 uses that device for all printer or display operations. The following are some of the HP-71 operations that use or affect the PRINTER IS or DISPLAY IS device:

DISP	LIST	WIDTH	CAT
PRINT	PLIST	PWIDTH	CATALL

Refer to the *HP-71 Reference Manual* for detailed information about these operations.

Example: Make a copy of a program listing. Include a heading that describes the program.

Input/Result

PRINTER IS :PRINTER END LINE

Assigns first printer device as PRINTER IS device.

PRINT "Revised Program ",DATE\$ END LINE

Prints a program heading and date.

PRINT END LINE

Prints a blank line.

PLIST INTPROG END LINE

Prints a listing of the program INTPROG.

Example: Print a catalog of the files in your HP-71. Use the CATALL statement to view the catalog of your files, then use the ▼ key to step through the catalog.

Input/Result

DISPLAY IS :PRINTER END LINE

Assigns the first printer as the DISPLAY IS device.

DATE\$ END LINE

Prints the current date.

CATALL END LINE

Prints the catalog entry of the first file in memory.

▼▼▼ ...

Steps through the catalog entries of all other files.

Example: Create key definitions that enable you to scroll the display of an HP 82163 Video Interface. Use these keys to scroll through a program listing.

Input/Result

```
DEF KEY "#162","DISPLAY IS
  DISPLAY@CHR$(27)&'S';@DISPLAY
  IS *": END LINE
```

Defines **[9][A]** as “scroll up.”

```
DEF KEY "#163","DISPLAY IS
  DISPLAY@CHR$(27)&'T';@DISPLAY
  IS *": END LINE
```

Defines **[9][V]** as “scroll down.”

```
DEF KEY "g ", "DISPLAY IS
  DISPLAY@CHR$(27)&'J';":
  END LINE
```

Defines **[9][ERRM]** as “restore display.”

```
f USER
```

Activates User keyboard.

```
g ERRM
```

Assigns first display device as `DISPLAY IS` device.

```
LIST END LINE
```

Lists current file.

```
g [V] (hold)
```

Scrolls display down.

```
g [A] g [A]
```

Scrolls display up.

```
g ERRM
```

Restores cursor.

```
f USER
```

Deactivates User keyboard.

Choosing a Device

You don't have to use a printer for the `PRINTER IS` device, and you don't have to use a video monitor for the `DISPLAY IS` device. You can assign *any* device for these purposes, although it should be a *character-oriented* device. A character-oriented device is a device that uses data as individual characters or lines of characters. Examples of character-oriented devices are:

- Printers.
- Video devices (monitors).
- Acoustic couplers (modems).
- Interface devices that connect to other character-oriented devices.

You'll probably get unpredictable results if you try to use a file-oriented device (such as a mass storage device) as the `PRINTER IS` or `DISPLAY IS` device.

Startup Assignments

Each time you install the HP-IL interface in the HP-71, it *automatically* tries to assign a `PRINTER IS` device and a `DISPLAY IS` device. If these startup assignments are acceptable, you don't have to assign them yourself. You can change or cancel the assignments if you want.

The startup `PRINTER IS` device is the first printer in the system. This is equivalent to executing `PRINTER IS ;PRINTER`. If you don't have a printer in the system at the time you install the HP-IL interface, the HP-71 remembers the assignment. If you add a printer later, it becomes the `PRINTER IS` device (unless you've cancelled the assignment).

Similarly, the HP-71 automatically designates the first display device in the system as the `DISPLAY IS` device. This is equivalent to executing `DISPLAY IS ;DISPLAY`. If you don't have a display device in the system at the time you install the HP-IL interface, the HP-71 remembers this assignment.

The HP-71 always remembers the most recent `PRINTER IS` assignment and `DISPLAY IS` assignment. The assignments are lost if you operate the HP-71 without the HP-IL interface or if you perform a memory reset—then it uses the default `PRINTER IS` and `DISPLAY IS` assignments.

If you have no printer in your system, and if you haven't cancelled the startup `PRINTER IS` assignment, the HP-71 searches for a printer every time you execute a printing operation. It does this before it sends the printed output to the HP-71 display (and `DISPLAY IS` device). This can slow down program execution. You can avoid this by cancelling the startup assignment (using `PRINTER IS *` or `PRINTER IS NULL`) or by suspending I/O activity (using `OFF IO`).

Section 3

Mass Storage Operations

This section is for those of you with “standard” mass storage devices in your HP-IL systems. In this section you’ll learn how to prepare the medium of your mass storage device and how to manipulate files.

The HP-71 defines a “standard” mass storage device as an HP-IL device with an accessory ID of 16. For example, the HP 82161A Digital Cassette Drive is a “standard” mass storage device. (Refer to the owner’s manual of your device to determine its accessory ID—it’s defined as the response to the Send Accessory ID message.) Recall from section 1 that `MASSMEM` corresponds to any accessory ID from 16 through 31, and that `TAPE` corresponds to accessory ID 16. Both of these device words can specify a “standard” mass storage device.*

Preparing the Medium

Before the HP-71 can interact with a mass storage device, the medium in the device must be *initialized*. This means that each medium must be prepared to accept information in a uniform format. The format used by the HP-71 is compatible with the LIF (Logical Interchange Format) standard. This standard is used by other Hewlett-Packard computers also. (Refer to appendix D for detailed information about the LIF standard.)

The `INITIALIZE` statement prepares the medium. It provides two parameters that let you control how the medium is formatted. If you don’t specify a value for a parameter, the HP-71 uses a default value.

- Volume label.
- Number of files in directory.

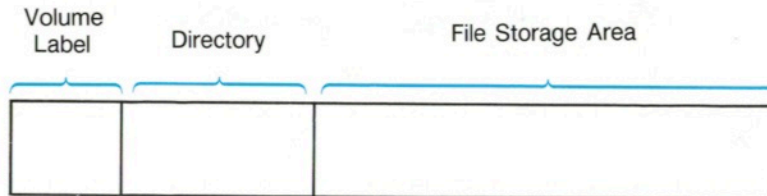
The HP-71 sets up the three sections of the medium:

- Volume label. This defines a string (or “label”) that you can use to identify the medium. (It’s a type of *device specifier* that’s especially useful for mass storage operations.) You don’t need to define a volume label, but you can’t change the volume label unless you initialize the medium again.

USE SUB VOLUME

* If your system contains a mass storage device with an accessory ID other than 16, the HP-IL interface won’t enable the HP-71 to use it for mass storage operations—even though `MASSMEM` can specify such a device.

- **Directory.** This is a list of file names and information about the files. The directory has a fixed size—it can't accommodate more files than you specify when you initialize the medium.
- **File storage area.** This stores the contents of the files.



The size of the directory should be determined by the number of files you expect to put on the medium. If you anticipate having many short files, you will need a larger directory than if you expect to have a few very long files on the medium.

Restrictions for volume labels and file names are given under “Definitions of Standard Terms” in the Keyword Dictionary.

Working With Files

The HP-IL interface builds upon the mass storage operations provided by the HP-71 for main-RAM files—it extends them to operate on files in mass storage devices. The primary difference is that you must specify the device that contains the intended file. To do this, use any of the device specifiers described in section 1 (including the volume label of the medium).

The following statements and functions can be used with files in mass storage devices:

COPY	CAT	CAT\$	PRIVATE
PURGE	RENAME	SECURE	UNSECURE

Each statement and function is described in the Keyword Dictionary.

Example: Copy a program from the HP-71 to a new medium, protect the file, and look at the directory of the medium.

Input/Result

```
INITIALIZE TAPE1:MASSMEM END LINE
```

Initializes medium in first mass storage device and gives it volume label TAPE1.

```
COPY MYFILE TO MYFILE1.TAPE1  
END LINE
```

Copies file MYFILE from main RAM to file MYFILE1 on medium with volume label TAPE1.

```
PRIVATE MYFILE1.TAPE1 END LINE
```

Protects file MYFILE1 so it can't be changed or inspected.

```
SECURE MYFILE1.TAPE1 END LINE
```

```
CAT .TAPE1 END LINE
```

NAME	S	TYPE	LE
MYFILE1	E	BASIC	

Secures file MYFILE1 so it can't be changed or purged.

Displays first directory entry of medium. (After you've stored more files, you can use ▼ and ▲ to step through directory.)

Using Data Files

The HP-71 can perform the same types of data file operations on mass storage devices as it can on main-RAM data files:

- Creating a data file (CREATE).
- Opening a data file (ASSIGN #).
- Writing to a data file (PRINT #), but not enlarging it.
- Reading from a data file (READ #).
- Restoring the data file pointer to a specific record (RESTORE #).
- Closing a data file (ASSIGN #).

These operations are extensions of the main-RAM operations provided by the HP-71. CREATE and ASSIGN # are described below and in the Keyword Dictionary. Other operations are identical to those described in the *HP-71 Reference Manual*.

Creating a Data File

The CREATE statement creates a new data file. You can supply up to four parameters that define the file that's created:

- File type. This defines the type of data file (DATA, SDATA, or TEXT).
- File name and device. This defines the name and location of the new file.
- File size. This defines the space on the medium that's allocated for the file. Unlike a data file in main RAM, a mass storage data file *can't be enlarged* after it's created.
- Record length. This defines the size of each record on the medium. This parameter is optional.

Each type of data file requires that the file size and record length be specified in certain units. Refer to the Keyword Dictionary.

The CREATE statement does not open the file. You must execute an ASSIGN # statement to access the file.

Accessing a Data File

To use a data file, you must first use `ASSIGN #` to assign an *I/O channel* to the file. An I/O channel includes a buffer in main RAM that's used to transfer data to and from the file.

A mass storage data file is accessed much like a main-RAM data file. The HP-71 automatically interacts with the mass storage device as needed to keep the file up-to-date with information stored in the buffer. It does this by transferring data between the buffer and the mass storage file as required.

To close a file, execute `ASSIGN # ... TO *`. (Other situations that cause a file to close are listed under `ASSIGN #` in the Keyword Dictionary.)

Example: Create a data file on a mass storage device, then use it to store and recall data.

Input/Result

```
CREATE DATA DATA1:TAPE,1 END LINE
```

Creates file `DATA1` on first mass storage device. File has a size of one record.

```
ASSIGN #1 TO DATA1:TAPE END LINE
```

Assigns I/O channel `#1` to file `DATA1` and opens the file.

```
PRINT #1; 1,2,3,4,5,6,7,0,  
"OCTAL" END LINE
```

Sends data to I/O channel (file `DATA1`).

```
RESTORE #1 END LINE
```

Sets file pointer to beginning of file.

```
DIM N(7) END LINE
```

Declares eight elements in array `N` (assuming `OPTION BASE 0`).

```
READ #1;N(),N$ END LINE
```

Reads data into array `N` and string variable `N$`.

```
DISP N(7);"-";N(6);N$ END LINE
```

Displays three variables.

0 - 7 OCTAL

```
ASSIGN #1 TO * END LINE
```

Closes the file.

Packing the Medium

After you've created and purged several files on a medium, the medium may have unused gaps between files. If these gaps significantly reduce the capacity of the medium, you may want to convert the gaps into usable free space.

Two statements are useful when you must gain additional free space on the medium:

- **PACKDIR** eliminates all unused gaps in the directory only. This is useful when the directory is full, but the file storage space has free space at the end of the medium.
- **PACK** eliminates all unused space in both the directory and the file storage area. This gives you the maximum amount of usable space. It also takes more time to perform than **PACKDIR** and causes greater wear on the medium.

Note: These operations cause each record to be read and re-stored on the medium. This requires that the medium be positioned to many different locations, possibly resulting in a reduction in its usable life, especially for tape cassettes. It is recommended that you avoid the casual or unnecessary use of **PACK** and **PACKDIR**.

General I/O Operations

Introduction

This section is for those of you who want to do more than perform standard operations using display, printer, or mass storage devices. Sections 2 and 3 cover the fundamental operations that involve these devices. This section gives an overview of other types of I/O (input/output) operations—operations that are somewhat more advanced, and also more versatile.

The operations described in this section are *not* intended for a particular type of HP-IL peripheral device—they are general-purpose operations that have a wide range of application. For this reason, the descriptions are less specific than those in the previous sections. In addition, you may have greater contact with HP-IL concepts since many of the general I/O capabilities directly relate to fundamental HP-IL operations.

For example, you can easily perform simple input and output operations using general I/O statements. The `ENTER` and `OUTPUT` statements provide convenient interaction with *any* HP-IL device. Using these statements, the HP-71 can fetch and send data using any specified device. These capabilities complement the display, printer, and mass storage capabilities described in sections 2 and 3.

In addition, the general I/O capabilities enable you to perform sophisticated interaction with HP-IL devices. At the extreme end of the spectrum, the `SEND` statement causes the HP-71 to send individual HP-IL instructions (messages) on HP-IL. However, most users probably won't use `SEND` for everyday control of their HP-IL systems—other general I/O statements can accomplish most of the required tasks without requiring complicated procedures.

For most applications, the HP-71 is the HP-IL *controller*—it controls the HP-IL system. It tells the other HP-IL components what to do and when to do it. But the HP-IL interface gives the HP-71 the ability to respond as an HP-IL *device*—to be controlled by another component in the HP-IL system, rather than to be the controlling component. This enables the HP-71 to interact with another computer, possibly one that must always be a controller.

The material in this section is divided into four parts:

- “Operation As a Controller.” This describes how you can use general I/O capabilities while the HP-71 is the HP-IL controller. (This is the situation that will probably be most common.)
- “Passing Control of HP-IL.” This describes how the HP-71 can give up control of the HP-IL system and how the HP-71 can acquire control of the HP-IL system. (You won’t need to use this portion if you’ll use the HP-71 only as a controller.)
- “Operation As a Device.” This describes how you can use general I/O capabilities while the HP-71 is an HP-IL device. (You won’t need to use this portion if you’ll use the HP-71 only as a controller.)
- “Using Binary Functions.” This describes five binary operations that are useful for working with numbers that you want to interpret as binary numbers. (You can use these functions while the HP-71 is operating as a controller or as a device.)

You can see from the organization that you may need to refer to only the first portion of this section—if the HP-71 will be the only computer connected to your HP-IL system. The fourth portion may be useful also.

On the other hand, you may need to refer to all four portions of this section—if you intend to use the HP-71 while another component controls the HP-IL system.

Within each portion, the topics are generally arranged in order of increasing complexity. You may want to read only as far as necessary to learn about the operations that you’ll need for your applications.

Refer to the Keyword Dictionary (part II) for detailed information about individual keywords that seem appropriate for your needs.

Operating As a Controller

The following topics give an overview of the general I/O operations that the HP-71 can perform while it’s the HP-IL controller.

The HP-71 normally operates as the HP-IL controller. Whenever you install the HP-IL interface or execute `RESET HPIL`, the HP-71 begins operating as a controller. (Refer to “Passing Control of HP-IL” on page 51 for information about giving up control of the HP-IL system.)

As controller, the HP-71 determines what operations other HP-IL devices are to perform.

Sending Data

Four statements enable you to send information from the HP-71 to an HP-IL device:

- `OUTPUT`. Accesses any device specified in the statement. Sends numeric and string data.
- `PRINT`. Accesses only the `PRINTER IS` device, which must be defined by a separate statement. Sends numeric and string data. (`PRINT` is described in the *HP-71 Reference Manual*.)

- **PRINT #.** Accesses any mass storage data file opened by an **ASSIGN #** statement. Sends numeric and string data. (**PRINT #** is mentioned in section 3 and is described in the *HP-71 Reference Manual*.)
- **COPY.** Accesses any device specified in the statement. Sends complete files. (**COPY** is mentioned in section 3 and is discussed in the Keyword Dictionary.)

The **OUTPUT** statement is the primary statement for sending numeric and string data from the HP-71 to *any* character-oriented HP-IL device—a device that processes data (for example, displays, prints, transmits, or interprets data).

The **OUTPUT** statement enables you to specify any HP-IL device as the destination for the transfer. It lets you specify the output data as a sequence of numeric and string variables and expressions.

Another feature of the **OUTPUT** statement is the **USING** option, which enables you to define how the data is formatted when it's sent. The format is specified using the same symbols as the HP-71 **IMAGE** statement.

The **PRINT** statement works much like the **OUTPUT** statement, although there are several differences:

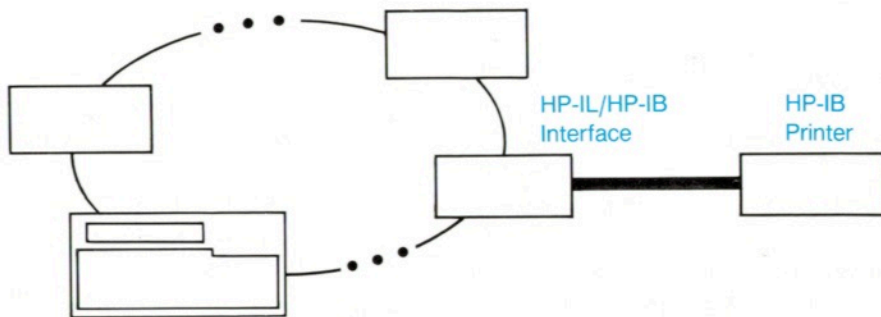
- **OUTPUT** and **PRINT** can both send a sequence of numeric and string data, which is often specified by a list of variable names. The data is transferred as a sequence of characters.
- **OUTPUT** has no restriction on the length of an output sequence. **PRINT** sends data that's formatted according to the line length specified by **PWIDTH**.
- **OUTPUT** and **PRINT** can both use the **USING** option for formatting the data that is sent.
- **OUTPUT** and **PRINT** both send an end-of-line sequence (defined by the HP-71 **ENDLINE** statement) at the end of the transfer; both statements can suppress the sequence by using an output format or a terminating semicolon or comma.
- **OUTPUT** defines the destination of the transfer right in the statement. The destination of the **PRINT** transfer must be defined separately by the **PRINTER IS** statement, and often remains unchanged for an individual application.
- **OUTPUT** and **PRINTER IS** can both use the **LOOP** option, for which the destination device(s) aren't set up by the **OUTPUT** or **PRINT** statement—they can be set up by another statement.

The **PRINT #** statement is a mass storage statement. It sends numeric and string data from the HP-71 to a data file in a file-oriented HP-IL device—a device that merely stores and retrieves encoded information on a medium, but doesn't interpret the data. The data file must be opened by an earlier **ASSIGN #** statement. Because the data is encoded by the HP-71, no special formatting is used.

The **COPY** statement is a mass storage statement. Among its capabilities is the ability to send complete file information from the HP-71 to an HP-IL device. The file information includes directory information and the actual file contents. **COPY** can also use the **LOOP** option for sending a file to a prearranged device.

The preceding discussion shows that `OUTPUT` and `PRINT` have many similarities, although the `OUTPUT` statement is more versatile for general-purpose output operations. The `PRINT #` statement is intended for storing data in a data file so that it can be retrieved later. The `COPY` statement is designed to send complete file information.

Example: Consider an HP-IL system that includes an HP 82169A HP-IL/HP-IB Interface that connects to an HP-IB printer. The printer's HP-IB address switch is set to 11. The following steps set up the HP-IL/HP-IB interface for using its "general addressing" scheme, then use the printer to print data.



Input/Result

```
OUTPUT :INTRFCE;"I;E6;A11"
```

`END LINE`

Sends data to the HP-IL/HP-IB interface, causing it to initialize itself, enable general addressing, and place 11 in its address table.

```
RESTORE IO END LINE
```

Reassigns addresses to all HP-IL devices. (The HP-IL/HP-IB interface interrupts normal addressing before it is set to use general addressing.)

```
PRINTER IS :11 END LINE
```

Sets the HP-IB printer as the `PRINTER IS` device.

```
PRINT "This is the printer."
```

`END LINE`

Prints using the HP-IB printer. This and all subsequent printer output goes to this printer.

Fetching Data

Three statements enable the HP-71 to fetch data from an HP-IL device:

- **ENTER**. Accesses any device specified in the statement. Fetches numeric and string data.
- **READ #**. Accesses any data file opened by an **ASSIGN #** statement. Fetches numeric and string data. (**READ #** is mentioned in section 3 and is described in the *HP-71 Reference Manual*.)
- **COPY**. Accesses any device specified in the statement. Fetches complete file information. (**COPY** is mentioned in section 3 and is discussed in the Keyword Dictionary.)

The **ENTER** statement is the primary statement for fetching numeric and string data to the HP-71 from a character-oriented HP-IL device. The **ENTER** statement enables you to specify any HP-IL device as the source for the transfer. It lets you specify a sequence of variables in which the data is stored.

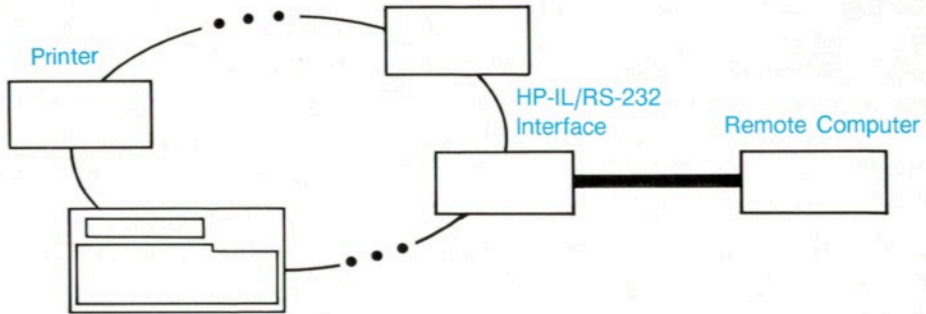
Like the **OUTPUT** statement, the **ENTER** statement provides the **USING** option. This enables you to define how the data is interpreted when it's received. The format is specified using symbols that are compatible with the conventions of the HP-71 **IMAGE** statement and the **OUTPUT** statement.

The **READ #** statement is a mass storage statement. It fetches numeric and string data to the HP-71 from a data file in a file-oriented HP-IL device. The data file must be opened by an earlier **ASSIGN #** statement. Because data is encoded on the medium, no special formatting is used when fetching the data.

The **COPY** statement is a mass storage statement. Among its capabilities is the ability to fetch file information from an HP-IL device. The file information includes directory information and the actual file contents. **COPY** can also use the **LOOP** option for fetching a file from a prearranged device.

The preceding discussion shows that **ENTER** is useful for general-purpose input operation. The **READ #** statement is intended for retrieving data stored in a data file. The **COPY** statement is designed to fetch complete file information.

Example: Consider an HP-IL system that includes an HP 82905B Printer and an HP 82164A HP-IL/RS-232-C Interface connected to a remote computer. The following program fetches data from the HP-IL/RS-232 interface (which it receives from the remote computer) and prints it on the printer.



```

10 DIM A$(80)
20 PRINTER IS %33
  :
100 ENTER :INTRFCE;A$
110 PRINT A$
120 GOTO 100

```

Dimensions A\$ to 80 characters.
Selects printer using its accessory ID.

Fetches line of characters from HP-IL/RS-232 interface and stores data in A\$.

Sends line of data to printer.

Branches for next line.

Using HP-IL Addresses

As described in section 1, the HP-IL interface provides six ways for you to specify HP-IL devices. For four of these methods, you can use a fixed device specifier that can work properly, no matter where the intended device is installed in the loop. For example, :DISPLAY always denotes the first display device in the loop, regardless of its actual position.

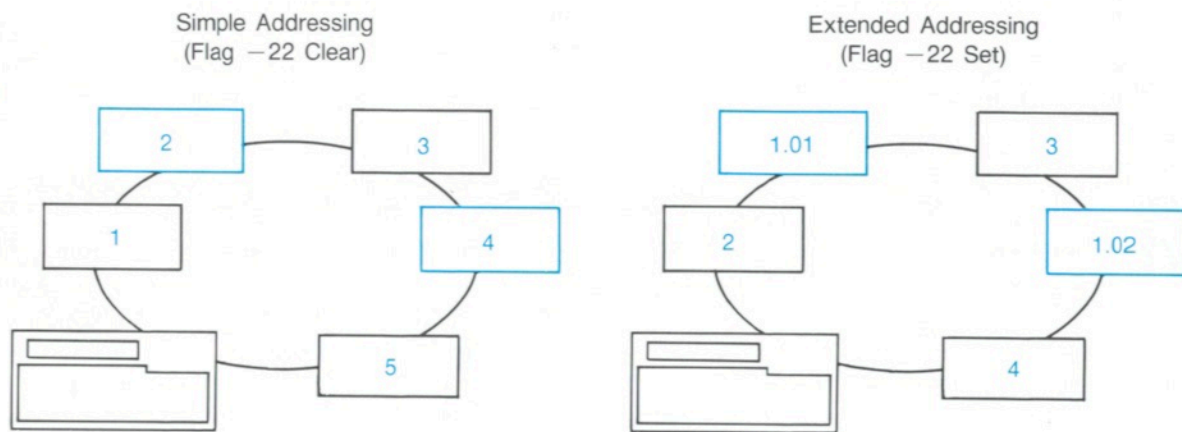
If you use HP-IL addresses to specify devices, you minimize the time required to perform I/O operations—it eliminates the need for the HP-71 to search for the intended devices. However, this method requires either that you know the addresses of certain devices or that you will determine the addresses of those devices.

The HP-71 provides two different schemes for establishing addresses. The scheme it uses is specified by the state of flag -22.

- Simple addressing (flag -22 clear). This provides convenient addressing for no more than 30 HP-IL devices. Each device receives a simple address, which is an integer. All HP-IL devices are capable of taking a simple address. (This is sometimes called “auto addressing.”)

- Extended addressing (flag -22 set). This is intended for systems containing more than 30 devices (although it can be used with fewer devices) or containing a device with a fixed address. Each device that is capable of taking an extended address receives an extended address, which consists of a primary part and a secondary part. (The primary and secondary parts are each integers—the HP-71 uses them in the form *pp.ss.*) The remaining devices, which can't take extended addresses, are assigned simple addresses. Up to 930 devices can be accommodated with extended addressing (addresses 1.01 through 30.31).

For example, the HP-IL system illustrated below has two devices that can take extended addresses (they're outlined in blue). The two diagrams show how the HP-71 assigns addresses for each of the addressing schemes.



If you use `ASSIGN IO` to establish assign codes for HP-IL devices, the HP-71 automatically uses simple addressing—the setting of flag -22 is ignored. Simple addressing is used until all assign codes are cancelled by `ASSIGN IO *`—then flag -22 determines the addressing scheme.

For many applications, the address of an individual device won't be known—especially by a program that is written for a variety of system setups. The `DEVADDR` function provides a way to get and remember the address of a device. You can use any legal device specifier and obtain that device's address. For example, `A=DEVADDR("MASSMEM")` sets variable `A` to the address of the first mass storage device—you can use `A` to indicate that device for all subsequent operations involving that device.

Example: Check the operation of simple and extended addressing using an HP 82164A HP-IL/RS-232-C Interface connected as the first HP-IL device. (This HP-IL device can take an extended address.)

Input/Result

ASSIGN IO * END LINE

Cancels assign codes.

SFLAG -22 END LINE

Specifies extended addressing.

RESTORE IO END LINE

Assigns new addresses.

DEVADDR("INTRFC") END LINE

Returns address of HP-IL/RS-232 interface, an extended address.

1.01

CFLAG -22 END LINE

Specifies simple addressing.

RESTORE IO END LINE

Assigns new addresses.

DEVADDR("INTRFC") END LINE

Returns address of HP-IL/RS-232 interface, a simple address.

1

Three statements *always* cause the HP-71 to assign new addresses to HP-IL devices: `ASSIGN IO`, `RESTORE IO`, and `CONTROL ON`. The HP-71 uses flag -24 to control whether the HP-71 assigns new addresses at other times:

- If flag -24 is clear, the HP-71 assigns new addresses at the next I/O operation after device addresses have been cancelled (such as by turning off the HP-71 with flag -21 clear, which turns off devices, or by cancelling addresses using certain `SEND` statements).
- If flag -24 is set, the HP-71 doesn't assign new addresses (except in response to the statements listed above).

The normal condition of flag -24 is clear, which provides normal addresses at all times. You can set flag -24 if you want to control the addressing of devices in a special situation.

Controlling the Loop

As controller of the HP-IL system, the HP-71 can do more than send and receive data. It can direct HP-IL devices to perform certain internal tasks—that is, tasks that affect the devices' internal conditions. The responses by individual devices may differ, since each device is designed to perform a certain type of operation. For example, an HP-IL test instrument may be designed to measure electrical signals, while an interface-type device may be designed to translate data from one I/O format to another.

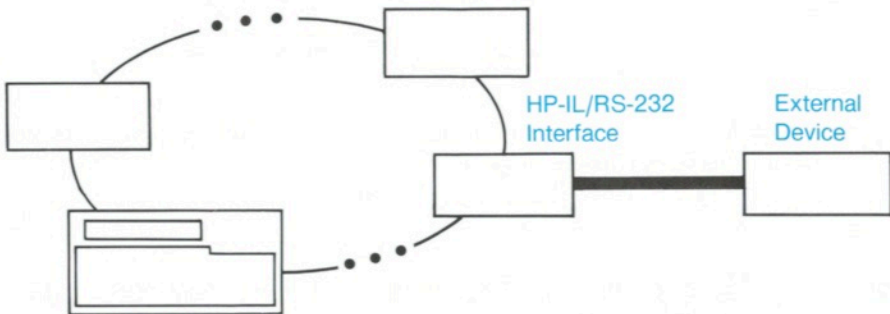
The table below lists the statements that enable the HP-71 to control HP-IL devices. They enable you to set internal conditions for devices.

Statements for Controlling HP-IL Devices

Statement	Typical Response
CLEAR	Clears an individual HP-IL device or all HP-IL devices. Typically, a device reverts to its startup conditions.
LOCAL	Sets an individual HP-IL device or all HP-IL devices to Local mode. Typically, a device in this mode can respond to its "local" controls, such as keys or switches.
LOCAL LOCKOUT	Sets all HP-IL devices to a "local lockout" condition. Typically, a device in this condition doesn't respond to its "local" controls, such as keys or switches.
REMOTE	Sets an individual HP-IL device or all HP-IL devices to Remote mode. Typically, a device in this mode can respond to instructions received from a "remote" source (another device connected on HP-IL).
TRIGGER	Causes an action by an individual HP-IL device or by all HP-IL devices. The action taken by a device depends entirely on the device. For example, a voltmeter may make a voltage measurement.

Remember, the responses to these statements by any HP-IL device depend entirely upon the device. Many devices have no response for some of these operations.

Example: Consider an HP-IL system that may contain an HP 82164A HP-IL/RS-232-C Interface that's connected to an external RS-232 device. The following program lines search for the HP-IL/RS-232 interface, set it for certain RS-232 conditions, and then send data to it (and the external device).



```
10 D=DEVADDR("HP82164A")
```

```
20 IF D=-1 THEN DISP "RS-232 not found"
   @ STOP
```

```
30 REMOTE D
```

```
40 OUTPUT D;"R0;R1;P1;SS0;SW0;SB6"
```

```
50 LOCAL
```

```
:
```

```
130 OUTPUT D;A$
```

Sets `D` to the address of the HP-IL/RS-232 interface.

Displays message if device not found.

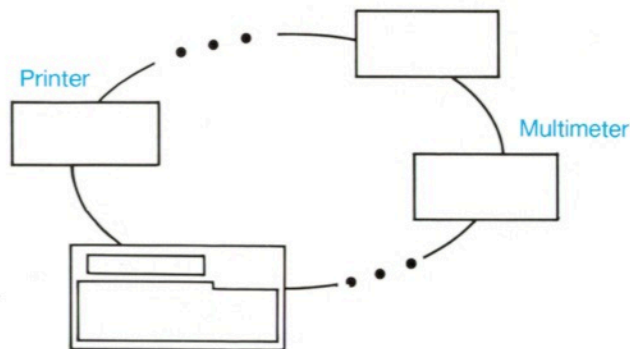
Sets device to Remote mode (so that RS-232 conditions can be set).

Sends data to HP-IL/RS-232 interface, setting its RS-232 conditions (clear buffers, odd parity, 1 stop bit, 8 bits per word, and 300 baud).

Sets all HP-IL devices to Local mode (so that HP-IL/RS-232 interface can transfer data to external device).

Sends data from `A$` to HP-IL/RS-232 interface (and external device).

Example: Consider an HP-IL system containing an HP 3468A Multimeter and an HP 82905B Printer. The HP-71 can use its internal timer to trigger periodic voltage measurements and to record them on the printer.



```
10 DIM V$(15)
```

```
20 PRINTER IS %33
```

```
30 D=DEVADDR("HP3468A")
```

```
40 REMOTE D
```

Dimensions voltage-reading string.

Sets `PRINTER IS` device using accessory ID.

Sets `D` to address of multimeter.

Sets multimeter to Remote mode (so that it can receive instructions from HP-71).


```
50 OUTPUT D;"F1T2N4RAZ1"
```

Sends data to multimeter, setting its operating conditions (dc volts, single triggering, four-digit display, autoranging, and autozero on).

```
60 LOCAL
```

Sets all devices to Local mode.

```
70 ON TIMER #1,10 GOTO 80
```

Starts timer for 10-second sampling.

```
80 TRIGGER D
```

Triggers multimeter to take reading.

```
90 ENTER D;V$
```

Fetches voltage measurement as string.

```
100 PRINT USING "8A,5X,K";TIME$,V$
```

Prints time and voltage reading on printer (using format).

```
110 GOTO 110
```

Waits for timer interrupt.

Fetching Device Information

Most HP-IL devices are able to provide useful information about themselves. This information can be separated into two categories:

- Identifying information: information that helps identify a device.
- Operating information: information that conveys a device's current operating condition.

The `DEVAID` function provides identifying information for a device. It returns the *accessory ID* of the specified device. The accessory ID is a number that is unique to one model of HP-IL device (or possibly a group of devices that are functionally similar). In addition, accessory ID's are grouped into intervals that correspond to nine classes of HP-IL devices.

For example, accessory ID's from 32 through 47 are reserved for printers; the HP 82905B Printer has an accessory ID of 33. If such a printer were located at address 3, the statement `X=DEVAID(3)` would set variable `X` to the value 33.

The `DEVID$` function provides identifying information for a device. It returns the *device ID* of the specified device. The device ID is a string that is unique to one model of HP-IL device—it usually consists of the device's model number.

For example, if an HP 82905B Printer were located at address 3, the statement `A$=DEVID$(3)` would set string variable `A$` to "HP82905B".

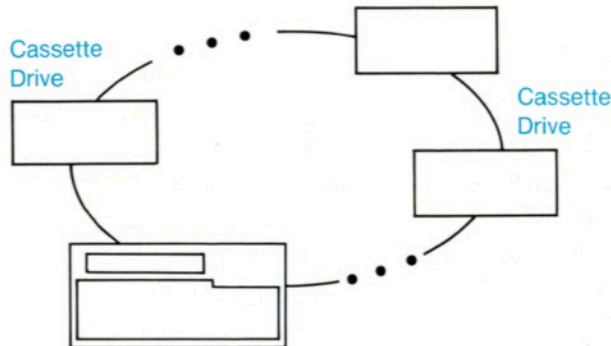
The `SPOLL` function provides status information from a device. A device's status conveys its current condition; the actual meaning of the value returned by `SPOLL` depends upon the device. Some devices send one byte of status information according to a standard set of meanings—a "system" status byte. Other devices don't use the "system" status convention, but rather indicate status according to their own needs—sometimes using several bytes of status. The `SPOLL` function returns a single numeric value that reflects up to four bytes of status sent by a device.

Example: The following program segment assigns the `PRINTER IS` device and sets its print width. The program assumes that a printer with an accessory ID of 32 has a print width of 24 characters (such as the HP 82162A Thermal Printer); otherwise, a printer has a width of 80 characters.

```
10 A=DEVAID("PRINTER")
20 IF A=32 THEN W=24 ELSE W=80
30 DISP "Print width is";W
40 PRINTER IS %A
50 PWIDTH W
```

Sets `A` to accessory ID of first printer.
If `A` is 32, sets `W` to 24; otherwise, sets `W` to 80.
Displays print width.
Assigns `PRINTER IS` device.
Sets print width to `W`.

Example: Consider an HP-IL system that may contain several HP 82161A Digital Cassette Drives. The following program segment checks that each drive is ready and has a tape installed.



```
10 FOR A=1 TO 30
20 I=DEVAID(A)
30 IF I#16 THEN 100

40 S=SPOLL(A)
50 IF S>=32 THEN 40

60 IF S#20 THEN 100

70 BEEP @ DISP "No tape: address";A
80 PAUSE
```

Sets loop to check all addresses.
Sets `I` to accessory ID of device at address `A`.
Branches to next address if accessory ID isn't 16 (isn't cassette drive).
Sets `S` to status from cassette drive.
Gets status again if `S` is greater than or equal to 32 (cassette drive is performing an operation).
Branches to next address if `S` isn't equal to 20 (status equals 20 if tape not installed).
Gives message for tape not installed.
Pauses execution awaiting installation of tape.

```

90 GOTO 40
100 NEXT A
110 DISP "All drives ready"

```

When continued, gets status again to verify tape installation.

Loops back for next address.

Gives message for all tapes installed.

Setting the HP-IL Timeout

The HP-IL interface concept (described briefly in section 1) is based on the sequential transfer of information around the loop. In general, when the controller of the system sends an HP-IL instruction, it doesn't send another instruction until the first one travels completely around the loop and returns to the controller. This ensures that each device has completed that step before it receives the next one. But this same process can cause the controller to be uncertain about whether an instruction is making its way around the loop—if the controller hasn't received a response yet, it isn't sure all devices are working. What would happen if a device were to malfunction or be turned off?

As controller of the HP-IL system, the HP-71 is able to verify that the system is responding to HP-IL messages that it sends around the loop. The HP-71 has two ways of checking the HP-IL system's response, and it maintains two parameters that define how it checks the response:

- By checking that messages return to the HP-71 within a reasonable period. The *timeout period* defines how long the HP-71 waits before it declares a "timeout error" (and cancels the operation).
- By checking the continuity of the loop at regular intervals. (The controller can do this using a special HP-IL message that is passed immediately by every device.) The *verify interval* defines the length of time between continuity checks; if the continuity check fails, the HP-71 declares a "loop broken" error (and cancels the operation).

When the HP-IL interface is first installed, the HP-71 sets the timeout period to 60 seconds and sets the verify interval to 2 seconds. This means that if any *single* step in an operation takes longer than 2 seconds, the HP-71 will begin checking loop continuity every 2 seconds. If the continuity remains good, the operation continues for up to 60 seconds. If instead the continuity goes bad, the operation is cancelled immediately. If the operation isn't completed within 60 seconds, it is cancelled. (Most HP-IL interactions take much less time than 60 seconds; however, there could be some interactions that take longer than 60 seconds.)

The `STANDBY` statement enables you to define how the HP-71 checks the HP-IL system. Using this statement, you can change the timeout period and the verify interval. Three options are provided:

- `STANDBY ON` effectively sets both parameters to infinity (the HP-71 waits forever for an operation to be completed, and it never checks continuity).
- `STANDBY` with one or two parameters defines both the timeout period and the verify interval.
- `STANDBY OFF` resets the parameters to their starting values (60-second timeout period and 2-second verify interval).

Using HP-IL Interrupts

The HP-71 (without the HP-IL interface) has the capability to recognize and respond to certain “interrupt” events. The HP-71 `ON TIMER` and `ON ERROR` statements cause branching after completing the statement during which a timer interval expires or an error occurs.

The HP-IL interface provides an interrupt capability based on HP-IL events. HP-IL interrupts cause *end-of-line* branching. (If a computer checks for interrupt events only at the end of each program line, the interrupt branching is called “end-of-line” branching.)

The HP-71 can recognize and respond to any of seven HP-IL events as interrupt events (while it is a controller):

- An HP-IL device has sent a *service request*, indicating that it wants the attention of the HP-71 for some reason.
- The HP-71 has received an HP-IL message from another device telling the HP-71 to clear its status as controller, talker, and listener.
- The HP-71 has received an HP-IL message normally intended to reset a device’s internal conditions.
- The HP-71 has received an HP-IL message normally intended to “trigger” some sort of action.
- The HP-71 has received an HP-IL message that normally conveys a “device-dependent” instruction.
- The HP-71 is set to receive data.
- The HP-71 is set to send data.

By using interrupt branching, an HP-71 program can carry out its normal processing, but also be able to monitor continuously for special events. If a special event occurs, the program can branch to a routine that takes any action that’s required. As an alternative, an interrupt event can turn on the HP-71 if it was turned off using `BYE` in a program.

Three statements determine how the HP-71 responds to HP-IL interrupt events:

- The `ENABLE INTR` statement defines which HP-IL events can cause interrupt branching—which events are *enabled*.
- The `ON INTR` statement defines how a program branches when an enabled event does occur.
- The `OFF INTR` statement prevents a program from branching—even if an enabled event occurs.

The HP-71 also provides three functions that enable you to determine what conditions exist when an interrupt occurs:

- The `READINTR` function returns the value of the *interrupt-cause byte*, which indicates the interrupt events that have occurred. (It also resets this byte.)

- The `STATUS` function returns the value of the *status byte*, which indicates the HP-IL condition that exists at the time it is read.
- The `SPOLL` function returns the value of any device's status information, which may indicate whether that device caused the interrupt and, if so, why.

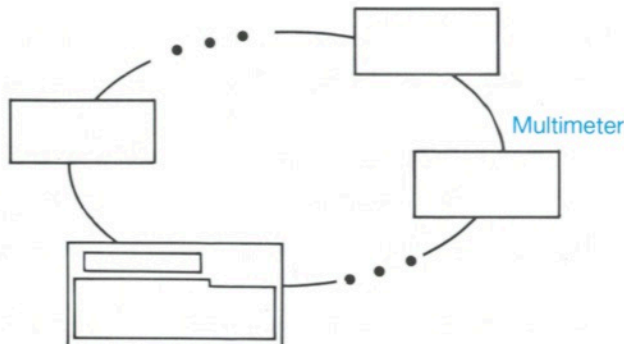
Each of these functions is useful for deciding what action to take when an interrupt occurs. `READINTR` enables you to take action based on the interrupt event that occurred. `STATUS` enables you to take action based on the HP-IL condition that exists at the moment. `SPOLL` enables you to take action based on the needs of the device that caused the interrupt.

Service requests are an important part of device-to-controller communication. They enable a device to notify the controller that the device requires attention. The HP-71 can recognize a service request as an HP-IL interrupt event.

A device indicates a service request by “flagging” an HP-IL message that’s passing around the loop. Only certain messages can be flagged with a service request: those that convey data (Data Byte and End Byte messages) and those used only for service request purposes (Identify messages). Any device may modify these messages to indicate a service request to the controller.

The HP-71 notices service requests—but you can program it either to respond to service requests or to ignore service requests, according to your application. If you ignore service requests in your application, then your program may need to periodically check the status of devices that may require the attention of the HP-71.

Example: Consider an HP-IL system containing an HP 3468A Multimeter. The following program segment uses HP-IL interrupts caused by the multimeter to indicate that a measurement is ready to be fetched by the HP-71. Each measurement is triggered by manually pressing the SGL TRIG (single trigger) key on the multimeter. Up to 10 measurements are stored.



```

10 DIM V(10)
20 A=DEVADDR("HP3468A")
30 IF A=-1 THEN DISP "Meter not found" @
  STOP
40 REMOTE A

50 OUTPUT A;"M01"

60 LOCAL
70 X=0
80 ENABLE INTR 8

90 ON INTR GOTO 1000
100 IF X=10 THEN STOP
110 SEND IDY @ GOTO 110

  :
1000 I=READINTR

1010 IF BIT(I,3)=0 THEN 1060

1020 S=SPOLL(A)
1030 IF BIT(S,0)#0 THEN 1060

1040 X=X+1
1050 ENTER A;V(X)

1060 ENABLE INTR 8 @ GOTO 100

```

Dimensions V for 10 readings.

Sets A to address of multimeter.

Gives message if multimeter address not found.

Sets multimeter to Remote mode (so that it can receive instructions from HP-71).

Sends data to multimeter, setting it to send a service request when reading taken—SGL TRIG pressed.

Sets HP-IL devices back to Local mode.

Sets counter X to 0.

Enables interrupt to occur for service requests only (value 8).

Defines branching when interrupts occur.

Stops execution if 10 readings taken.

Repeatedly sends HP-IL message that can be modified to indicate a service request.

Sets I to value of interrupt-cause byte and resets the byte.

Bypasses subroutine if interrupt isn't caused by service request (indicated by bit 3 of I).

Sets S to status of multimeter.

Bypasses subroutine if multimeter doesn't have reading available (indicated by bit 0 of S).

Increments counter.

Fetches reading from multimeter and stores it in $V(X)$.

Enables interrupts as before and branches to main routine.

Line 110 could actually be replaced by any program lines, as long as the HP-71 periodically sends data on HP-IL (or sends any other HP-IL messages that can indicate a service request). This allows the multimeter to indicate a service request to the HP-71.

Certain devices are able to send an "asynchronous" service request—by which messages are initiated by the device on an idle loop. (This differs from a regular service request, in which a device "flags" messages passing around an active loop.) The purpose of the asynchronous request condition is to conserve power—if a service request will be the only HP-IL activity, the controller and all devices don't have to

continually pass messages around the loop. An asynchronous request condition is set up by the controller (using an Enable Asynchronous Requests message, which can be sent using `SEND CMD 24`). The only message that doesn't cancel this condition is the Loop Power Down message, which sets devices to their low-power states (those that have this capability).

Sending HP-IL Messages

Each time you perform an HP-IL operation while the HP-71 is the *controller* of the HP-IL system, the HP-71 actually sends and receives a sequence of *HP-IL messages*. These messages are used in a way that follows a set of strict guidelines—often called *HP-IL protocol*.

For most situations, the statements and functions described elsewhere in sections 1 through 4 will provide convenient ways to perform most HP-IL operations—you won't have to concern yourself with sending HP-IL messages according to HP-IL protocol. However, in certain situations it may be necessary to exercise more direct control of the interaction with the HP-IL system.

The `SEND` statement enables the HP-71 to send almost any sequence of HP-IL messages. If you have a situation that requires a special sequence of HP-IL messages, you can use the `SEND` statement. If you have a situation in which speed is critical, you can use the `SEND` statement, avoiding some of the built-in “housekeeping” messages used by other HP-IL operations.

You can use `SEND` successfully *only if you observe HP-IL protocol*. A full discussion of HP-IL protocol is beyond the scope of this manual.* However, the next few paragraphs illustrate HP-IL protocol using some typical sequences of HP-IL messages used by standard HP-IL operations.

When you turn off the HP-71, it sends this sequence of HP-IL messages (if flag -21 is clear):

```
No Operation
Loop Power Down
```

When you perform the first HP-IL operation after turning on the HP-71, it uses the following sequence of messages to initialize the HP-IL system before performing the operation (if flag -21 was clear at turn-off):

```
No Operation
Auto Address Unconfigure
Auto Extended Secondary  } if flag -22 is set
Auto Extended Primary
Auto Address
```

* Two references for learning about HP-IL protocol are

- Kane, Gerry, et al. *The HP-IL System: An Introductory Guide to the Hewlett-Packard Interface Loop*. Osborne/McGraw-Hill, Berkeley, California, ©1982.
- Hewlett-Packard Company. *The HP-IL Interface Specification*. HP part number 82166-90017, ©1982.

Similarly, the `RESTORE IO` statement restores HP-IL communication by initializing the HP-IL system. To do this, the HP-71 uses the following sequence of messages:

```
Interface Clear
Auto Address Unconfigure
Auto Extended Secondary } if flag -22 is set
Auto Extended Primary
Auto Address
```

When using the `OUTPUT` statement (without the `LOOP` option) to send data to a device, the HP-71 uses this sequence of messages:

```
Unlisten
Listen Address
Untalk
Data Byte
:
Data Byte
Untalk
Unlisten
```

When using the `ENTER` statement (without the `LOOP` option) to fetch data from a device, the HP-71 uses this sequence of messages:

```
Unlisten
Talk Address
Send Data
(Data Byte from device)
:
(Data Byte from device)
Not Ready For Data
(last Data Byte from device)
Untalk
```

Passing Control of HP-IL

The HP-71 can operate as the HP-IL controller, as described in the previous portion of this section. The HP-71 is also capable of operating as an HP-IL device under the direction of another controller, as described in the next portion of this section. The transition between these two conditions is the subject of the next two topics:

- Giving up control of HP-IL. Describes how the HP-71 can stop being the controller of the HP-IL system.
- Acquiring control of HP-IL. Describes how the HP-71 can gain control of the HP-IL system.

In many situations, HP-IL devices must carefully transfer the responsibility of “controller” so that HP-IL operation isn’t disrupted. If the transition isn’t an orderly one, the situation could occur in which either there are two controllers (which isn’t permitted) or there is no controller (which isn’t practical).

In other situations, the loop may be inactive, and controller responsibility can be changed without worrying about system conflicts.

The HP-IL interface status returned by `STATUS` indicates the current controller status of the HP-71:

- If bit 5 is set, the HP-71 is controller.
- If bit 5 is clear, the HP-71 is operating as a device.

Giving Up Control

The HP-71 normally operates as the HP-IL controller. Whenever you install the HP-IL interface or execute `RESET HPIL`, the HP-71 begins operating as a controller.

The HP-71 provides two statements for giving up control of the HP-IL system. For both statements, the HP-71 directs the transfer of control.

- `PASS CONTROL`. This statement permits an orderly transfer of control from the HP-71 to another HP-IL device. The transfer follows standard HP-IL procedures and ensures that another device does take control.
- `CONTROL OFF`. This statement sets the HP-71 to operate as an HP-IL device, but *doesn’t* instruct another device to take control. The HP-71 “assumes” that another device is in control.

If the HP-71 must transfer control during program execution, you’ll probably want to use the `PASS CONTROL` statement. This statement eliminates the problem of an improper transfer. The need to pass control to another device can arise from either of two situations:

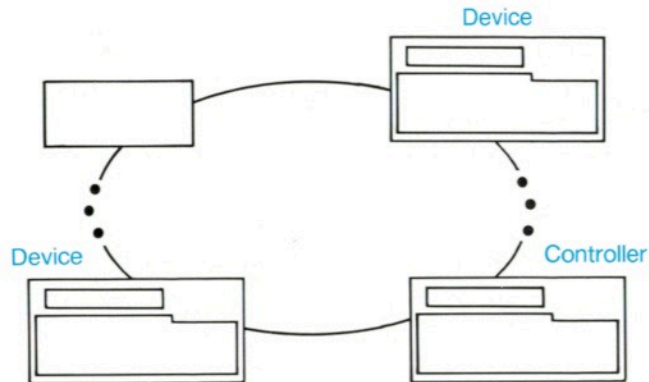
- The HP-71 needs another device to take control.
- A device requests control from the HP-71. This request is normally conveyed as a *service request* from the device. The HP-71 can detect a service request either in the HP-IL interface status (using `STATUS`) or else as an interrupt event (using `ENABLE INTR`, `ON INTR`, and `READINTR`). In either case, the HP-71 might have to determine the source and nature of the request by checking the status of each device (using `SPOLL`).

For both situations, the process of passing control is directed by the HP-71.

It’s possible for another HP-IL device to seize control from the HP-71. The HP-71 *automatically* gives up control of the HP-IL system whenever a device sends an Interface Clear message—an HP-IL message that instructs every device to clear its controller (and listener and talker) status. Although the HP-71 may not anticipate this shift in responsibility, it can check for such a change (using `STATUS`) or can enable this HP-IL interrupt event (using `ENABLE INTR`).

For example, if another HP-71 executes `RESTORE IO` or `CONTROL ON` (which send an Interface Clear message), all remaining HP-71's give up control.

Example: Suppose an HP-71 is the controller of an HP-IL system in which other HP-71 “devices” are connected. While executing a main program, the HP-71 controller periodically executes a subprogram that checks for another HP-71 that is requesting control of the system, and then passes control to that device.



The following subprogram can be used by the controller.

```

10 SUB PASSCNTL
20 SEND IDY @ I=0

30 IF BIT(STATUS,3)=0 THEN END

40 I=I+1
50 H$="HP71(" & STR$(I) & ")"
60 S=SPOLL(H$)
70 IF S=224 THEN 90
80 IF S=-1 THEN END ELSE 40

90 PASS CONTROL H$
100 ENDSUB
  
```

Sends HP-IL message that can be modified by service request.

Ends subprogram execution if bit 3 of HP-IL interface status (service request) is clear.

Sets device counter.

Sets H\$ as device specifier for Ith HP-71.

Sets S to status of specified HP-71.

Branches if status is 224 (request control).

Ends subprogram execution if no status returned (no HP-71 for I).

Passes control to Ith HP-71.

Ends subprogram execution.

Acquiring Control

The HP-71 provides two statements for acquiring control of the HP-IL system. For both statements, the HP-71 initiates the transfer of control.

- **REQUEST.** This statement enables the HP-71 to request control of the system from the controller. This statement can request service from the controller and set the HP-71 device status to a standard “request control” condition. If the controller recognizes this condition, it can pass control to the HP-71. This process provides an orderly transfer of control from the old controller to the HP-71.
- **CONTROL ON.** This statement immediately sets the HP-71 to operate as the HP-IL controller. It then sends an Interface Clear message, which should cancel the controller status of the old controller (if possible). The HP-71 “assumes” that it is possible and acceptable to take control of the system. Other HP-71’s give up control in response to this statement.

If the HP-71 must acquire control during program execution, you’ll probably want to use the **REQUEST** statement. This statement eliminates the problem of an improper transfer. The old controller directs the process of passing control.

It’s possible for the controller to initiate the transfer of control to the HP-71. The HP-71 *automatically* assumes control of the HP-IL system whenever the controller sends it a Take Control message—an HP-IL message that instructs the receiver (which must be a talker) to take control. Although the HP-71 may not anticipate this shift of responsibility, it can check for such a change (using **STATUS**) or can enable this HP-IL interrupt event (using **ENABLE INTR**).

When the HP-71 acquires control of the system, you’ll normally want it to assign new addresses to HP-IL devices. Addresses are assigned by executing **RESTORE IO** or **CONTROL ON** or by turning the HP-71 off and on.

Example: In the previous example, the controller periodically checks for an HP-71 that is requesting control. If an HP-71 is operating as a device, it can use the following subprogram to request control from the controller.

```
10 SUB RQSTCNTL
20 IF BIT(STATUS,5)=1 THEN END

30 REQUEST 224

40 IF BIT(STATUS,5)=0 THEN 40

50 REQUEST 128
60 ENDSUB
```

Ends execution of subprogram if bit 5 of the HP-IL interface status is set (the HP-71 is already controller).

Sets the HP-71 “device” status to 224, which requests service from the controller and indicates a “request control” condition.

If bit 5 of the HP-IL interface status is clear (not controller), branches to this same statement. (Continues only when bit 5 is set—when the HP-71 is controller.)

Resets HP-71 “device” status.

Operating As a Device

The following topics give an overview of the general I/O operations that the HP-71 can perform while it's an HP-IL *device*. This means that the HP-71 *isn't* controlling the operation of the HP-IL system—instead, it's being controlled by another device (the controller). (The controller *could* be another HP-71.)

Refer to “Operating As a Controller” on page 35 for information about how the HP-71 operates while it's the HP-IL controller—this is the condition of the HP-71 when you first install the HP-IL interface and when you execute `RESET HPIL`. Refer to “Passing Control of HP-IL” on page 51 for information about how the HP-71 can acquire control of the system and give up control of the system.

While the HP-71 is operating as an HP-IL device, it automatically processes routine HP-IL instructions—those that don't involve a data transfer. However, when the HP-71 is to perform any specialized operation (specifically, sending or receiving data), its interaction with HP-IL must be defined by one of two methods:

- The HP-71 executes a program that controls its interaction with the HP-IL system. This program can direct the HP-71 to perform HP-IL operations (such as sending or receiving data) or set parameters that affect its interaction.
- The HP-71 receives BASIC instructions that control its interaction. These instructions can be executed from the keyboard or received from the controller (as described under “Receiving BASIC Commands” on page 59).

In addition, the interaction with HP-IL must be *coordinated* with the operations being conducted by the controller. For example, if the controller instructs the HP-71 to send data, the HP-71 should be prepared to send data—otherwise, HP-IL operation may be suspended.

Advanced User's Note: Appendix B provides reference information that describes the responses of the HP-71 to individual HP-IL messages while it's operating as a device. This information may be needed for some advanced applications.

Taking an HP-IL Address

All devices are normally assigned HP-IL addresses by the controller. The HP-71 can be assigned two types of addresses:

- A simple address (an “auto” address). The HP-71 is assigned an integer address.
- An extended address. The HP-71 is assigned a primary address and a secondary address—together, they form the address of the HP-71.

Certain HP-IL controllers may be able to identify devices by their HP-IL characteristics, as an alternative to using HP-IL addresses. As a device, the HP-71 has these characteristics:

- Accessory ID: 3.
- Device ID: “HP71”.

As described in section 1, the HP-71 as a controller can specify a device by using its accessory ID or its device ID. However, an HP-71 controller has no device word that corresponds to an HP-71 operating as a device.

Sending Data

In order to transfer data between devices, the controller designates both the sender and the receiver(s) of the data, then instructs the sender to begin sending.

Three statements cause the HP-71 to send data when directed by the controller:

- `OUTPUT LOOP`. Sends numeric and string data.
- `PRINT` (after `PRINTER IS LOOP`). Sends numeric and string data.
- `COPY ... TO :LOOP`. Sends complete files.

Note that none of these statements allow you to specify the device that receives the data—that’s the responsibility of the controller.

The `OUTPUT` statement is the primary statement for sending data from the HP-71. However, while the HP-71 is operating as a device, `OUTPUT` and `PRINT` have almost identical capabilities.

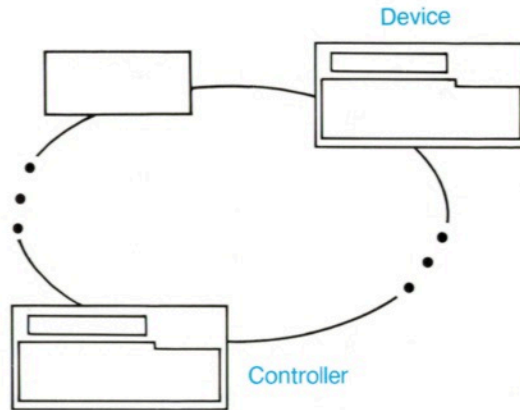
- `OUTPUT` and `PRINT` let you specify the output data as a sequence of numeric and string variables and expressions.
- `OUTPUT` has no restriction on the length of an output sequence. `PRINT` sends data that’s formatted according to the line length specified by `PWIDTH`.
- `OUTPUT` and `PRINT` can use the `USING` option, which enables you to define how the data is formatted when it’s sent. The format is specified using the same symbols as the HP-71 `IMAGE` statement.

The `COPY` statement, which is mentioned in section 3, is primarily a mass storage statement. Among its capabilities is the ability to send information from the HP-71. It compares with the `OUTPUT` statement as indicated below:

- `OUTPUT` transfers numeric and string data, which normally has the form of sequences of characters in one or more lines of output. `COPY` transfers file information, which includes directory information and the actual contents of the file, which may be unintelligible to devices that aren’t file-oriented devices.
- `OUTPUT` can use the `USING` option, which enables the output to be formatted. `COPY` transfers the file information in its stored form.

The preceding comparison shows that `OUTPUT` (and `PRINT`) and `COPY` differ in their fundamental purposes: `OUTPUT` (and `PRINT`) sends numeric and string data that's specified by variables and expressions, `COPY` sends complete file information.

Example: Consider an HP-IL system containing two HP-71 computers. The HP-71 device executes the first program below, which sends a numeric array on HP-IL. The HP-71 controller executes the second program, which fetches the data and stores it internally. (The programs assume that the computers are already set up as controller and device.)



Device Program:

```

10 DIM X(20)
  :
100 INPUT N
110 IF N>20 THEN N=20
120 OUTPUT LOOP;N
130 FOR I=1 TO N
140 OUTPUT LOOP;X(I)
150 NEXT I
  :

```

Dimensions `X` for up to 20 numbers.

Accepts number of values from keyboard.

Checks size of `N`.

Sends number of values on HP-IL.

Sends one value on HP-IL.

Controller Program:

```

10 A=DEVADDR("HP71")
20 ENTER A;N
30 DIM X(N)
40 FOR I=1 TO N
50 ENTER A;X(I)
60 NEXT I
  :
```

Sets *A* to address of HP-71 device.

Fetches number of values from HP-71 device.

Dimensions *X* for *N* numbers.

Fetches one value from HP-71 device.

Receiving Data

As mentioned above, the controller transfers data between devices by designating the sender and the receiver(s), then instructs the sender to begin sending.

Two statements cause the HP-71 to receive data when instructed by the controller:

- **ENTER LOOP.** Receives numeric or string data.
- **COPY ;LOOP.** Receives complete file information.

Note that neither statement allows you to specify the device that sends the data—that's the controller's responsibility.

The **ENTER** statement is the primary statement for receiving data from HP-IL. The **ENTER** statement lets you specify a sequence of variables in which the data is stored.

Like the **OUTPUT** statement, the **ENTER** statement provides the **USING** option. This enables you to define how the data is interpreted when it's received. The format is specified using symbols that are compatible with the conventions of the HP-71 **IMAGE** statement and the **OUTPUT** statement.

The **COPY** statement is primarily a mass storage statement. Among its capabilities is the ability to receive file information into the HP-71. It compares with the **ENTER** statement as indicated below:

- **ENTER** receives numeric and string data, which normally has the form of sequences of characters. **COPY** receives file information, which includes directory information and the actual contents of the file.
- **ENTER** can use the **USING** option, which enables the input to be interpreted according to a specified format. **COPY** receives the file information in its stored form.

The preceding comparison shows that **ENTER** and **COPY** differ in their fundamental purposes: **ENTER** fetches numeric and string data and stores it in variables, **COPY** fetches file information and stores it in a file.

Example: Consider the same HP-IL system as illustrated in the previous example. The HP-71 device executes the first program below, which specifies a file to be sent to it. The HP-71 controller executes the second program, which copies the specified program.

Device Program:

```
10 INPUT "File name: ";N$
20 OUTPUT LOOP;N$
30 ENTER LOOP;M$
40 DISP M$;
50 IF M$="Error" THEN BEEP @ GOTO 10

60 COPY :LOOP
70 DISP " Done"
80 END
```

Accepts file name from keyboard.

Sends file name on HP-IL.

Receives message from HP-IL.

Displays message (file name or error).

If message is for error, returns for another file name.

Receives file from HP-IL into main RAM.

Displays completion message.

Controller Program:

```
10 A=DEVADDR("%3")
20 ENTER A;N$
30 ON ERROR GOTO 90
40 CAT N$

50 OFF ERROR
60 OUTPUT A;N$
70 COPY N$ TO :A
80 STOP
90 OUTPUT A;"Error" @ BEEP
100 GOTO 20
```

Sets A to address of HP-71 device.

Fetches file name from HP-71 device.

Defines branch for error (file not found).

Displays catalog entry of file (and tests whether file exists).

Cancels error branching.

Sends file name to HP-71 device.

Copies file to HP-71 device.

Stops execution.

For error, sends error message to HP-71 device.

Branches for next attempt.

Receiving BASIC Commands

Although the controller is in charge of the HP-IL system, it *doesn't* dictate the internal operation of the HP-71 device. The controller controls only the *interaction* between devices.

However, the HP-71 provides a way for the controller to "tell" the HP-71 what to do. Under certain conditions, the controller can issue commands to the HP-71, and the HP-71 will execute them:

- The controller has set the HP-71 to Remote mode, in which it will accept commands from a “remote” source, and
- The HP-71 isn’t busy—it isn’t executing a program or any BASIC statement and it isn’t in CALC mode.

The natural “language” for these HP-71 commands is the BASIC programming language used by the HP-71. Under the conditions mentioned above, the HP-71 accepts strings of characters and interprets them as BASIC statements. Each valid statement is executed as though it were entered from the keyboard.

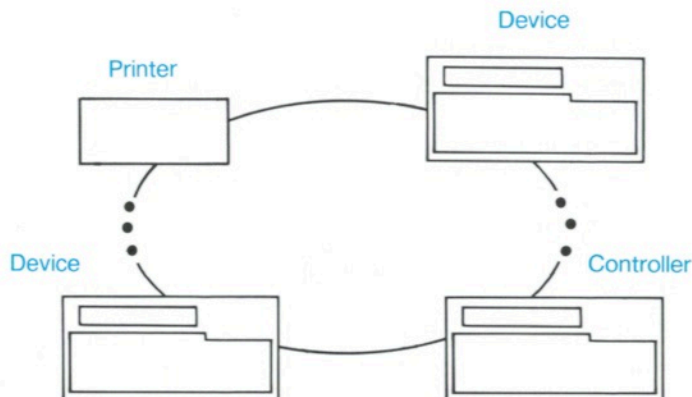
If you happen to be keying an instruction into the HP-71 when a BASIC command is received, the HP-71 isn’t busy—the displayed information is lost as the command is executed. However, if you’re typing a response to an `INPUT` statement, the HP-71 *is* busy—your input isn’t disturbed.

If the HP-71 is in Remote mode and it’s busy, incoming characters are retained in a 64-byte input buffer—waiting to be processed. If the HP-71 becomes not busy, the characters are processed as a BASIC command. On the other hand, if the HP-71 remains busy and executes an `ENTER LOOP` or `COPY :LOOP` statement, the characters are processed as input.

If the HP-71 has been turned off, it automatically turns on and executes the commands.

If the HP-71 is in Local mode, it *won’t* accept HP-IL data as commands—instead, it will use the data as ordinary data, which it will accept with an `ENTER LOOP` or `COPY :LOOP` statement.

Example: Consider an HP-IL system consisting of several HP-71 computers operating as devices under the control of another HP-71. Using the following program, the controller instructs each HP-71 device to request the name of its operator and send the name to the controller, and then prints a list of all operators.



```

10 PRINTER IS PRINTER
20 I=1
30 REMOTE

40 A=DEVADDR("HP71("&STR$(I)&")")
50 IF A=-1 THEN LOCAL @ STOP

60 OUTPUT A;"INPUT 'Enter name:';N$"

70 OUTPUT A;"OUTPUT LOOP;N$"

80 ENTER A;N$
90 PRINT "Operator";I;"is ";N$
100 I=I+1
110 GOTO 40

```

Sets PRINTER IS device.

Initializes counter.

Enables device to change to Remote mode when it receives data.

Sets A to address of Ith HP-71.

If no HP-71 found, sets all devices to Local mode and stops execution.

Sets HP-71 device to Remote mode and sends instruction telling it to accept name from keyboard.

Sends instruction to HP-71 device, telling it to send name on HP-IL.

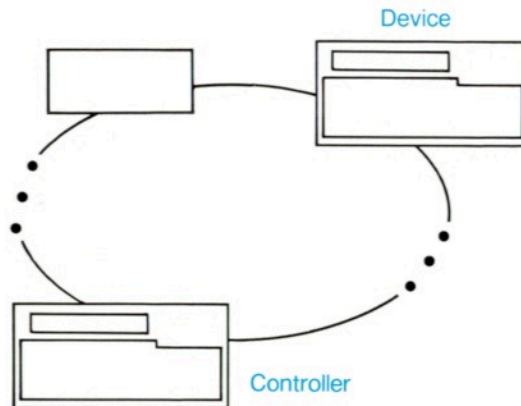
Fetches name from HP-71 device.

Prints counter and name on list.

Increments counter.

Branches for next HP-71.

Example: Consider two HP-71 computers connected in an HP-IL system. One HP-71 operates as controller; the other HP-71 operates as a device. You can use the following keystrokes on the controller to fetch a copy of a file that's stored in the other HP-71.



Input/Result

```
REMOTE HP71 [END LINE]
OUTPUT HP71;"COPY DATA
  TO :LOOP" [END LINE]
COPY :HP71 TO DUPL [END LINE]
LOCAL [END LINE]
CAT DUPL [END LINE]
```

DUPL	DATA	123
------	------	-----

Sets HP-71 device to Remote mode.

Sends instruction to HP-71 device, telling it to send file DATA on HP-IL.

Copies file from HP-71 device to main RAM file DUPL.

Sets all devices to Local mode.

Displays catalog entry of main RAM file DUPL.

Sending Device Information

The HP-71 can provide useful information about itself. This information is often used by the controller for deciding what actions to perform. HP-71 information can be separated into two categories:

- Identifying information: information that helps identify the HP-71.
Accessory ID: 3.
Device ID: "HP71".
- Operating information: information that conveys the condition of the HP-71.

The controller may instruct a device to send its accessory ID or its device ID, and then use this information to determine what kind of device it is. All accessory ID values from 0 through 15 indicate devices that are primarily controller-type units—the HP-71 is in this class.

In certain applications, the controller may check the *status* of HP-IL devices to decide whether they're ready to perform a certain operation. The information that the HP-71 provides to the controller must be defined by the REQUEST statement. REQUEST defines the status information—it's sent out when the controller instructs the HP-71 to send it. This status information can convey the current condition of the HP-71; the actual meaning of the status value that you set using REQUEST depends upon your application. You can either use standard HP-IL conventions for status information, or else define your own status conventions.

Notice that the HP-71 doesn't automatically define its own status. Use REQUEST to define the status (unless you want the status to be 0).

Example: The following lines, when included at the beginning and end of a program, set the HP-71 device status to show whether the HP-71 is executing a program or whether it's idle. (It defines a status of 2 to mean that it's executing a program, and a status of 6 to mean that it's idle.) The controller can then check the HP-71 device status to find out whether the HP-71 is ready for another operation.

```
10 REQUEST 2
```

```
:
```

```
998 REQUEST 6
```

```
999 END
```

Sets status to 2, meaning that a program is executing.

Sets status to 6, meaning that the HP-71 is (soon to be) idle.

Ends execution.

Requesting Service

The controller directs all actions of the HP-IL system—a device normally can't tell another device what to do. However, in some applications, a device may occasionally need to have a particular action taken. It can do this by indicating a *service request* to the controller, and then expect the controller to take action as required.

A service request is normally conveyed to the controller by a special bit in certain HP-IL messages. Any device is permitted to set this “service request” bit—however, the controller isn't required to recognize or respond to a service request.

A service request doesn't carry any indication of which device requests service. If the controller detects a service request, it can check devices in two ways:

- Serial poll. The controller asks each device in turn to send its status information. A device's status information usually indicates whether it has requested service.
- Parallel poll. The controller gets an indication from all devices simultaneously regarding which device has requested service. The controller sets up each device's response to a parallel poll. The HP-71 responds automatically according to its service request condition. (Refer to page 218 for more information.)

The HP-71 can indicate a service request to the controller. The `REQUEST` statement defines the status of the HP-71 device. Bit 6 of the status byte controls the service request condition of the HP-71. So `REQUEST` actually serves two purposes:

- It controls whether the HP-71 indicates a service request.

If bit 6 is set, the HP-71 requests service from the controller.

If bit 6 is clear, the HP-71 doesn't request service.

- It defines the HP-71 status, which the HP-71 sends when instructed by the controller. (This is discussed in the previous topic.)

Example: Consider an HP-71 operating as a device in an HP-IL system. The following subroutines use service requests to indicate a need for receiving data and a need for sending data. The controller can respond to a service request by checking the HP-71 status and taking action according to the “standard” system status meanings.

```
1000 "DATAIN":
1010 REQUEST 225
```

```
1020 ENTER LOOP;X$
```

```
1030 REQUEST 128
```

```
1040 RETURN
```

```
⋮
```

```
2000 "DATAOUT":
```

```
2010 REQUEST 226
```

```
2020 IF BIT(STATUS,4)=0 THEN 2020
```

```
2030 OUTPUT LOOP;Y$
```

```
2040 REQUEST 128
```

```
2050 RETURN
```

Label for input subroutine.

Sets status to “ready to receive data” state and requests service.

Receives string data from HP-IL (when sent by controller).

Resets status to “all okay” state.

Returns to main routine.

Label for output subroutine.

Sets status to “ready to send data” state and requests service.

Ensures that controller makes HP-71 a talker before sending data and cancelling service request.

Sends string data on HP-IL.

Resets status to “all okay” state.

Returns to main routine.

Using HP-IL Interrupts

The HP-71 (without the HP-IL interface) has the capability to recognize and respond to certain “interrupt” events. The HP-71 `ON TIMER` and `ON ERROR` statements cause branching after completing the statement during which a timer interval expires or an error occurs.

The HP-IL interface provides an interrupt capability based on HP-IL events. HP-IL interrupts cause *end-of-line* branching. (If a computer checks for interrupt events only at the end of each program line, the interrupt branching is called “end-of-line” branching.)

The HP-71 can recognize and respond to any of seven HP-IL events (while it is a device):

- The HP-71 has been instructed to become the controller of the HP-IL system.
- The HP-71 has been instructed to send data.
- The HP-71 has been instructed to receive data.
- The HP-71 has received a “device-dependent” instruction.

- The HP-71 has been “triggered”—instructed to perform an action, which isn’t specified by the HP-IL event.
- The HP-71 has been instructed to reset its internal conditions (clear its HP-IL input and output buffers).
- The HP-71 has been instructed to clear its status as controller, talker, and listener.

By using interrupt branching, an HP-71 program can carry out its normal processing, but also be able to monitor continuously for special events. If a special event occurs, the program can branch to a routine that takes any action that’s required by the controller.

Three statements determine how the HP-71 responds to HP-IL interrupt events:

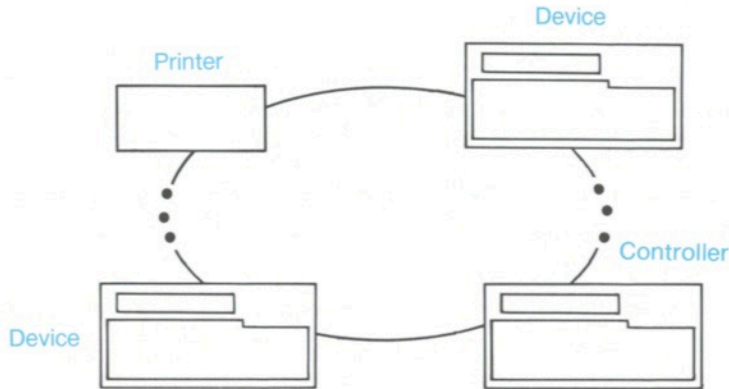
- The `ENABLE INTR` statement defines which HP-IL events can cause interrupt branching—which events are *enabled*.
- The `ON INTR` statement defines how a program branches when an enabled event does occur.
- The `OFF INTR` statement prevents a program from branching—even if an enabled event occurs.

The HP-71 also provides three functions that enable you to determine what conditions exist when an interrupt occurs:

- The `READINTR` function returns the value of the *interrupt byte*, which indicates the interrupt events that have occurred. (It also resets this byte.)
- The `STATUS` function returns the value of the *status byte*, which indicates the HP-IL condition that exists at the time it is read.
- The `READDDC` function returns the number of a “device-dependent” instruction that has been received.

Each of these functions is useful for deciding what action to take when an interrupt occurs. `READINTR` enables you to take action based on the interrupt event that occurred. `STATUS` enables you to take action based on the HP-IL condition that exists at the moment. If the interrupt event is the receipt of a device-dependent instruction, `READDDC` enables you to take action based on which device-dependent instruction was received.

Example: Consider an HP-IL system containing several HP-71 computers that are controlled by another HP-71. An operator stationed at each HP-71 can enter messages from the keyboard. The controller monitors the system for service requests and instructs the devices to send needed information.



Device Program:

```

10 ON INTR GOTO 100
20 INPUT "Enter name: ";N$
30 SFLAG -3
40 INPUT M$

50 ENABLE INTR 1

60 REQUEST 226

70 GOTO 70
100 D=READDDC @ R=READINTR

110 IF D=0 OR D>2 THEN 70
120 IF D=1 THEN OUTPUT LOOP;N$
130 OUTPUT LOOP;M$
140 REQUEST 0
150 GOTO 40

```

Defines branching for HP-IL interrupts.

Accepts operator's name from keyboard.

Prevents HP-71 from turning off.

Accepts message for controller from operator. (Execution waits at this line until message needed.)

Enables one interrupt event: receiving device-dependent command.

Sets status to "ready to send data" state and requests service.

Waits for interrupt event.

Sets \square to number of device-dependent command, which caused interrupt; resets interrupt-cause byte.

Branches for improper command number.

For command number 1, sends operator name.

For command numbers 1 and 2, sends message.

Resets status to "all okay" state.

Branches for next message.

Controller Program:

```

10 PRINTER IS PRINTER
20 SFLAG -3
30 J=0
40 ON INTR GOTO 100
50 ENABLE INTR 8
60 SEND IDY @ GOTO 60

100 I=1 @ R=READINTR

110 A=DEVADDR("HP71("&STR$(I)&")")
120 IF A=-1 THEN 50
130 S=SPOLL(A)
140 IF S#226 THEN I=I+1 @ GOTO 110

150 IF I=J THEN X=2 ELSE X=1

160 SEND TALK A DDT X UNT

170 IF X=1 THEN ENTER A;N$ ELSE N$=""

180 ENTER A;M$
190 PRINT N$,M$

200 J=I
210 GOTO 50

```

Assigns PRINTER IS device.

Prevents HP-71 from turning off.

Initializes "last operator" number.

Defines branching for HP-IL interrupts.

Enables one interrupt event: service request.

Repeatedly sends HP-IL messages that can indicate a service request.

Initializes HP-71 counter; resets interrupt-cause byte.

Sets *R* to address of *I*th HP-71.

Branches to main routine after last HP-71.

Sets *S* to status of *I*th HP-71.

Loops back for next HP-71 if status isn't "ready to send data."

Sets device-dependent command number to 2 if same HP-71 as last time; otherwise, sets number to 1.

Prepares HP-71 (address *R*) to receive device-dependent command, then sends the command.

For command number 1, fetches operator name; otherwise, sets name string to null string.

Fetches message from HP-71.

Prints name string (null if same name as last time) and message.

Sets *J* to last HP-71 number.

Branches to main routine.

Sending HP-IL Messages

For most situations, the statements and functions described elsewhere in sections 1 through 4 will provide convenient ways to perform most HP-IL operations—you won't have to concern yourself with

The HP-IL interface provides five “binary” functions. These function are useful for working with decimal numbers when you want to interpret them as binary numbers. Four of the functions perform standard binary operations on the individual bits of one or two numbers. The following table lists the binary functions and their operation.

Binary Functions

Function	Operation
<code>BINAND</code>	Bit-by-bit AND of two numbers.
<code>BINCOMP</code>	Bit-by-bit complement (1's complement) of one number.
<code>BINEOR</code>	Bit-by-bit exclusive-OR of two numbers.
<code>BINIOR</code>	Bit-by-bit inclusive-OR of two numbers.
<code>BIT</code>	Value of one bit of a number.

The first four binary functions are useful for deriving a new value that's a modification of an old value. For example, you can use the `BINAND` function to “mask out” certain bits of a binary number, giving a new decimal value for the revised binary number. (Refer to the first example below.)

The `BIT` function is useful for finding the state of one bit of a binary number. This function is often used in an `IF ... THEN` statement to control execution according to the state of a single bit.

Example: The following program segments accept a value for an interrupt mask. This value will be used to enable certain HP-IL interrupt events to cause branching within the program.

The `BINAND` function in the first segment revises the input value, preventing the program from using two particular interrupt events: “service request” (bit 3, value 8) and “device-dependent command” (bit 0, value 1). (Note that $255 - 8 - 1 = 246$.)

```
10 INPUT "Interrupt mask";M
20 ON INTR GOSUB 100

30 ENABLE INTR BINAND(M,246)

:
```

Accepts interrupt mask from keyboard.

Defines subroutine branching for HP-IL interrupt.

Enables all specified interrupt events except those for bits 3 and 0.

The `BINIOR` function in the next segment revises the input value, ensuring that the program also uses two particular interrupt events described above (values 8 and 1).

```
10 INPUT "Interrupt mask";M
20 ON INTR GOSUB 100

30 ENABLE INTR BINIOR(M,9)

:
```

Accepts interrupt mask from keyboard.

Defines subroutine branching for HP-IL interrupt.

Enables all specified interrupt events plus those for bits 3 and 0.

Example: The function definition in the following program uses `BIT` to convert a decimal number into its `N`-bit binary representation.

```
10 INPUT "Number, word size ";X,N
20 DISP X;FNB$(X,N)

30 PAUSE
40 GOTO 10
100 DEF FNB$(X,N)
110 X=INT(X)
120 B$=""
130 FOR I=1 TO N
140 B$[I,I]=STR$(BIT(X,N-I))

150 NEXT I
160 FNB$=B$
170 END DEF
```

Accepts a decimal number and binary word size from the keyboard.

Displays the number and its binary representation.

Pauses execution for viewing.

Branches for next number.

Begins function definition.

Truncates fractional part.

Initializes binary string.

Repeats for `N` bits.

Defines one character in string according to value of bit in input number. (Note that bit numbering differs from string index.)

Defines function value.

Ends function definition.

Part II

Keyword Dictionary

Keyword Dictionary

This “Keyword Dictionary” provides a complete description of each statement and function implemented by the HP 82401A HP-IL Interface. The dictionary serves several purposes:

- To summarize the effects of each keyword.
- To give complete information about each keyword so that you can learn how to use it properly.
- To provide reference information that gives a concise summary of each keyword’s operation.
- To give cross-references to other keywords that perform related or similar operations.

The keyword dictionary is organized as a *reference* tool—you’re not expected to read it from beginning to end. To use this manual most efficiently, refer to part I for an overview of the HP-IL capabilities of the HP-71 and its HP-IL interface. From that overview, you’ll probably decide that certain keywords will be useful for your applications. Refer to those entries in the keyword dictionary to learn how to use them (and to learn of other related keywords that may be useful for your applications). In this way, you’ll focus your effort on only those keywords that are useful to you. (Refer to “How To Use This Manual” on page 9.)

Organization

Entries in the keyword dictionary are arranged in alphabetical order. The same format is used for every keyword entry so that you can quickly find the information you need. The format is similar to that used in the *HP-71 Reference Manual*—refer to that manual for additional details.

Each keyword entry provides the following information for the keyword:

- **Keyword name.** Shows the basic keyword.
- **Purpose.** Gives a one-line summary of the operation that the keyword performs.
- **Keyword type.** Identifies the keyword as a *statement* or as a *function*. (None of the keywords are operators.)
- **Execution options.** Indicates situations in which you can execute the keyword:
 - ☐ From the keyboard.
 - ☐ In CALC mode.
 - ☐ After THEN or ELSE in an IF ... THEN ... ELSE statement.
 - ☐ While the HP-71 is operating as an HP-IL device (not as controller).

Note that all keywords provided by the HP-IL interface are *programmable*—they can be used in programs.

- **Syntax diagram.** Defines the required and optional components within the statement or function for proper syntax. (Refer to “Reading Syntax Diagrams” below for additional information.)
- **Examples.** Illustrates and explains some ways that the keyword can be used, and shows some possible syntax variations.
- **Input parameters.** Defines the parameters used in the syntax diagram, gives their default values (if applicable), and lists restrictions on parameter values or structure. (This heading isn’t included for keywords that use no parameters.) “Standard” parameters that are widely used are defined under “Definitions of Standard Terms” below.
- **Operation.** Gives a detailed description of the keyword’s operation and other information that’s useful for learning and using the keyword. (Using a keyword improperly usually causes an error condition—refer to appendix E for information about error messages that the HP-71 uses.)
- **HP-IL messages.** Lists the primary HP-IL messages that the HP-71 sends as it executes the keyword. This is included for cross-reference purposes—you can often anticipate a device’s reaction by checking in its owner’s manual for its response to the listed HP-IL messages. (This list isn’t intended to show the entire message sequence used by the HP-71, but rather to give only the most important messages.) This information is included for general I/O keywords only.
- **Related keywords.** Lists other keywords that either influence the results of the subject keyword or else are similar in function.

Reading Syntax Diagrams

A syntax diagram is included for each keyword. These diagrams indicate all acceptable ways of using the keywords. (If a keyword is used incorrectly, the HP-71 indicates an error condition.)

Use the following guidelines for interpreting the syntax diagrams:

- Items in ovals and circles must be entered as indicated, using either uppercase letters or lowercase letters.
- Items in boxes describe parameters that are defined under the heading “Input Parameters.” You must enter parameters that are consistent with the definitions.
- Arrows define valid paths through the syntax diagrams—any sequence of items that’s allowed by a valid path is a proper construction for that keyword. Looping paths indicate items that can be repeated.
- An item is optional if there’s a valid path around it.
- Spaces should not be included within items in ovals unless explicitly shown. Indicated spaces are optional—they may be omitted.

- Spaces normally are entered between the items in ovals, circles, and boxes. Such spaces may be omitted if the omission doesn't cause confusion with another meaning.
- Double-quote marks are shown in circles before and after certain items. As an alternative, you can use single-quote marks—but the opening and closing quote marks must match.
- The diagram for every statement has an arrow at the exit path. This means that you can use the @ symbol to concatenate another statement.

Definitions of Standard Terms

Several syntax items are common to many of the keyword syntax diagrams. These “standard” items are used to define an HP-IL device or to define a file in an HP-IL device.

The following table gives standard definitions for common syntax items. They're listed in alphabetical order. Certain items use other items as part of their definitions.

Item	Description	Restrictions
accessory ID	Numeric expression, which is rounded to an integer. (This indicates the accessory ID of an HP-IL device.)	0 through 255.
accessory type	Two-part item: 1. A % symbol followed by an <i>accessory ID</i> . 2. Optional <i>sequence number</i> .	None.
assign code	String expression (or unquoted string) that evaluates either to one letter or else to a letter followed by a letter or digit. (Avoid using as an unquoted string—if an assign code is also a valid variable name, it will be interpreted as an HP-IL address.)	None.
device ID	String expression (or unquoted string). (This indicates the device ID of an HP-IL device—only the specified characters are compared.)	None.

Item	Description	Restrictions
device specifier	Any one of the following: <ul style="list-style-type: none"> • <i>accessory type</i> with option of starting with : (colon). • <i>assign code</i> with option of starting with : (colon). • <i>device type</i> with option of starting with : (colon). • <i>device word</i> with option of starting with : (colon). • <i>HP-IL address</i> with option of starting with : (colon). • A . (period) followed by a <i>volume label</i>. 	None.
device type	Two-part item: <ol style="list-style-type: none"> 1. A <i>device ID</i>. 2. Optional <i>sequence number</i>. 	None.
device word	Two-part item: <ol style="list-style-type: none"> 1. Any one of the following reserved words: <ul style="list-style-type: none"> • DISPLAY • GPIO • GRAPHIC • HPIB • INSTRMT • INTRFCE • MASSMEM • MODEM • PRINTER • RS232 • TAPE 2. Optional <i>sequence number</i>. 	None.
file name	String expression (or unquoted string) that evaluates to a letter and up to nine additional letters and digits.	None.

Item	Description	Restrictions
file specifier	Two-part item: <ol style="list-style-type: none">1. A <i>file name</i>.2. A <i>device specifier</i> that <i>must</i> include either a : (colon) or a . (period).	None.
HP-IL address	Numeric expression, which is rounded to two decimal places: <ul style="list-style-type: none">• Integer part (indicates device's simple address or primary part of extended address).• Fractional part (if nonzero, indicates device's secondary part of extended address).	1 through 30. .01 through .31.
loop	String expression (or unquoted string) that evaluates to LOOP, with option of starting with : (colon).	None.
sequence number	Numeric expression enclosed in < >; expression is rounded to an integer. (This defines <i>n</i> th device of specified type.) Default: 1.	1 through 16.
volume label	String expression (or unquoted string) that evaluates to a letter and up to five additional letters and digits.	None.

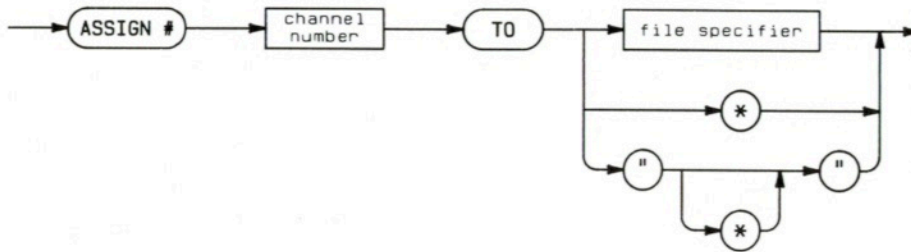
Other terms that have special meanings in this manual are defined in the glossary in appendix F (or in the glossary in the *HP-71 Reference Manual*).

ASSIGN

Associates an I/O channel number with a file and opens the file.

- Statement
- Function
- Operator

- Keyboard Execution
- CALC Mode
- IF...THEN...ELSE
- Device Operation



Examples

ASSIGN # 1 TO DATAFILE:TAPE

Associates channel 1 with file DATAFILE located in the first mass storage device and opens the file.

ASSIGN #50 TO NOTES:3

Associates channel 50 with file NOTES on device at address 3 and opens the file.

ASSIGN #(C-3) TO FILE1,VOLUM1

Associates computed channel number with file FILE1 on medium with volume label VOLUM1.

ASSIGN #4 TO *

Closes the file associated with channel 4 and cancels that association.

Input Parameters

Item	Description	Restrictions
channel number	Numeric expression, which is rounded to an integer.	1 through 255.
file specifier	See standard definition.	None.

ASSIGN # (continued)

Operation

The `ASSIGN #` statement associates a symbolic channel number with the specified file and opens the file (makes it accessible). The specified device must be a file-oriented device—that is, one that normally contains files. The specified file must already exist—a file can be created using `CREATE`. Refer to the *HP-71 Reference Manual* for information about using files in main RAM (or independent RAM).

To reduce the number of times that the HP-71 accesses the device, the HP-71 creates an internal I/O buffer for that channel and reads the first 256 bytes (“physical” record) from the file into that buffer. Subsequent `READ #`, `PRINT #`, and `RESTORE #` operations interact with the I/O buffer—the HP-71 reads and writes to the external file only when necessary.

Assigning a channel to `*` or `""` closes the associated file—that is, it cancels that channel number’s association with a file. This also updates the file according to the latest information in the I/O buffer.

Similarly, if a channel number is already associated with a file and you assign it to another file, the previous file is closed.

A file is *automatically* closed by any of the following operations:

- Executing `RUN` (but not `CHAIN`).
- Executing `END`, `STOP`, or an “implied” `END` (after last program line).
- Executing `END SUB`. (This closes only the files opened by the subprogram.)

ASSIGN # (continued)

The following table summarizes the effects of various operations upon I/O channels.

Effects of Operations on I/O Channels

Operation	Specified I/O Channel		All I/O Channels	
	Updates File*	Closes File	Updates Files*	Closes Files
ATTN halts running program			X	
Error halts running program				
PRINT # from keyboard	X			
ASSIGN # TO *	X	X		
ASSIGN # to new file	X	X		
PAUSE			X	
STOP			X	X†
END			X	X†
END SUB			X	X†
END ALL			X	X
RUN				X
CALL				
CHAIN				
* Only if contents have been changed.				
† Only files opened within the program or subprogram.				

If the HP-71 is operating as an HP-IL device, you can execute **ASSIGN #** only if the specified file is in main RAM (or in independent RAM).

Related Keywords

CREATE.

ASSIGN IO

Associates assign codes with HP-IL devices.

■ Statement

□ Function

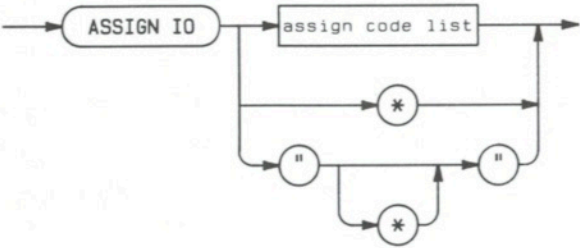
□ Operator

■ Keyboard Execution

□ CALC Mode

■ IF...THEN...ELSE

□ Device Operation



Examples

ASSIGN IO ":TV,:TA,:PR"

Sets up specified assign codes for first three HP-IL devices.

ASSIGN IO A\$

Sets up assign codes specified by string variable.

ASSIGN IO *

Cancels all assign codes.

Input Parameters

Item	Description	Restrictions
assign code list	String expression that evaluates to assign codes separated by commas. Each assign code has option of starting with : (colon).	None.

Operation

The `ASSIGN IO` statement sets up assign codes and associates them with devices at sequential HP-IL addresses. You can then use these assign codes to specify the corresponding devices in subsequent operations.

ASSIGN IO (continued)

Assign codes are associated with HP-IL devices starting at the first address. That is, the first assign code is associated with the first device in the loop (the one connected to the HP-IL interface's OUT receptacle) and so on. Although you can't skip devices when assigning codes, you may specify fewer codes than the number of devices in the loop. (Too many assign codes cause an error and cancel all previous assign codes and device assignments.)

Whenever `ASSIGN IO` is executed, it enables I/O operation (if it has been disabled by `OFF IO`), cancels previous assign codes, and cancels previous `PRINTER IS` and `DISPLAY IS` devices. It also establishes simple (sequential) addresses for HP-IL devices—that is, extended addresses aren't used, even if flag `-22` is set.

Specifying `*` or `" "` cancels all assign codes.

You should note that assign codes are associated internally with HP-IL *addresses*, not with individual *devices*. When you use an assign code to specify a device, the HP-71 doesn't have to search for the device—it already knows the address. If you change the loop configuration, you should execute `ASSIGN IO` for the new configuration—this ensures that assign codes are associated with the proper addresses (and devices).

HP-IL Messages

Auto Address.

Related Keywords

`LIST IO`.

BINAND

Returns the value of the binary AND operation performed on two integers.

☐ Statement

☒ Function

☐ Operator

☒ Keyboard Execution

☒ CALC Mode

☒ IF...THEN...ELSE

☒ Device Operation



Examples

A=BINAND(Y,Z)

X=BINAND(X,2^N-1)

IF N THEN N=BINAND(C,7)

Sets A to the value of an AND operation performed on Y and Z.

Sets X to the value of its last N bits.

If N is true, sets N to value corresponding to last three bits of C.

Input Parameters

Item	Description	Restrictions
integer expression	Numeric expression, which is rounded to an integer.	0 through 1,048,575.

Operation

The BINAND function returns the decimal value of a bit-by-bit AND operation performed on two integer values. Each integer value is specified as a decimal value and is considered by the binary operation as a 20-bit number.

BINAND (continued)

Each bit in the result is determined by the corresponding bits in the two arguments according to the following table:

First Argument	Second Argument	AND Result
0	0	0
0	1	0
1	0	0
1	1	1

HP-IL Messages

None.

Related Keywords

`BINCMP`, `BINEOR`, `BINIOR`, `BIT`.

BINCMP

Returns the value of the binary (1's) complement of an integer.

☐ Statement

☒ Function

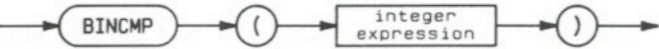
☐ Operator

☒ Keyboard Execution

☒ CALC Mode

☒ IF...THEN...ELSE

☒ Device Operation



Examples

```
C1=BINCMP(C)
```

Sets C1 to the value of the complement of C.

```
DISP BINCMP(A DIV 2)
```

Shifts the binary representation of A one bit to the right and displays its complement.

Input Parameters

Item	Description	Restrictions
integer expression	Numeric expression, which is rounded to an integer.	0 through 1,048,575.

Operation

The BINCMP function returns the decimal value of the binary complement of an integer value. The integer value is specified as a decimal value and is considered by the binary operation as an unsigned 20-bit number.

BINCMP (continued)

Each bit in the result is determined by the corresponding bit in the argument according to the following table:

Argument	Complement Result
0	1
1	0

If you're assuming that bytes are less than 20 bits long, you can form complements using the `BINEOR` function. For example, the eight-bit complement of `B` is given by `BINEOR(B, 255)`.

HP-IL Messages

None.

Related Keywords

`BINAND`, `BINEOR`, `BINIOR`, `BIT`.

BINEOR

Returns the value of the binary exclusive-OR operation performed on two integers.

☐ Statement

☒ Function

☐ Operator

☒ Keyboard Execution

☒ CALC Mode

☒ IF...THEN...ELSE

☒ Device Operation



Examples

IF M=1 THEN A=BINEOR(Y,Z)

If M = 1, sets A to the value of the exclusive-OR operation performed on Y and Z.

Y=BINEOR(X,2^N-1)

Sets Y to the value of N-bit complement of X.

Input Parameters

Item	Description	Restrictions
integer expression	Numeric expression, which is rounded to an integer.	0 through 1,048,575.

Operation

The BINEOR function returns the decimal value of a bit-by-bit exclusive-OR operation performed on two integer values. Each integer value is specified as a decimal value and is considered by the binary operation as a 20-bit number.

BINEOR (continued)

Each bit in the result is determined by the corresponding bits in the two arguments according to the following table:

First Argument	Second Argument	Exclusive-OR Result
0	0	0
0	1	1
1	0	1
1	1	0

HP-IL Messages

None.

Related Keywords

BINAND, BINCMP, BINIOR, BIT.

BINIOR

Returns the value of the binary inclusive-OR operation performed on two integers.

☐ Statement

☒ Function

☐ Operator

☒ Keyboard Execution

☒ CALC Mode

☒ IF...THEN...ELSE

☒ Device Operation



Examples

IF BINIOR(Y,Z)=255 THEN 410

Branches to line 410 if the value of the inclusive-OR operation performed on Y and Z is 255 (last eight bits set).

X=BINIOR(X,8)

Sets the fourth from last bit in X to 1.

Input Parameters

Item	Description	Restrictions
integer expression	Numeric expression, which is rounded to an integer.	0 through 1,048,575.

Operation

The BINIOR function returns the decimal value of a bit-by-bit inclusive-OR operation performed on two integer values. Each integer value is specified as a decimal value and is considered by the binary operation as a 20-bit number.

BINIOR (continued)

Each bit in the result is determined by the corresponding bits in the two arguments according to the following table:

First Argument	Second Argument	Inclusive-OR Result
0	0	0
0	1	1
1	0	1
1	1	1

HP-IL Messages

None.

Related Keywords

`BINAND`, `BINCMPL`, `BINEOR`, `BIT`.

BIT

Returns the value of one bit of an integer.

☐ Statement

☒ Function

☐ Operator

☒ Keyboard Execution

☒ CALC Mode

☒ IF...THEN...ELSE

☒ Device Operation



Examples

B=BIT(R,B1)

IF BIT(R,3) THEN R=R-2^3

IF BIT(S,6) THEN GOSUB "SRQ"

Sets B to the value of bit B1 in R.

If bit 3 is set in R, changes R so that bit 3 isn't set.

If bit 6 is set in S, branches to subroutine labelled SRQ.

Input Parameters

Item	Description	Restrictions
integer expression	Numeric expression, which is rounded to an integer.	0 through 1,048,575.
bit position	Numeric expression, which is rounded to an integer.	0 through 19.

Operation

The `BIT` function returns the value 0 or 1 according to the value of the bit at a specified position in an integer value. The integer value is specified as a decimal value and is considered by the binary operation as a 20-bit number. The bit position is from 0 (the least significant bit) to 19 (the most significant bit).

This function enables you to test the state of individual bits, which is useful for branching decisions in programs. The returned values are compatible with logic operators: 1 (true) and 0 (false).

HP-IL Messages

None.

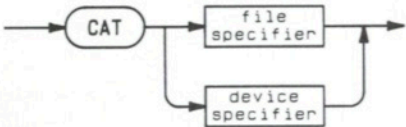
Related Keywords

BINAND, BINCMP, BINEOR, BINIOR.

CAT

Gives a catalog of file information.

- ☒ Statement
☐ Function
☐ Operator
- ☒ Keyboard Execution
☐ CALC Mode
☒ IF...THEN...ELSE
☐ Device Operation



Examples

CAT PHONE:TAPE(3)

Gives catalog entry of file PHONE in third mass storage device.

CAT ,VOLUM1

Gives catalog of mass storage medium having volume label VOLUM1.

CAT A\$(2)

Gives catalog of file or medium specified by second element in string array A\$.

Input Parameters

Item	Description	Restrictions
file specifier	See standard definition.	None.
device specifier	See standard definition.	Must begin with : (or ,).

Operation

The CAT statement provides the following information about each file in an HP-IL device:

- Name.










CAT (continued)

- Protection:

P	Private.
S	Secure.
E	Execute only (private and secure).
- Type:

BASIC	BASIC program.
BIN	Binary program.
DATA	Data (fixed-length records).
LEX	Language extension.
KEY	Key assignment.
SDATA	Stream data (compatible with HP-41).
TEXT	ASCII data in LIF (type 1) format (variable-length records).
- Length (in bytes).
- Creation date and time (shown as *mm/dd/yy hh:mm*).

If you specify a file name in the CAT statement, the HP-71 displays the catalog entry for that file. In a program, the catalog entry is displayed and execution continues.

If you don't specify a file name, CAT displays the catalog header and the catalog of the first file stored in the device. You can examine other entries by pressing the  and  keys—entries are displayed in the same order that they're stored in the device. Press   or   to display the entry for the first or last file. In a program, execution pauses at that statement so that the entire catalog can be reviewed manually. Press   (or any key other than a display control “arrow” key or ) to continue execution.

Refer to the *HP-71 Reference Manual* for information about using CAT with files in main RAM (or independent RAM).

If a file has a type that doesn't correspond to one of the types listed above, CAT displays the type as a number from -32768 to 32767. This number is obtained from the type field according to the LIF (Logical Interchange Format) standard. (Refer to appendix D for information about the LIF standard.)

If the HP-71 is operating as an HP-IL device, the CAT statement can be used only if the file is located in main RAM (or in independent RAM).

Related Keywords

CAT#, COPY, CREATE, PURGE.

CAT\$

Returns a string containing catalog information for a file.

☐ Statement

☒ Function

☐ Operator

☒ Keyboard Execution

☐ CALC Mode

☒ IF...THEN...ELSE

☐ Device Operation



Examples

```
F$=CAT$(2,";MASSMEM")
```

DISP CAT\$(X,".DATA")

Sets F\$ equal to the catalog string for the second file in the first mass storage device.

Displays the catalog string for file number X on the mass storage medium with volume label DATA.

Input Parameters

Item	Description	Restrictions
file number	Numeric expression, which is rounded to an integer.	None.
device specifier	See standard definition.	Must not be unquoted string.

Operation

The CAT\$ function returns a string that contains catalog information for the specified file on the specified mass storage medium. The file is specified by its number in the catalog—the first catalog entry corresponds to file number 1, and so on.

CAT\$ (continued)

The string returned by CAT\$ has a length of 40 characters and contains the same information as displayed by the CAT statement:

NNNNNNNNNN	S	TTTTT	ZZZZZ	MM/DD/YY	HH:XX
Name	↓	Type	↓	Month	Year
	Security		Size	Day	Hour

If there is no file on the medium at the specified position, CAT\$ returns a null string. (This includes 0 and negative file numbers.)

Refer to the *HP-71 Reference Manual* for information about using files in main RAM (or independent RAM).

If the HP-71 is operating as an HP-IL device, you can use the CAT\$ function only if the specified file is located in main RAM (or in independent RAM).

Related Keywords

CAT, COPY, CREATE, PURGE.

CLEAR

Clears an individual HP-IL device or all HP-IL devices.

■ Statement

□ Function

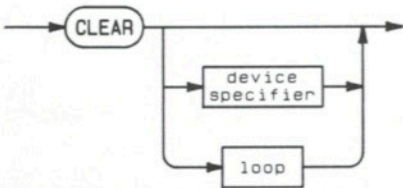
□ Operator

■ Keyboard Execution

□ CALC Mode

■ IF...THEN...ELSE

□ Device Operation



Examples

IF X=2 THEN CLEAR ":DISPLAY(2)"

Clears device with assign code ":I1".

IF A THEN CLEAR LOOP

If A is true, clears all devices.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.
loop	See standard definition.	None.

Operation

The `CLEAR` statement either clears an individual HP-IL device or else clears all HP-IL devices. If a device specifier is included, the HP-71 clears the indicated device. If no device is specified, or if `LOOP` is specified, the HP-71 clears *all* HP-IL devices.

CLEAR (continued)

When a typical device is cleared, it reverts to its startup conditions in terms of operating conditions. For example, the HP 82162A Thermal Printer homes its printhead, clears its print buffer, and sets its print modes to their startup conditions. **CLEAR** does *not* affect the device's interface status, such as its HP-IL address.

HP-IL Messages

Device Clear (for **LOOP** only), Listen Address and Selected Device Clear (for *device* only).

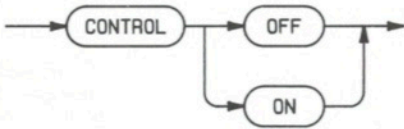
Related Keywords

SEND.

CONTROL OFF/ON

Sets the controller status of the HP-71, either causing the HP-71 to operate as the HP-IL controller or causing it to operate as an HP-IL device (*not* as the controller).

- | | |
|-------------|----------------------|
| ■ Statement | ■ Keyboard Execution |
| □ Function | □ CALC Mode |
| □ Operator | ■ IF...THEN...ELSE |
| | ■ Device Operation |



Examples

CONTROL OFF

Sets the HP-71 to immediately start operating as an HP-IL device.

IF X=20 THEN CONTROL ON

If $X = 20$, sets the HP-71 to immediately assume control of the HP-IL system.

Operation

If a **CONTROL** statement hasn't been executed since the HP-IL interface was installed, the HP-71 will be operating as *controller* of the HP-IL system. Similarly, whenever the HP-IL interface is reset (using **RESET HPIL**), the HP-71 automatically becomes the HP-IL controller.

The **CONTROL OFF** statement causes the HP-71 to immediately stop acting as the controller of the HP-IL system. The HP-71 doesn't "pass" control to another device. If another device is not acting as controller, the loop will remain idle. While the HP-71 is not controller, it can't access individual HP-IL devices—except as arranged by another device that acts as the controller.

Note: The HP-IL capabilities that the HP-71 can exercise while it is *not* the controller are indicated in this keyword dictionary by a solid box next to "Device Operation." Refer to appendix B for detailed information about how the HP-71 responds as a device.

CONTROL OFF/ON (continued)

The `CONTROL ON` statement causes the HP-71 to immediately begin acting as the controller of the HP-IL system. The HP-71 doesn't "request" control from another controller. If another device is still acting as controller, conflicts may arise and the loop may not operate properly. (The HP-71 tries to clear the controller status of other devices, but some devices don't automatically clear their controller status. This statement *will* clear the controller status of another HP-71.)

Because `CONTROL ON` involves HP-IL interaction, it can't be executed if I/O operations have been disabled by `OFF IO`. (If I/O operations have been disabled, execute `RESTORE IO` first.)

`CONTROL ON` sets HP-IL conditions for assuming control of the HP-IL system:

- Activates the `DISPLAY IS` device.
- Assigns addresses to HP-IL devices.
- Clears the "busy" status of all HP-IL devices (including controller, talker, and listener status).

HP-IL Messages

Interface Clear (only for `CONTROL ON`), Auto Extended Secondary and Auto Extended Primary (only for `CONTROL ON` and only if flag -22 set), Auto Address (only for `CONTROL ON`).

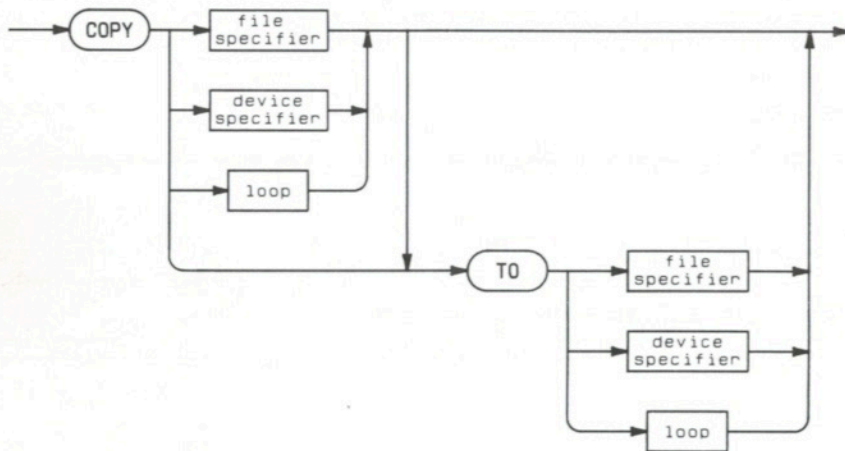
Related Keywords

`PASS CONTROL`, `REQUEST`.

COPY

Copies a file from one location to another.

- | | |
|-------------|----------------------|
| ■ Statement | ■ Keyboard Execution |
| □ Function | □ CALC Mode |
| □ Operator | ■ IF...THEN...ELSE |
| | ■ Device Operation |



Examples

`COPY START:TAPE(2)`

Copies file `START` on second mass storage device to new file `START` in main RAM.

`COPY TO BACKUP.MASTER`

Copies current file to file `BACKUP` on medium with volume label `MASTER`.

`COPY PRGM1:CA TO PRGM2`

Copies file `PRGM1` in device `" :CA"` to new file `PRGM2` in main RAM.

`COPY TO :MASSMEM`

Copies current file to first mass storage device.

`COPY FLIM:2 TO FLAM:3`

Creates file `FLAM` in device at address 3, a duplicate of file `FLIM` in device at address 2.

`COPY DATA:TAPE TO :INTRFCE`

Copies file `DATA` in first mass storage device to first interface device.

COPY (continued)

COPY "RPT" TO :LOOP

Sends the contents of file RPT in main RAM to HP-IL, but not to a particular device. (Useful when operating as HP-IL device.)

Input Parameters

Item	Description	Restrictions
file specifier	See standard definition. Source default: Current file. Destination defaults: <ul style="list-style-type: none">• File: Same name as source.• Device: Main RAM.	Device specifier is optional. Also see "Operation."
device specifier	See standard definition.	Must begin with : (or .). Also see "Operation."
loop	See standard definition.	Must begin with :.

Operation

The COPY statement copies a file from one device to another. This I/O operation can involve files in HP-71 main RAM and in HP-IL devices. (For information about performing COPY operations that don't involve HP-IL devices, refer to the *HP-71 Reference Manual*.)

The source and destination devices for COPY operations can be grouped into three categories:

- Internal HP-71 devices (main RAM, independent RAM, external ROM).
- HP-IL devices (mass storage device, interface device).
- LOOP (unspecified HP-IL device).

The first two categories involve "specified" devices in the sense that the source and destination are defined by the COPY statement. The LOOP option involves an "unspecified" device in the sense that the COPY statement doesn't determine the device indicated by LOOP.

For all COPY operations, the HP-71 is directly involved in the data transfer. It receives the data from the source device and then sends it to the destination device. It also checks the file's directory entry to ensure that the file is a valid HP-71 file.

COPY (continued)

Using “Specified” Devices. The COPY statement normally must create a new file at the destination—the single exception occurs when the destination is an existing file in an HP-IL mass storage device. This exception covers the common situation in which you wish to update a mass storage file (without having to purge it first). (The mass storage file must not be secure.) For all other situations, the destination file must *not* exist when COPY is executed.

When copying files between two specified devices (including to or from an internal HP-71 device), the specifier for either the source or destination (or both) must include a file name—with two exceptions:

COPY TO *device specifier*

Gives new file same name as current file.

COPY *device specifier*

If device *isn't* a mass storage device, gives new main RAM file the name specified by the device when it sends the directory entry (see below). If device *is* a mass storage device, operation isn't possible.

The name of the source file either is specified (by the file specifier), is the same as the destination file name (for a device specifier), or is the current file (for no source specifier).

When a new file is created at the destination, it is given the name defined by the COPY statement—either specified or default (same as the source name). The new file is the same type, size,* and record length as the source file. If you copy a system file (*keys* or *workfile*) and don't specify a destination file name, the HP-71 modifies the name to uppercase letters (*KEYS* or *WORKFILE*).

You should be careful when using COPY with a device that *isn't* a file-oriented device (mass storage device). The following paragraphs describe this situation.

The HP-71 receives and sends file information that is formatted according to a defined standard—the Logical Interchange Format (LIF) standard. (The LIF standard is described in appendix D.) A 32-byte LIF directory entry is sent just before the actual contents of the file. Devices should consider the first 32 bytes as directory information.

In addition, files used by the HP-71 are encoded according to the type of information in the file. A file-oriented device merely stores the encoded information for later use—the actual information isn't used by the device.

If you copy a file *from* a device that isn't a file-oriented device, the HP-71 will expect a directory entry and will interpret the first 32 bytes as that entry. The entry it receives must be a proper directory entry.

* The size of a TEXT file is rounded to the next multiple of 256 bytes.

COPY (continued)

If you copy a file *to* a device that isn't a file-oriented device, the HP-71 will send the 32-byte directory entry followed by the encoded contents of the file. The destination device may try to process the information, giving results that may not be useful.

Using LOOP. The `LOOP` option can be used to copy files to or from an internal HP-71 device. It is useful for two types of situations:

- The HP-71 has already prepared HP-IL devices for receiving (or sending) information and the HP-71 is already designated to send (or receive) data. Specifying an internal HP-71 device and `:LOOP` as the two devices causes the HP-71 to transfer the contents immediately, *without* setting up HP-IL devices. In this situation, the `COPY` statement doesn't set up the sender or the receiver—it just transfers the information from an HP-IL device that's a talker or to any HP-IL devices that are listeners (as may be set up by a previous `SEND` statement). (Refer to the `SEND` statement.)
- The HP-71 is operating as an HP-IL device. In this situation, the HP-71 can't control other HP-IL devices. However, it *can* send or receive files on HP-IL. If you specify `:LOOP` as the source and an internal HP-71 device as the destination, the HP-71 waits for the directory entry to arrive on HP-IL, creates the appropriate new file, and then copies the file contents from HP-IL to the new file. If you specify an internal HP-71 device as the source and `:LOOP` as the destination, the HP-71 prepares itself to send out the directory entry and file contents (filled out to the next 256-byte record boundary). For each of these operations, the HP-71 issues no HP-IL instructions.

If the HP-71 is operating as an HP-IL device, it *must* use `:LOOP` to specify an HP-IL transfer.

Refer to appendix D for information about the LIF standard and mass storage compatibility.

Related Keywords

CAT, INITIALIZE, PACK, PACKDIR, PURGE, RENAME.

CREATE

Creates a data file (DATA, SDATA, or TEXT).

■ Statement

□ Function

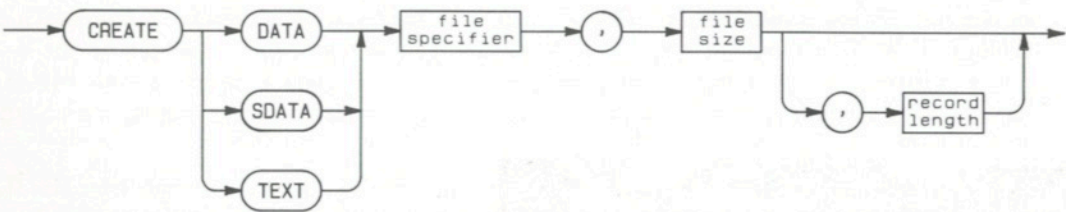
□ Operator

■ Keyboard Execution

□ CALC Mode

■ IF...THEN...ELSE

□ Device Operation



Examples

```
CREATE TEXT "FILE6:1",500
```

Creates TEXT file FILE6 with 500 bytes in device at address 1.

```
CREATE SDATA F$,40
```

Creates SDATA file with name and device specified by F\$, and 40 “registers” long.

```
CREATE DATA RIM.SHOT,10,50
```

Creates DATA file RIM with ten 50-byte records on medium with volume label SHOT.

Input Parameters

Item	Description	Restrictions
file specifier	See standard definition.	None.
file size	Numeric expression, which is rounded to an integer.	See “Operation.”
record length	Numeric expression, which is rounded to an integer. Default: See “Operation.”	0 through 65,535.

CREATE (continued)

Operation

The `CREATE` statement creates a new data file. The type of file determines how information is stored in the file:

- **DATA.** This type of file has fixed-length records that can contain numeric and string values. It allows both sequential and random access. (This type of file is similar to HP Series 80 data files.)
- **SDATA.** This type of file has fixed, eight-byte records that contain numeric values and ASCII strings (up to six characters). (This type of file is compatible with HP-41 DA (data) files.)
- **TEXT.** This type of file has variable-length records that contain ASCII data. (This type of file is compatible with the Logical Interchange Format (LIF) standard, with HP-75 LIF1 files, and with HP-41 AS (ASCII) files.)

The file name specified in the `CREATE` statement must not already be used by the specified device. (If it is already used, `CREATE` does *not* destroy that file—instead, an error occurs.)

The file size specified in the `CREATE` statement determines the maximum number of records that can be stored in the file—the file can't be lengthened. The record length specifies the number of bytes in each logical record. The way these two parameters are interpreted depends upon the type of data file being created:

Input Parameters

File Type	File Size	Record Length
DATA	Number of logical records. Restrictions: 1 through 65,535.	Number of bytes.* Default: 256.
SDATA	Number of "registers" (8 bytes/register). Restrictions: 1 through 65,535.	Ignored—always 8 bytes.
TEXT	Number of bytes. Restrictions: 1 through 1,048,575.	Ignored.
* A record length of 0 specifies 256 bytes.		

The `CREATE` statement does not *open* the data file—it only *creates* it. Use the `ASSIGN #` statement to access the file.

CREATE (continued)

Refer to the *HP-71 Reference Manual* for information about using files in main RAM (or independent RAM).

If the HP-71 is operating as an HP-IL device, you can use `CREATE` only if the file will be created in main RAM (or independent RAM).

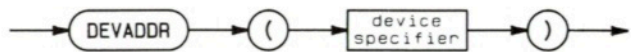
Refer to appendix D for information about the LIF standard and mass storage compatibility.

Related Keywords

`ASSIGN #`, `PURGE`.

Returns the address of a device.

- ☐ Statement
- ☒ Function
- ☐ Operator
- ☒ Keyboard Execution
- ☒ CALC Mode
- ☒ IF...THEN...ELSE
- ☐ Device Operation



Examples

```
C=DEVADDR(D$)
X=DEVADDR("PRINTER(2)")
@ IF X#-1 THEN OUTPUT X;A,B
P=DEVADDR("HP82164A")
@ IF P#-1 THEN ENTER P;X$
T=DEVADDR("%16") @ IF T#-1
THEN COPY FILE1 TO :T
```

Sets C to address of device specified by D\$.

Sets X to address of second printer device, then sends that device the values of A and B.

Sets P to address of HP-IL/RS-232 interface (device ID of "HP82164A"), then fetches data from that device and stores it in X\$.

Sets T to address of first device with accessory ID of 16, then copies file FILE1 to that device.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	Must not be unquoted string.

Operation

The DEVADDR function returns the HP-IL address of the specified device. The device may be indicated by any proper specifier.

If the device has a simple address, the returned number is an integer value equal to the address. If the device has an extended address, the integer part of the returned number equals the primary part and the first two digits of the fractional part define the secondary part. If the specified device isn't in the loop, the returned number is -1.

DEVADDR (continued)

For example, if DEVADDR returns the following numbers, the device is at the address described:

3	The device is at address 3.
5.08	The device is at extended address 5.08.
-1	The device is not in the loop.

The returned address is an acceptable device specifier for other HP-IL operations.

An HP-IL operation is performed most quickly when a device is specified by its address. In this situation the HP-71 doesn't have to search for the desired device. But in many applications the address of a device won't be known in advance. The DEVADDR function enables a program to set a variable to a device's address, so that the address is available for the remainder of the program. In this way, the HP-71 has to search for the device only once.

If DEVADDR is to be used in CALC mode, the argument must be numeric (an address). In this situation, DEVADDR returns the address of the device specified by an address.

HP-IL Messages

Send Accessory ID (for accessory ID only), Send Device ID (for device ID only).

Related Keywords

DEVAID, DEVID\$, SPOLL.

DEVAID

Returns the accessory ID of a device.

<input type="checkbox"/> Statement	<input checked="" type="checkbox"/> Keyboard Execution
<input checked="" type="checkbox"/> Function	<input checked="" type="checkbox"/> CALC Mode
<input type="checkbox"/> Operator	<input checked="" type="checkbox"/> IF...THEN...ELSE
	<input type="checkbox"/> Device Operation



Examples

`DISP I;DEVAID(I)`

Displays the value of I and the accessory ID of the device at address I.

`T=DEVAID("PRINTER")`

Sets T to the accessory ID of the first printer device.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	Must not be unquoted string.

Operation

The **DEVAID** function returns the accessory ID of the specified device. The device may be indicated by any valid specifier.

If a device doesn't have an accessory ID, or if the specified device isn't in the loop, **DEVAID** returns a value of -1.

If the device's accessory ID has more than one byte, **DEVAID** accepts only two bytes and returns the value $256 \times (\text{first byte}) + (\text{second byte})$.

If **DEVAID** is to be used in CALC mode, the argument must be numeric (an address).

The following table indicates accessory ID ranges and the corresponding classes of HP-IL devices. HP-71 device words are listed for appropriate classes.

DEVAID (continued)

Accessory ID Ranges

Accessory ID Range*	Device Class	HP-71 Device Word
0-15	Controller	—
16-31	Mass Storage Device	MASSMEM
32-47	Printer	PRINTER
48-63	Display	DISPLAY
64-79	Interface Device	INTRFCE
80-95	Electronic Instrument	INSTRMT
96-111	Graphics Device	GRAPHIC
112-127	Analytical Instrument	—

* The HP-71 uses the highest accessory ID in each class to indicate *any* device in that class—it can be used when specifying a device. For example, accessory ID 47 is equivalent to device word PRINTER.

HP-IL Messages

Send Accessory ID.

Related Keywords

DEVADDR, DEVID\$, SPOLL.

Returns a string containing the device ID of a device.

- ☐ Statement
☒ Function
☐ Operator
- ☒ Keyboard Execution
☐ CALC Mode
☒ IF...THEN...ELSE
☐ Device Operation



Examples

- ```
DISP I;DEVID$(I)
```

Displays the value of `I` and the device ID of the device at address `I`.
- ```
D$=DEVID$("PRINTER")
```

Sets `D$` to the device ID of the first printer device.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	Must not be unquoted string.

Operation

The `DEVID$` function returns a string containing the device ID of the specified device. The device may be indicated by any valid specifier.

If a device doesn't have a device ID, or if the specified device isn't in the loop, `DEVID$` returns a null string.

HP-IL Messages

Send Device ID.

Related Keywords

`DEVADDR`, `DEVAID`, `SPOLL`.

DISPLAY IS

Assigns one HP-IL device to be the display device.

■ Statement

□ Function

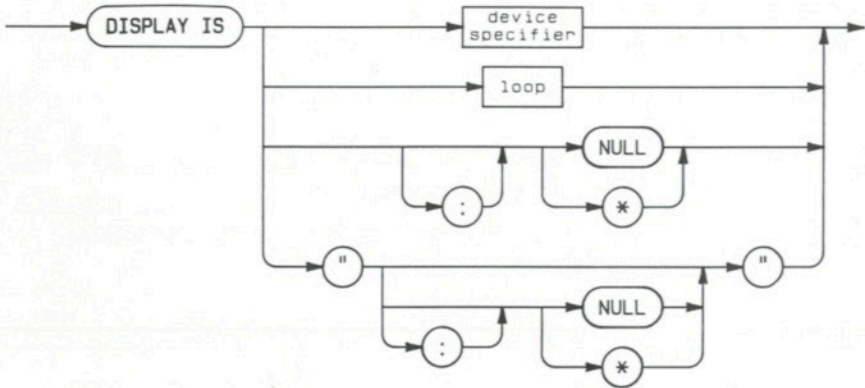
□ Operator

■ Keyboard Execution

□ CALC Mode

■ IF...THEN...ELSE

□ Device Operation



Examples

DISPLAY IS PRINTER(2)

Assigns the second printer device to be the DISPLAY IS device.

DISPLAY IS DISPLAY

Assigns the first display-type device to be the DISPLAY IS device.

DISPLAY IS *

Cancels the DISPLAY IS device.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.
loop	See standard definition.	None.

DISPLAY IS (continued)

Operation

The `DISPLAY IS` statement assigns one HP-IL device as the “display” device. All subsequent display output goes to that device, as well as to the HP-71 display. The assigned `DISPLAY IS` device is used only in BASIC mode—it’s not used in CALC mode.

Whenever a memory reset condition occurs in the HP-71, the HP-71 automatically performs `DISPLAY IS DISPLAY`, which sets the first display device in the loop as the `DISPLAY IS` device.

Assigning `*` or `NULL` or `LOOP` as the display device assigns *no* `DISPLAY IS` device. In this situation, all display output is sent only to the HP-71 display.

Several types of HP-IL devices can be used as a `DISPLAY IS` device. Such devices differ in their capabilities for processing data. For this reason, the HP-71 separates devices into three categories, which determine how it sends display information to them:

- Standard video interface (accessory ID 48). Each character is sent when it is entered from the keyboard; insertion and deletion cause the revised line to be sent. Escape sequences are sent for left (ESC D) and right (ESC C) cursor movement, for normal cursor (ESC R) and insert cursor (ESC Q), and for cursor on (ESC >) and off (ESC <). For displayed output (as from `DISP`), each character is sent as it is generated.
- All printer devices (accessory ID 32 through 47). All characters in a line are sent only after the `END LINE` key is pressed. For displayed output, all characters in a line are sent only after a carriage return is specified.
- All other devices. Each character is sent when it is entered from the keyboard. Escape sequences are sent for begin insertion (ESC N) and end insertion (ESC R), for delete character (ESC O), for left (ESC D) and right (ESC C) cursor movement, and for delete to end of line (ESC K). For displayed output, each character is sent as it is generated.

The HP-71 assumes that the `DISPLAY IS` device is a character-oriented device—that is, one that normally operates with individual characters or lines of characters. Examples are video interfaces and printers. You shouldn’t use a file-oriented device (such as a mass storage device) for receiving display output—the results may be unpredictable.

DISPLAY IS (continued)

Advanced User's Note: While a `DISPLAY IS` device is specified, the HP-71 sends all display data to that device. At appropriate times, the HP-71 automatically prepares the HP-IL system for only that device to accept the display data and for the HP-71 to send data, and later cancels all devices from accepting data.

This causes no conflicts with other I/O operations.

However, an I/O operation may depend upon HP-IL conditions set up by a previous operation (for example, using `SEND` with `OUTPUT LOOP`). Any occurrence that causes an HP-71 display between the two operations will disturb HP-IL conditions set up by the first operation. For this situation, you may want to execute `DISPLAY IS *` first.

If the HP-71 has an assigned `DISPLAY IS` device when it begins operating as an HP-IL device, it automatically stops using that device—all display output is sent only to the HP-71 display. If the HP-71 resumes operating as controller, it automatically resumes using the former `DISPLAY IS` device as soon as the HP-71 assigns new addresses to HP-IL devices. The only `DISPLAY IS` operation the HP-71 can perform while it's a device is `DISPLAY IS *` (or equivalent), which cancels the `DISPLAY IS` device that's used while the HP-71 is controller.

Related Keywords

`PRINTER IS.`

ENABLE INTR

Specifies the events that can cause an HP-IL interrupt.

- ☒ Statement

☐ Function

☐ Operator
- ☒ Keyboard Execution

☐ CALC Mode

☒ IF...THEN...ELSE

☒ Device Operation



Examples

ENABLE INTR L+2^N

Enables an interrupt to occur only for events specified by the value of the expression.

IF E THEN ENABLE INTR 16

If E is true, enables an interrupt to occur only when the HP-71 becomes an active talker device (mask value 16).

Input Parameters

Item	Description	Restrictions
interrupt mask	Numeric expression, which is rounded to an integer.	0 through 255.

Operation

The `ENABLE INTR` statement specifies the events that the HP-71 will consider for causing an HP-IL interrupt. The action that the HP-71 takes when a valid interrupt occurs is defined by the `ON INTR` and `OFF INTR` statements—`ENABLE INTR` specifies which events are to be considered as valid interrupts.

The interrupt mask is a decimal value—when expressed in binary form, its eight bits correspond to the events that are enabled to cause an interrupt. The value of an interrupt mask should be the sum of the decimal values for the events that are to be enabled, as described in the following table. (Notice that certain events can occur while the HP-71 is a controller, and that others can occur while the HP-71 is a device.)

ENABLE INTR (continued)

Interrupt Mask

Bit	Decimal Value*	Interrupt Event	Controller†	Device†
7	128	Interface Clear. Received an Interface Clear message that HP-71 didn't send. (Normally clears controller, talker, and listener status.)	X	X
6	64	Listener. Made a listener by HP-IL controller. (Normally prepares device to receive data.)	(X)	X
5	32	Controller. Made active controller by previous controller. (Tells device to take control of HP-IL system.)		X
4	16	Talker. Made a talker by HP-IL controller and instructed to send data (by Send Data message). (Tells device to send data.)	(X)	X
3	8	Service Request. HP-71 detected service request bit set in HP-IL message. (Normally tells controller that a device needs attention.)	X	
2	4	Device Clear. Received a Device Clear or (while a listener) a Selected Device Clear message. (Normally resets device to operational startup conditions. Clears HP-IL input and output buffers in HP-71.)	X	X
1	2	Trigger. Received a Group Execute Trigger message while a listener. (Normally triggers an event at a device.)	X	X
0	1	Device Dependent. Received a Device Dependent Talker message while a talker, or else received Device Dependent Listener message while a listener. (Meaning depends upon device.)	X	X

* Interrupt mask is sum of values of all events that are to be enabled.

† An "X" indicates that the interrupt event can occur while the HP-71 is a controller or while the HP-71 is a device. An "(X)" indicates that the event can occur using `SEND LISTEN 0` or `SEND TALK 0` only.

For example, if you want to enable an interrupt only when the HP-71 becomes a listener or talker, use `ENABLE INTR 80` or `ENABLE INTR 64+16`.

If the interrupt mask is set to 0, no events are enabled, and no interrupts can occur.

The interrupt mask is automatically set to 0 whenever a valid interrupt occurs. This prevents another interrupt from occurring before a program is ready for one.

ENABLE INTR (continued)

The interrupt mask retains the value assigned by the `ENABLE INTR` statement until:

- An enabled interrupt event occurs.
- Another `ENABLE INTR` statement redefines the interrupt mask.
- A program starts execution (`RUN`, but not `CALL`).
- A program finishes execution (`STOP` or `END`).
- A `RESET HPIL` statement sets the value to 0.

Notice that no interrupt events are enabled when a program starts running. An `ENABLE INTR` statement in the program remains in effect until execution stops or ends—even while another program is running, such as from `CALL` or `CHAIN`.

For additional information about certain HP-IL interrupt events, refer to the following keyword entries:

- Service Request: `SPOLL` and `REQUEST`.
- Device Clear: `CLEAR`.
- Trigger: `TRIGGER`.

HP-IL Messages

None.

Related Keywords

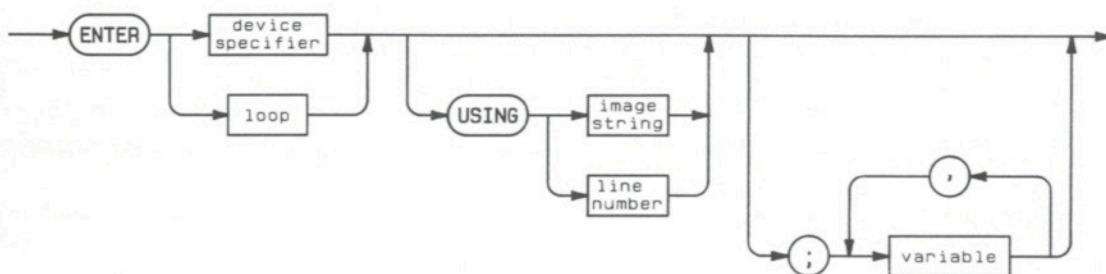
`OFF INTR`, `ON INTR`, `READINTR`.

ENTER

Reads data from HP-IL into numeric and string variables.

- Statement
- Function
- Operator

- Keyboard Execution
- CALC Mode
- IF...THEN...ELSE
- Device Operation



Examples

ENTER 2; N,A\$

Enters from device at address 2 two values, a number stored in N and a string stored in A\$.

ENTER 3 USING 1000; X,Z(1)

Enters from device at address 3 two numeric values stored in X and Z(1). Uses formatting from IMAGE statement at line 1000.

ENTER INTRFACE(2); A1,A2

Enters from second interface device two numeric values, stored in A1 and A2.

ENTER "HP82164A" USING "80A"; X\$,Y\$

Enters two strings from first HP-IL/RS-232 interface (device ID "HP82164A") and stores them in X\$ and Y\$. Each string has 80 characters.

ENTER 3 USING "#,B"; I

Enters from device at address 3 one byte, whose value is stored in I.

ENTER LOOP; B1\$

Enters from HP-IL (device not specified) a string, which is stored in B1\$.

ENTER (continued)**Input Parameters**

Item	Description	Restrictions
device specifier	See standard definition.	None.
loop	See standard definition.	None.
format string	String expression that specifies a valid sequence of image symbols.	None.
line number	Four-digit integer constant that refers to <code>IMAGE</code> statement.	0 through 9999.
variable	Numeric or string variable name.	None.

Operation

The `ENTER` statement fetches numeric or string data and stores it in the specified variables. The data is received as a sequence of characters, which are interpreted by the HP-71 according to a format that is specified or implied by the `ENTER` statement. If no format is specified, then data is entered using “free-field” input; if `USING` is specified, the data is entered according to the format specified by the format string or `IMAGE` statement. Detailed information about these two forms of input are provided under the next two subheadings below.

While accepting numeric input, the HP-71 separates characters into two groups:

- Numeric characters: all digits (0 through 9) and certain other characters when their placement in the input string occurs at a meaningful place in the number (decimal point, plus sign, minus sign, and exponent indicator “E”). Note that the comma is not a numeric character.
- Non-numeric characters: all other characters, including spaces, commas, and line feed characters.

The state of flag `-23` affects the operation of `ENTER`. While flag `-23` is clear, `ENTER` operates as described below. While flag `-23` is set, the operation is altered as described under “Effects of Flag `-23`,” the last subheading below.

The `ENTER` statement normally terminates only when one of three conditions occurs:

- A line feed character is received after all variables have been given values. (Line feed is a common end-of-line character.) If the format *requires* a line feed character at a certain position, that line feed doesn’t terminate the statement.
- All variables have been given values, and the format specifically *doesn’t require* a line feed character.
- An HP-IL End Byte message is received after all variables have been given values. (An End Byte message is a special form of data byte that indicates the end of a logical group of characters.)

ENTER (continued)

Notice that reading data into all specified variables does *not* terminate the ENTER statement (except for formats that eliminate the line feed requirement). Extra characters are ignored.

Advanced User's Note: If the HP-71 is the controller when any one of the terminating conditions occurs or whenever a line feed character or End Byte message is received, the HP-71 sends an HP-IL Not Ready For Data message, which instructs the talker to stop sending data.

An HP-IL End Of Transmission message does *not* terminate the ENTER statement (except as provided by flag -23 set). If the HP-71 is controller, it instructs the device to send additional data until a valid terminating condition occurs.

The LOOP option provides the ability to enter data in two special situations:

- The HP-71 has already prepared a device to send data by making it a talker, and the HP-71 is already designated to receive data (it is a listener). ENTER LOOP causes the current talker to send its data to all listeners. (Refer to the SEND statement.)
- The HP-71 is operating as an HP-IL device. It can't specify the device that is to send data—the controller defines the data source.

If the HP-71 is operating as an HP-IL device, it *must* use the LOOP option. The HP-71 can read data from HP-IL only after the controller makes it a listener. Program execution is affected by the order of events:

- If the HP-71 executes ENTER before it receives data from HP-IL, the program waits at that statement until it receives data.
- If the HP-71 receives data from HP-IL before it executes ENTER, it will accept and store up to 64 characters in its HP-IL input buffer. These characters are fetched by the next ENTER statement. If the HP-71 receives more than 63 characters, the HP-71 holds up loop operation until it executes ENTER. (RESET HPIL clears the input buffer.)

Free-Field Input. For free-field input, numeric data is treated differently from string data. Numeric data must make sense as a numeric value; string data can be *any* sequence of characters.

When entering numeric data using free-field input, the HP-71 basically ignores leading non-numeric characters, accepts numeric characters until a non-numeric character is received, then stores the numeric value in the variable. (Numeric and non-numeric characters are defined above.)

For example, if the HP-71 were trying to derive numeric values for X and Y from the string shown below, the values -12.5×10^{-31} and 400 would be found. (CR and LF represent the carriage return and line feed characters, respectively.)

ENTER (continued)

--TEST--12.5E-31,400(CR)(LF)
 $X = -12.5E-31 \quad Y = 400$

When entering string data using free-field input, the HP-71 basically accepts all characters into a string variable until the string variable is full (extra characters are lost) or until the variable assignment is terminated. Any one of the following conditions terminates the variable assignment:

- A line feed character is received. (This is an end-of-line sequence.)
- A carriage return, line feed sequence is received. (This is an end-of-line sequence.)
- An HP-IL End Byte message is received. (This is a special form of data byte that indicates the end of a logical group of characters.)

Notice that there are two end-of-line sequences that the HP-71 recognizes and *excludes* from the string variable, and that carriage return by itself (not followed by line feed) *isn't* excluded from the string variable.

For example, if A\$, B\$, and C\$ were each dimensioned to a length of five characters, the HP-71 would derive these string variables from the input string shown below.

TEST(LF)(CR)(LF)PATTERN(CR)(LF)
 A\$ = "TEST" B\$ = " " C\$ = "PATTE"

Formatted Input. By specifying a format in an ENTER statement, you obtain additional control in two areas:

- Accurately describing to the HP-71 how the incoming data is formatted and what should be done with it.
- Precisely specifying what condition(s) constitutes the end point of an entry to a variable and the end point of the ENTER statement itself.

The input format can be specified as a format string in the ENTER statement; otherwise, it must be specified as a line number that references an IMAGE statement.

The format string itself must be made up of legitimate *image symbols*, which are described in the table below. One or more symbols form an *image field*, which corresponds to one input variable. Certain symbols may be replicated, as indicated in the table—such a symbol may be preceded by a number that specifies how many times that symbol should be used, or repeated.

ENTER (continued)

Image Symbols for ENTER

Image Symbol	Type of Input	Meaning	May Be Replicated
Numeric Input:			
D	Digit	Accepts one character to be used for building a number. All symbols do the same thing—the different symbols enable you to document the input format so that the OUTPUT statement can share same format. Only the <i>number</i> of characters is important.	Yes
Z	Digit		Yes
*	Digit		Yes
.	Digit		No
S	Digit		No
M	Digit		No
E	Five Digits	Accepts five characters for building a number. The five characters don't have to be in exponential form, but they can be.	No
C	Comma	Accepts one character for building a number, and also causes a comma at this position to be ignored while building this number. Without the "C", a comma would end the number.	No
R	Comma Radix	Accepts one character for building a number, and also causes a comma at this position to be considered the radix point. This is useful for European formatting.	No
P	Period Separator	Accepts one character for building a number, and also causes a period at this position to be ignored while building this number. This is useful for European formatting.	No
K	Free-Field	Accepts a sequence of characters for building a number according to free-field conventions. Field terminated only by line feed character or End Byte message (unless flag —23 set).	No
H	Free-Field	Accepts a sequence of characters for building a number according to free-field conventions. (Exception: For building a number, a comma is considered to be the radix point.) Field terminated only by line feed character or End Byte message (unless flag —23 set).	No
^	Hide	Accepts a sequence of characters for building a number according to free-field formatting, but leaves the corresponding variable unchanged (the data is discarded).	No

ENTER (continued)**Image Symbols for ENTER (continued)**

Image Symbol	Type of Input	Meaning	May Be Replicated
String Input:			
A	Character	Accepts one character for building a string. Any character is accepted.	Yes
K H	Free-Field	Accepts a sequence of characters for building a string according to free-field conventions. Field terminated only by line feed character or End Byte message (unless flag -23 set).	No
^	Hide	Accepts a sequence of characters for building a string according to free-field formatting, but leaves the corresponding variable unchanged (the data is discarded).	No
Binary Input:			
B	Byte	Accepts one character as an eight-bit binary byte and stores it as a (decimal) numeric value.	No
Special Symbols:			
X	Ignore	Accepts, but ignores, one character.	Yes
"..." '...'	Ignore	Accepts, but ignores, the number of characters included within the quotes.	Yes
,	Field Separator	Defines the end of an image field.	No
@	Field Separator	Defines the end of an image field.	Yes
/	Line Feed	Defines the end of an image field and ignores all characters until a line feed is received. (That is, when the previous format item has been satisfied, "/" causes subsequent characters to be ignored, up to and including the next line feed.)	Yes
<...>	Group	Defines an image "group", which is a sequence of one or more image fields. This is useful for repeating one or more fields.	Yes
#	Terminator	Eliminates requirement for line feed to terminate ENTER statement. Statement terminates when last variable is satisfied. This must be the first symbol in the format.	No

ENTER (continued)

Commas, slashes, and “@” signs are used to separate fields. They indicate the end of the field for a variable.

Within a numeric field, the HP-71 ignores leading non-numeric characters, accepts numeric characters until a non-numeric character is received (unless specifically provided by C, R, or P), then stores the numeric value in the variable.

Within a string field, the number of characters specified by the format must not exceed the dimensioned size of the string variable.

Consider the following examples of format strings. (Note that a format string can be included in quotes in the ENTER statement instead of using an IMAGE statement.)

IMAGE DDDCCCC.DD,80A/10A

Interprets the first 10 characters as a number (ignoring the fourth character if it's a comma), the next 80 characters as a string, and the first 10 characters after the next line feed as a string.

IMAGE 5D,2X,4DE

Interprets the first five characters as a number, ignores the next two characters, and interprets the next nine characters as a number.

IMAGE Z,DD,K

Interprets the first four characters as a number and the remaining characters as free-field input.

IMAGE #,B

Interprets the first character as a binary byte, then terminates the operation.

If the ENTER statement hasn't terminated by the time the HP-71 has stepped all the way through the format string, the same format string is used again. Essentially, the format string repeats until the ENTER statement finishes.

Effects of Flag -23. While flag -23 is set, the HP-71 changes the operation of the ENTER statement—but only if the HP-71 is operating as controller.

While flag -23 is clear, or while the HP-71 operates as an HP-IL device, then ENTER operates as described in the preceding discussions above.

The following discussion applies to only the condition in which flag -23 is set and the HP-71 is controller.

ENTER (continued)

The revised operation of **ENTER** provides compatibility with devices that use an HP-IL End Of Transmission message as an end-of-line indicator, rather than the more common line feed character or End Byte message. Specifically, while flag -23 is set, the HP-71 looks for End Of Transmission messages as “terminators”—instead of looking for line feed characters or End Byte messages. In addition, this revised operation is designed for receiving data from interface devices in such a way that usually prevents data from being lost.

While entering numeric data (using free-field input), the HP-71 ignores leading non-numeric characters and begins forming a numeric value when it detects the first numeric character. The HP-71 accepts characters into the numeric value until:

- A non-numeric character is detected. (The HP-71 continues to receive characters until the device stops sending them—when it sends the next End Of Transmission message. The HP-71 uses these additional characters for subsequent variables in the variable list, if any.)
- An End Of Transmission message is received. (The device stops sending characters.)

While entering string data (using free-field input), the HP-71 accepts all characters into the string until:

- The string becomes full—that is, the number of characters reaches the dimensioned size of the string variable. (The HP-71 interrupts the device, stopping it from sending additional characters.*)
- An End Of Transmission message is received. (The device stops sending characters.)

Advanced User’s Note: The HP-71 uses a Not Ready For Data message to interrupt the device. This causes the device to send an End Of Transmission message, which closes the variable definition.

If additional variables in the variable list remain undefined by the **ENTER** statement, the HP-71 instructs the device to send additional data.

The **ENTER** statement terminates when one of the following conditions occurs:

- The last variable in the variable list is a string variable, and the string becomes full. (The HP-71 interrupts the device, stopping it from sending additional characters.)
- All variables have been defined and an End Of Transmission message is received. (The device stops sending characters.)

Notice that a line feed character will close a numeric input (it’s not a numeric character)—but it’s accepted as part of a string, and it doesn’t terminate the **ENTER** operation.

* An exception occurs if a numeric variable is followed by a string variable: the HP-71 doesn’t interrupt the device when the string variable is filled—additional characters are ignored until the next End Of Transmission message.

ENTER (continued)

Notice also that while flag -23 is set, the HP-71 normally interrupts the device when a string becomes full—no characters are lost. (While flag -23 is clear, additional characters are usually ignored.)

HP-IL Messages

Send Data (only for operation as controller), Not Ready For Data.

Related Keywords

OUTPUT, SEND.

INITIALIZE

Initializes a mass storage medium (sets the volume label, creates the directory, and clears the medium).

■ Statement

☐ Function

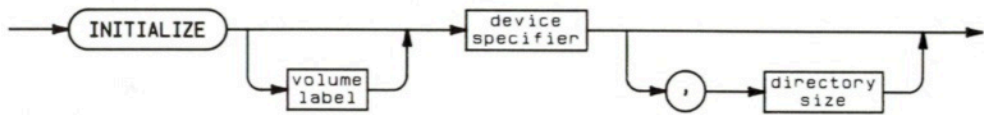
☐ Operator

■ Keyboard Execution

☐ CALC Mode

■ IF...THEN...ELSE

☐ Device Operation



Examples

INITIALIZE :TAPE(2)

Initializes the medium in the second mass storage device, giving it a blank volume label and default directory size.

INITIALIZE TEST1:A,55

Initializes the medium in the device at address A, giving it a volume label of "TEST1" and a 55-entry directory.

INITIALIZE A\$,35

Initializes the medium in the device specified in A\$, giving it a volume label also specified in A\$ and a 35-entry directory.

Input Parameters

Item	Description	Restrictions
volume label	See standard definition. Default: Six spaces.	None.
device specifier	See standard definition.	Must begin with : (or .).
directory size	Numeric expression, which is rounded to an integer, that defines the number of file entries in the directory. Default: See "Operation."	See "Operation."

INITIALIZE (continued)

Operation

The INITIALIZE statement prepares a mass storage for storing information. To do this, the HP-71 performs these steps:

1. Erases the entire medium.
2. Writes the volume label on the medium.
3. Allocates the directory space on the medium.

The stored volume label is up to six characters long. If a longer volume label is specified, only the first six characters are used. If no volume label is specified, six spaces are used—this volume label can't be used to indicate that medium for mass storage operations.

The directory size defines the number of file entries that is to be provided in the directory—that is, the medium can store that many files. (Eight entries can be stored in each directory record—so the actual directory size is the smallest multiple of eight that contains the specified number of entries.) The smallest directory size is 1 (although 8 entries will be provided). The largest size depends upon the number of records on the medium:

$$\text{directory size} \leq \frac{8}{9} (\text{number of records} - 2).$$

However, a larger directory leaves less space for storing information than a smaller directory does. The directory size that you specify might depend upon whether you'll have many short files or a few long files. If the directory size isn't specified, the number of directory *records* is automatically set to $\frac{1}{32}$ of the number of *records* on the medium (any fractional portion is ignored)—the number of directory file *entries* is 8 times that number.

For example, consider the HP 82161A Digital Cassette Drive, which provides 512 records on a cassette tape. The maximum directory size is 453 file entries. If a directory size isn't specified when the cassette is initialized, the directory size is set to 128 file entries.

Related Keywords

CAT, COPY, PURGE.

INITIALIZE: TAPE

INITIALIZE: TAPE, 100

128 files by default.
100 FILES

LIST IO

Lists all defined assign codes and associated HP-IL addresses.

- | | |
|---|--|
| <input checked="" type="checkbox"/> Statement | <input checked="" type="checkbox"/> Keyboard Execution |
| <input type="checkbox"/> Function | <input type="checkbox"/> CALC Mode |
| <input type="checkbox"/> Operator | <input checked="" type="checkbox"/> IF...THEN...ELSE |
| | <input checked="" type="checkbox"/> Device Operation |

LIST IO

Examples

`LIST IO`

Lists all active assign codes.

`IF L THEN LIST IO`

If L is true, lists active assign codes.

Operation

The `LIST IO` statement displays a header that indicates the number of active assign codes, and then sequentially lists each HP-IL address and assign code. It requires that at least one assign code is already active from previously executing an `ASSIGN IO` statement.

The list of assign codes is displayed by the `DISPLAY IS` device (or HP-71 display). The format is shown by the following example:

```
3 Device(s) assigned
Device # 1=':TV'
Device # 2=':PR'
Device # 3=':C '
```

HP-IL Messages

None.

Related Keywords

`ASSIGN IO.`

LOCAL

Sets an individual HP-IL device or all HP-IL devices to Local mode.

- ☒ Statement

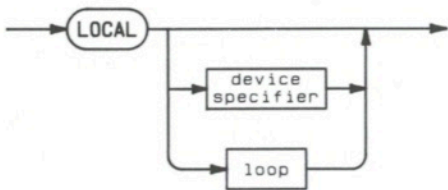
☐ Function

☐ Operator
- ☒ Keyboard Execution

☐ CALC Mode

☒ IF...THEN...ELSE

☐ Device Operation



Examples

`LOCAL`

`IF NOT R THEN LOCAL "HP82164A"`

Sets all HP-IL devices to Local mode (and makes them not Remote enabled).

If R is not true, sets HP-IL/RS-232 interface to Local mode.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.
loop	See standard definition.	None.

Operation

The `LOCAL` statement either sets an individual HP-IL device to Local mode or else sets all HP-IL devices to Local mode. Not all HP-IL devices implement Local mode—for such devices, `LOCAL` has no effect. For a device that does implement Local mode (and Remote mode), its response in Local mode depends upon its design. Typically, a device in Local mode can respond to its “local” controls, such as keys or switches—although certain I/O interface devices reserve Local mode for transferring data between systems.

Regarding Local and Remote modes, the HP-IL system can be in either of two conditions:

- Not Remote Enabled. Devices may not be in Remote mode—all devices must be in Local mode.

LOCAL (continued)

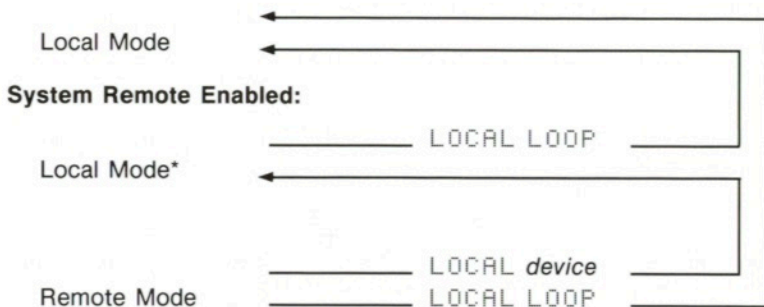
- Remote Enabled. Devices may be in Remote mode or in Local mode, but any device in Local mode automatically changes to Remote mode when it is next instructed to receive data (when it next becomes a listener).

The LOCAL statement has three forms that serve two purposes:

- LOCAL *device* puts one device into Local mode. If the system is Remote enabled, all devices remain Remote enabled—the device changes to Local mode, but it will change back to Remote mode when it is next instructed to receive data (when it next becomes a listener).
- LOCAL and LOCAL LOOP put all devices into Local mode and make the system not Remote enabled.

The following diagram shows how the LOCAL statement affects a device's mode. It also shows how LOCAL affects the system's condition.

System Not Remote Enabled:



* Returns to Remote mode when next instructed to receive data.

The HP-71 is also set to Local mode by RESET HPIL.

If the HP-71 is operating as a device, it can be set to Local mode by the controller. In this mode, the HP-71 interprets data that it receives as normal HP-IL data, which must be entered using the ENTER or COPY statement.

HP-IL Messages

Go To Local (for *device* only), Not Remote Enable (except for *device*).

Related Keywords

LOCAL LOCKOUT, REMOTE.

LOCAL LOCKOUT

Sets all HP-IL devices to Local Lockout condition, but only if the system is Remote enabled.

- | | |
|---|--|
| <input checked="" type="checkbox"/> Statement | <input checked="" type="checkbox"/> Keyboard Execution |
| <input type="checkbox"/> Function | <input type="checkbox"/> CALC Mode |
| <input type="checkbox"/> Operator | <input checked="" type="checkbox"/> IF...THEN...ELSE |
| | <input type="checkbox"/> Device Operation |



Examples

REMOTE @ LOCAL LOCKOUT

Makes the system Remote enabled, then sets all devices to Local Lockout condition.

IF I=10 THEN LOCAL LOCKOUT

If I = 10 and if system is Remote enabled, sets all devices to Local Lockout condition.

Operation

The LOCAL LOCKOUT statement provides a way to prevent devices from responding to their “local” controls, such as keys or switches. In the Local Lockout condition, the device can be set to respond to instructions received from the HP-IL controller only. This condition is useful for preventing an operator from inadvertently changing a setting manually at a critical time.

The Local Lockout condition can exist only when the system is Remote enabled. This means that devices are able to operate in Remote mode, in which they receive instructions from a “remote” source (the HP-IL controller). If the system is not Remote enabled, LOCAL LOCKOUT has no effect. Therefore, you should execute a REMOTE statement before executing LOCAL LOCKOUT.

While in the Local Lockout condition, an HP-IL device can change between Local mode and Remote mode in response to LOCAL *device* and REMOTE *device*. This enables the device’s controls to be used when it’s in Local mode (Remote enabled), but locks out the controls when it changes to Remote mode (such as when it’s next instructed to receive data).

To cancel a Local Lockout condition, you should execute LOCAL or LOCAL LOOP, which makes the system not Remote enabled. (Note that LOCAL *device* won’t cancel the Local Lockout condition.)

LOCAL LOCKOUT has no effect on the HP-71, including while it’s operating as an HP-IL device.

LOCAL LOCKOUT (continued)

HP-IL Messages

Local Lockout.

Related Keywords

LOCAL, REMOTE, STATUS.

OFF INTR

Cancels HP-IL interrupt branching.

- | | |
|---|--|
| <input checked="" type="checkbox"/> Statement | <input checked="" type="checkbox"/> Keyboard Execution |
| <input type="checkbox"/> Function | <input type="checkbox"/> CALC Mode |
| <input type="checkbox"/> Operator | <input checked="" type="checkbox"/> IF...THEN...ELSE |
| | <input checked="" type="checkbox"/> Device Operation |



Examples

`OFF INTR`

Disables HP-IL interrupt branching.

`IF S THEN OFF INTR`

If S is true, disables HP-IL interrupt branching.

Operation

The `OFF INTR` statement cancels HP-IL interrupt branching. Even though an `ENABLE INTR` statement may have defined certain events as interrupt events, `OFF INTR` cancels branching caused by any of those events.

`OFF INTR` doesn't prevent an interrupt event from occurring, nor does it prevent the HP-71 from remembering that it occurred—it only prevents branching from happening. (Refer to `ON INTR` for more information about the relation between `OFF INTR` and `ON INTR`.)

HP-IL Messages

None.

Related Keywords

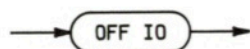
`ENABLE INTR`, `ON INTR`, `READINTR`.

OFF IO

Suspends HP-71 I/O operation.

- Statement
- Function
- Operator

- Keyboard Execution
- CALC Mode
- IF...THEN...ELSE
- Device Operation



Examples

`OFF IO`

Suspends I/O operation.

`IF BIT(STATUS,5) THEN OFF IO`

If HP-71 isn't controller, then suspends I/O operation.

Operation

The `OFF IO` statement suspends all I/O operation by the HP-71. In this condition, operations that would normally use the `DISPLAY IS` device or `PRINTER IS` device produce their output on the HP-71 display instead. Any other I/O operation that would normally use HP-IL devices isn't allowed in this condition—it results in an error condition.

For example, you might want to suspend I/O operation if you're checking a program that includes print and display operations, but you don't have these devices connected in the loop. Or you may have a program that must refrain from using I/O operations until a certain condition exists.

While I/O operation is suspended, the HP-71 maintains HP-IL operating information. Specifically, the HP-71 retains:

- The `DISPLAY IS` definition as you last specified it—either linked to an address or linked to a device characteristic.
- The `PRINTER IS` definition as you last specified it—either linked to an address or linked to a device characteristic.
- The `ASSIGN IO` definitions as you last specified them—each assign code associated with an HP-IL address.

I/O operation is restored by the `RESTORE IO` statement. Executing `ASSIGN IO` also restores operation.

OFF IO (continued)

HP-IL Messages

None.

Related Keywords

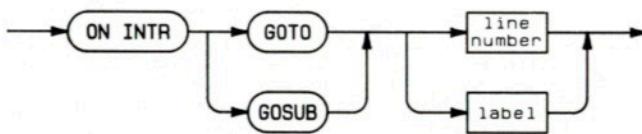
RESTORE IO.

ON INTR

Defines how a program branches when an enabled HP-IL interrupt event occurs.

- Statement
- Function
- Operator

- Keyboard Execution
- CALC Mode
- IF...THEN...ELSE
- Device Operation



Examples

```
ON INTR GOTO 5000
```

Defines direct branching to line 5000 when an enabled HP-IL interrupt event occurs.

```
ON INTR GOSUB ISUB
```

Defines subroutine branching to label ISUB when an enabled HP-IL interrupt condition occurs.

Input Parameters

Item	Description	Restrictions
line number	Four-digit integer constant.	1 through 9999.
label	String constant (or unquoted string) starting with a letter and having up to eight letters and digits.	None.

Operation

The `ON INTR` statement defines how a program branches when an enabled HP-IL interrupt event occurs. The HP-71 checks for interrupt events only at the end of each program line—the interrupt branching is called “end-of-line” branching.*

* The HP-71 actually checks for “end-of-line” branching immediately before the first statement of each program line, rather than after the last statement of each line. This distinction matters only if execution continually branches to labels *not* at the beginning of program lines.

ON INTR (continued)

If the `GOTO` option is used, program execution jumps to the specified line when an interrupt event occurs, and execution continues from there. If the `GOSUB` option is used, program execution jumps to the specified line when an interrupt event occurs—when the next `RETURN` statement is reached, execution returns to the statement after the one during which the interrupt event occurred.

The HP-IL events that are *enabled*—those that can cause interrupt branching—are defined by the `ENABLE INTR` statement. `ENABLE INTR` uses an *interrupt mask* to define which of the eight events are enabled. If no events are enabled, no branching can occur.

Branching that is defined by the `ON INTR` statement is cancelled by the `OFF INTR` statement.

An `ON INTR` statement defines branching only within the program containing the `ON INTR` statement. If the program stops or ends, interrupt branching is disabled. If the program calls a subprogram, interrupt branching is disabled while the subprogram is running. (Of course, the subprogram can contain its own `ON INTR` statement—interrupt events are still enabled according to the most recent `ENABLE INTR` statement.)

The following topics describe how interrupts are processed.

Interrupt-Cause Byte. The HP-71 doesn't ignore HP-IL events that might cause an interrupt. That is, it watches for any of the eight HP-IL events that can be enabled (refer to `ENABLE INTR`). If one of these events occurs, the HP-71 sets the corresponding bit in the *interrupt-cause byte*. The interrupt-cause byte is cleared by only two operations: `READINTR` and `RESET HPIL`.

There is only one case in which the HP-71 ignores one type of interrupt event. If a service request causes interrupt branching, the interrupt-cause byte *doesn't* reflect subsequent service requests until the next `ENABLE INTR` statement is executed. This prevents false indications of new service request conditions during a serial poll or parallel poll.

Processing an Interrupt. The HP-71 processes HP-IL interrupts according to the following procedures:

- If interrupt branching has been set (by an `ON INTR` statement) and an interrupt event occurs, the interrupt-cause byte is compared with the interrupt mask. If any enabled event has occurred, the mask is set to 0 and the interrupt branch occurs (as soon as the current program line has finished executing). The interrupt mask is set to 0; the interrupt-cause byte isn't changed.
- If interrupt branching has been set and the interrupt mask is redefined (by an `ENABLE INTR` statement), it is immediately compared with the latest interrupt-cause byte. If any enabled event has occurred, branching occurs as described above.
- If interrupt branching has been cancelled (by an `OFF INTR` statement or by no `ON INTR` statement) and an interrupt event occurs, the interrupt-cause byte is updated—but no branching can

ON INTR (continued)

occur. If a subsequent `ON INTR` statement is executed, the latest interrupt-cause byte is compared with the interrupt mask, and branching may occur as described above.

These operations result in two recommendations for a subroutine that responds to HP-IL interrupts:

- Use `READINTR` to clear the interrupt-cause byte at the start of the subroutine. If this byte isn't cleared, a subsequent `ENABLE INTR` statement may cause unexpected branching. Any event that occurs *during* the subroutine's execution will be saved in the interrupt-cause byte.
- End the subroutine with `ENABLE INTR mask @ RETURN` (if you want to enable interrupt events). (The mask was set to 0 when the interrupt branch occurred.) If the two statements are put on the same line, no interrupt branch can occur until after `RETURN` has been executed, even if an interrupt is pending.

Interaction With Other Interrupts. Because the HP-71 can process several kinds of interrupts (timer, error, and HP-IL), it's important to understand how these interrupts can interact.

First, if an interrupt occurs while the HP-71 is already executing an interrupt subroutine, the program branches out of the first interrupt subroutine according to the new interrupt. To avoid unexpected branching, disable other interrupts at the beginning of each interrupt subroutine (using `OFF TIMER`, `OFF ERROR`, or `OFF INTR`). Then enable them at the end (using `ON TIMER`, `ON ERROR`, or `ON INTR`).

Second, if more than one type of interrupt is pending when a program statement or line is completed, the HP-71 takes *one* branch according to the following precedence:

1. `ON ERROR` (checked at end of every *statement*).
2. `ON INTR` (checked at end of every *line*).
3. `ON TIMER` (checked at end of every *statement*).

This means that if two interrupt events have occurred when a program statement or line is finished executing, only the higher-precedence branch occurs *at that time*. But as soon as the next program statement or line finishes, the next-higher branch occurs if it is still enabled. Again, you can avoid unexpected branching by disabling other interrupts at the beginning of each interrupt subroutine—this effectively cancels other pending interrupts.

HP-IL Messages

None.

Related Keywords

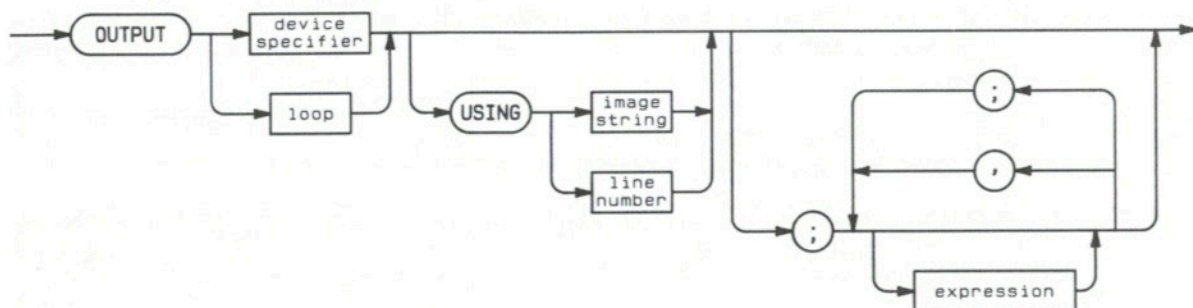
`ENABLE INTR`, `OFF INTR`, `READINTR`.

OUTPUT

Sends data from numeric and string expressions to HP-IL.

- Statement
- Function
- Operator

- Keyboard Execution
- CALC Mode
- IF...THEN...ELSE
- Device Operation



Examples

```
OUTPUT 2; "MOVE";N
```

Sends to device at address 2 a sequence of characters representing the string "MOVE" and the number stored in N.

```
OUTPUT INTRFCE(2); A1,A2
```

Sends to second interface device a sequence of characters representing the numbers stored in A1 and A2.

```
OUTPUT "HP82164A" USING "80A";  
X$,Y$
```

Sends to the first HP-IL/RS-232 interface (device ID "HP82164A") two 80-character sequences of characters representing the strings stored in X\$ and Y\$.

```
OUTPUT 3 USING "#,B"; I
```

Sends to device at address 3 one byte having the value I.

```
OUTPUT LOOP; B1$
```

Sends to HP-IL (device not specified) a sequence of characters representing the string stored in B1\$.

OUTPUT (continued)

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.
loop	See standard definition.	None.
format string	String expression that specifies a valid sequence of image symbols.	None.
line number	Four-digit integer constant that refers to <code>IMAGE</code> statement.	1 through 9999.
expression	Numeric or string expression.	None.

Operation

The `OUTPUT` statement sends numeric or string data to HP-IL devices. The data is sent as a sequence of characters according to a format that is specified or implied by the `OUTPUT` statement. If no format is specified, then data is sent using “free-field” output; if `USING` is specified, the data is sent according to the format specified by the format string or `IMAGE` statement. Detailed information about these two forms of output are provided under the last two headings below.

The `OUTPUT` statement finishes executing when all of the specified data is sent. The data is followed by one of these sequences of characters:

- The end-of-line sequence defined by the `ENDLINE` statement—which is usually a carriage return character and a line feed character (character codes 13 and 10). This sequence is sent unless specifically deleted by the methods listed next.
- No additional characters. This occurs for free-field output if the list of output data ends with a semicolon (;) or comma (,); it occurs for formatted output if the format specifically suppresses the end-of-line sequence.

Notice that `OUTPUT` *can't* terminate with an HP-IL End Byte message, a special form of data byte that indicates the end of a logical group of data.

The `LOOP` option provides the ability to send data in two special situations:

- The HP-71 has already prepared one or more devices to receive data by making them listeners, and the HP-71 is already designated to send data (it is a talker). `OUTPUT LOOP` causes the HP-71 to send its data to all listeners simultaneously. (Refer to the `SEND` statement.)
- The HP-71 is operating as an HP-IL device. It can't specify the device that is to receive data—the controller defines the receiver.

OUTPUT (continued)

If the HP-71 is operating as an HP-IL device, it *must* use the `LOOP` option. The HP-71 can send data on HP-IL only after the controller makes it a talker. Program execution is affected by the order of events:

- If the HP-71 is instructed by the controller to send data on HP-IL before it executes `OUTPUT`, the HP-71 holds up loop operation until it executes `OUTPUT` (or any other operation that sends data on HP-IL).
- If the HP-71 executes `OUTPUT` before it is instructed by the controller to send data on HP-IL, it will send and store up to 65 characters in its HP-IL output buffer. These characters are transmitted when the HP-71 is next instructed to send data. If the HP-71 tries to send more than 62 characters, the program waits at the `OUTPUT` statement until the HP-71 is instructed to send data on HP-IL. (`RESET HPIL` clears the output buffer.)

Advanced User's Note: If the HP-71 is operating as an HP-IL device, it expects an `OUTPUT` operation to be terminated by an HP-IL Not Ready For Data message from the receiver. (For example, the HP-71 `ENTER` statement automatically sends this message when the required data has been received.) The HP-71 doesn't automatically send an HP-IL End Of Transmission message after all data has been sent by an `OUTPUT` statement—it sends this message only in response to a Not Ready For Data message.

Free-Field Output. For free-field output, numeric data is treated differently from string data.

Each numeric value is sent as a sequence of characters that “write” the number using the current round-off setting: a leading sign character (space or minus sign) and as many additional characters as required by the round-off setting. The additional characters may be digits (0 through 9), decimal point, exponent indicator (`E`), and negative exponent sign (minus sign). (The current round-off setting is defined by `STD`, `FIX`, `SCI`, or `ENG`.)

String data can be *any* sequence of characters—it's sent “as is.”

The spacing following an output item in free-field output depends upon the punctuation that follows that item in the `OUTPUT` statement. This provides some flexibility for output formatting without using an image string.

- Semicolon (`;`)

Numeric data has one trailing space (for separation).

String data has no trailing spaces.

OUTPUT (continued)

- Comma (,)

Numeric data has trailing spaces as required to make the output a total of 20 characters long (for alignment), plus one additional trailing space (for separation).

String data has up to 20 trailing spaces as required to make the output string a multiple of 20 characters long (for alignment), plus one additional trailing space (for separation).

Remember that numeric data always starts with a space or minus sign.

For example, suppose the HP-71 were sending $X = 14.55$ and $A\# = \text{"THIS IS A TEST"}$ and $Y = -2.6666666667 \times 10^{27}$ using the STD round-off setting. These values could be sent in these ways:

```
OUTPUT 1;X,A#,Y
14.55          THIS IS A TEST          -2.66666666667E27

OUTPUT 1;X,A#;Y
14.55          THIS IS A TEST-2.66666666667E27

OUTPUT 1;X;A#;Y
14.55 THIS IS A TEST-2.66666666667E27
```

A semicolon or comma after the last item in the OUTPUT list has a special effect. First, it adds trailing spaces as described above. Then, it suppresses the end-of-line sequence normally sent at the end of the OUTPUT statement. (The sequence is defined by the ENDLINE statement.) If there is no final semicolon or comma, trailing spaces are added as for a semicolon, then the end-of-line sequence is sent.

Formatted Output. By specifying a format in an OUTPUT statement, you obtain a high degree of control over how the output data is structured when it's sent, including the use of end-of-line sequences.

The output format can be specified as a format string in the OUTPUT statement; otherwise, it must be specified as a line number that references an IMAGE statement.

The format string itself must be made up of legitimate *image symbols*, which are described in the table below. One or more symbols form an *image field*, which corresponds to one output item. Certain symbols may be replicated, as indicated in the table—such a symbol may be preceded by a number that specifies how many times that symbol should be used, or repeated, within a field.

OUTPUT (continued)

Image Symbols for OUTPUT

Image Symbol	Type of Output	Meaning	May Be Replicated
Numeric Input:			
D	Digit	Sends one digit character of a number. If the digit is a leading zero, it's replaced by a space. If the number is negative and "S" or "M" isn't included for the minus sign, the last leading zero is replaced by the minus sign.	Yes
Z	Digit	Sends one digit character of a number. If the digit is a leading zero, it's sent as a zero digit. If the number is negative and "S" or "M" isn't included for the minus sign, the first leading zero is replaced by the minus sign. (This can't be used after the radix point.)	Yes
*	Digit	Sends one digit character of a number. It operates just like "Z" except that a leading zero is sent as "*". (This can't be used after the radix point.)	Yes
.	Decimal	Sends a decimal point radix.	No
S	Sign	Sends a "+" or "-" as the number's sign.	No
M	Sign	Sends a space or "-" as the number's sign.	No
E	Exponent	Sends an "E", a "+" or "-", and three digits as the number's exponent. Leading zeros of the exponent are sent.	No
C	Comma	Sends a comma as a digit separator.	No
R	Comma Radix	Sends a comma as the radix point. This is useful for European formatting.	No
P	Period Separator	Sends a period as a digit separator. This is useful for European formatting.	No
K	Compact	Sends an entire number using the current round-off setting. It sends no leading or trailing spaces and no leading zeros. If the number is negative, it sends a leading minus sign.	No
H	Compact	Sends an entire number using the current round-off setting. It operates just like "K" except that a comma is used for the radix point. This is useful for European formatting.	No

OUTPUT (continued)

Image Symbols for OUTPUT (continued)

Image Symbol	Type of Output	Meaning	May Be Replicated
^	Hide	Hides an entire number. The output item is evaluated, but it isn't sent.	No
String Output:			
A	Character	Sends one character of a string. If the entire string has been sent, it sends a space.	Yes
K H	Compact	Sends all characters of a string. It sends no leading or trailing spaces.	No
^	Hide	Hides all characters of a string. The output item is evaluated, but it isn't sent.	No
Binary Output:			
B	Byte	Sends one character according to the value of the numeric output item. The value is rounded to an integer, then is folded into the range 0 through 255 (using modulo 256). (This is equivalent to CHR\$.)	No
Editing Symbols:			
X	Blank	Sends a space. This can be used within a numeric or string field or as a separate field.	Yes
"..." '...'	Quotes	Sends the keyboard characters included within the quotes. This can be used within a numeric or string field or as a separate field.	Yes

OUTPUT (continued)

Image Symbols for OUTPUT (continued)

Image Symbol	Type of Output	Meaning	May Be Replicated
Special Symbols:			
,	Field Separator	Defines the end of an image field.	No
@	Form Feed	Defines the end of an image field and sends a form feed character (character code 12).	Yes
/	Line Ending	Defines the end of an image field and sends an end-of-line sequence. The sequence is determined by <code>ENDLINE</code> .	Yes
<...>	Group	Defines an image "group", which is a sequence of one or more image fields. This is useful for repeating one or more fields.	Yes
#	Terminator	Suppresses sending the end-of-line sequence at the end of the <code>OUTPUT</code> operation. If used, it must be the first field of the image string.	No

Commas, slashes, and "@" signs are used to separate fields. They indicate the end of an image field for an output expression.

Consider the following examples of format strings. (Note that a format string can be included in quotes in the `OUTPUT` statement instead of using an `IMAGE` statement.)

```
IMAGE DDDDDDD.DD,40A/10A
```

Sends a number with a comma as digit separator and two decimal places, a string of 40 characters, an end-of-line sequence, and a string of 10 characters.

```
IMAGE 5D,2X,4DE
```

Sends a number with five digits, two spaces, and a number with four digits plus exponent field.

```
IMAGE Z.DD,K
```

Sends a number using leading zero and two decimal places, and a variable in compact format.

```
IMAGE #,B
```

Sends one value as a binary byte with no end-of-line sequence after.

```
IMAGE 6<DDXDDD.D,3X,  
'$'DDZ.DD/>@
```

Sends six pairs of numbers followed by a form feed character. First number has space for digit separator; second number preceded by "\$". Each pair is on separate line.

OUTPUT (continued)

If the `OUTPUT` statement hasn't terminated by the time the HP-71 has stepped all the way through the format string, the same format string is used again. Essentially, the format string repeats until the `OUTPUT` statement finishes.

If a number has more digits to the left of the decimal point than provided by the image field, an *overflow* occurs. This condition is reported in the same way as math overflows: an error or warning is given. (Refer to `TRAP` and `DEFAULT` in the *HP-71 Reference Manual*.) If a warning is given, the number is sent as a field of “*” symbols. Note that a leading minus sign requires a position in the output—an overflow occurs if the minus sign doesn't fit.

Refer to `IMAGE` in the *HP-71 Reference Manual* for additional information about image strings.

HP-IL Messages

Data Byte.

Related Keywords

`ENTER`, `SEND`.

PACK

Packs directory and storage space on a medium.

- ☒ Statement
- ☐ Function
- ☐ Operator
- ☒ Keyboard Execution
- ☐ CALC Mode
- ☒ IF...THEN...ELSE
- ☐ Device Operation



Examples

PACK %16

Packs the medium in the first HP 82161A Digital Cassette Drive (accessory ID 16).

IF V THEN PACK TAPE(2)

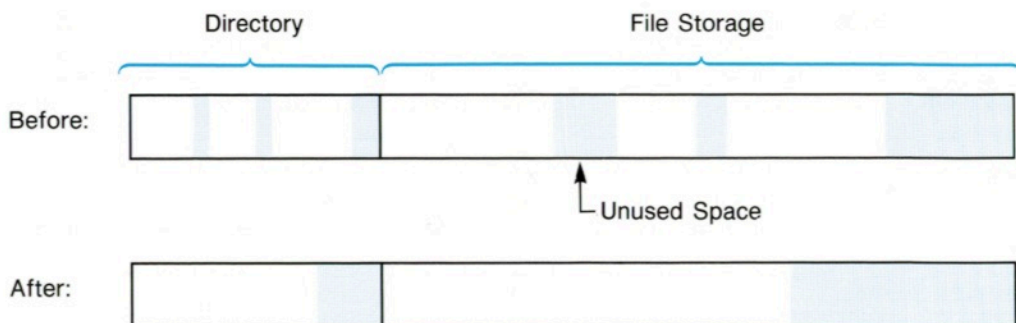
If V is true, packs the medium in the second mass storage device.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.

Operation

The **PACK** statement provides the maximum amount of usable free space on a medium. It does this by deleting all gaps in the directory and in the file storage area—it moves all valid directory information and all valid data information toward the beginning of those areas. **PACK** eliminates gaps caused by purging and changing files.

PACK (continued)

The specified device must be a *file-oriented* device.

Note: Executing **PACK** causes each record to be read and re-stored on the medium. This requires that the medium be positioned to many different locations, possibly resulting in a reduction in its usable life, especially for tape cassettes. It is recommended that you avoid the casual or unnecessary use of **PACK**.

HP-IL Messages

None.

Related Keywords

CAT, PACKDIR, PURGE.

PACKDIR

Packs only directory space on a medium.

<input checked="" type="checkbox"/> Statement	<input checked="" type="checkbox"/> Keyboard Execution
<input type="checkbox"/> Function	<input type="checkbox"/> CALC Mode
<input type="checkbox"/> Operator	<input checked="" type="checkbox"/> IF...THEN...ELSE
	<input type="checkbox"/> Device Operation



Examples

```
PACKDIR :MASSMEM(3)
```

Packs the directory of the medium in the third mass storage device.

```
IF M=10 THEN PACKDIR 1
```

If M = 10, packs the directory of the medium in the device at address 1.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.

Operation

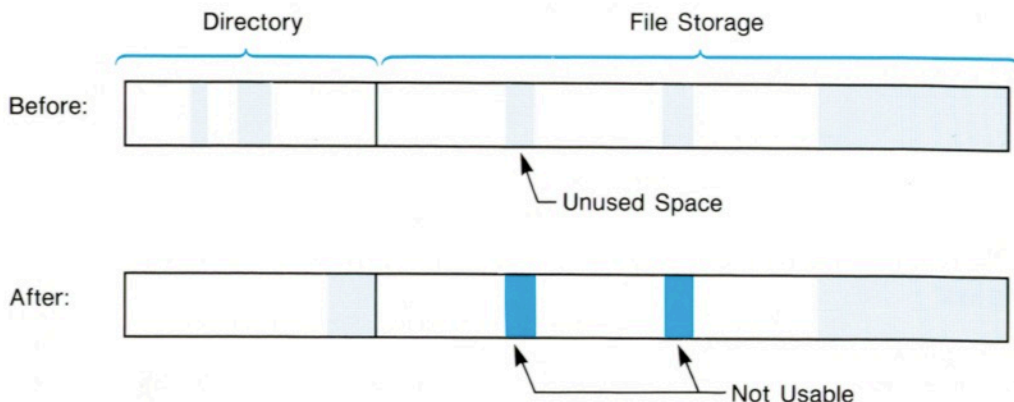
The PACKDIR statement eliminates gaps in a medium’s directory, providing space for additional directory entries after the last entry.

A medium’s directory contains entries that correspond *in order* to the files stored on the medium. When a file is purged, a gap is formed in both the directory and the file storage area. These gaps will be used later for storing a file that fits in the file storage gap—otherwise, the gaps remain unused.

For example, if you significantly lengthen a file and re-store it on the medium, the original file is purged and the longer version is stored in a new location (in both the directory and the file storage area). The gaps that result may be used later to store another file that fits in the file storage gap.

PACKDIR (continued)

The `PACKDIR` statement is useful when the directory is “full” (the last entry space has been used) but the file storage area still has unused space at the end—especially if the gaps that exist in the file storage space are too short for your needs. (Without executing `PACKDIR`, the unused file storage space at the end can’t be used because no directory entry can be created for it.) However, `PACKDIR` eliminates the directory gaps (the corresponding file storage gaps become unusable—until you execute `PACK`). The unused entry spaces created at the end of the directory enable you to use the file storage area at the end of the medium.



The specified device must be a *file-oriented* device.

Note: Executing `PACKDIR` causes each record in the directory to be read and re-stored. This requires that the medium be positioned to many different locations within the directory, possibly resulting in a reduction in its usable life, especially for tape cassettes. It is recommended that you avoid the casual or unnecessary use of `PACKDIR`.

`PACKDIR` executes in a significantly shorter time than `PACK`, which packs both the directory and the file storage area.

HP-IL Messages

None.

Related Keywords

CAT, PACK, PURGE.

PASS CONTROL

Gives control of the HP-IL system to another device, causing the HP-71 to operate as an HP-IL device (*not* as the controller).

- ☒ Statement

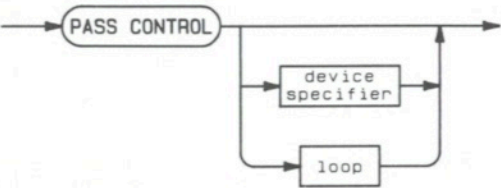
☐ Function

☐ Operator
- ☒ Keyboard Execution

☐ CALC Mode

☒ IF...THEN...ELSE

☐ Device Operation



Examples

PASS CONTROL :HP71

Gives control of the loop to the next HP-71 (device ID “HP71”) in the loop.

PASS CONTROL LOOP

Gives control of the loop to whatever device is a talker. (The device isn’t specified by this statement.)

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.
loop	See standard definition.	None.

Operation

The PASS CONTROL statement gives control of the HP-IL system to another device. It provides a way to transfer in a coordinated manner the responsibility of being the HP-IL *controller*. That is, the HP-71 gives up control at the same time that the new controller takes control. Then, the HP-71 begins operating as an HP-IL device under control of the new controller.

PASS CONTROL (continued)

If a device is specified, `PASS CONTROL` provides the complete transaction needed to make that device the controller.

If no device is specified, or if the `LOOP` option is used, `PASS CONTROL` doesn't prepare any device to receive control—that must be done by a previous operation. (To receive control, a device must first be designated as a *talker*—this can be done using the `SEND` statement while no `DISPLAY IS` device is assigned.)

The `PASS CONTROL` statement doesn't finish executing until one of two conditions occurs:

- A device accepts control, and this is indicated to the HP-71. This happens when the new controller sends any HP-IL message (other than a Take Control message, which is used by `PASS CONTROL`). In this situation, the HP-71 begins operating as an HP-IL device under the control of the new controller.
- No device accepts control, and this is indicated to the HP-71. This happens when the Take Control message travels around the loop and returns to the HP-71 without being intercepted by the new controller. In this situation, an error occurs and the HP-71 retains control of the HP-IL system.

Note: The HP-IL capabilities that the HP-71 can exercise while it is *not* the controller are indicated in this keyword dictionary by a solid box next to "Device Operation." Refer to appendix B for detailed information about how the HP-71 responds as a device.

When the HP-IL interface is first installed, the HP-71 will be operating as *controller* of the HP-IL system. Similarly, whenever the HP-IL interface is reset (using `RESET HPIL`), the HP-71 automatically becomes the HP-IL controller.

HP-IL Messages

Talk Address (except for `LOOP`), Take Control.

Related Keywords

`CONTROL OFF/ON`, `REQUEST`.

PRINTER IS

Assigns one HP-IL device to be used for all printing operations.

■ Statement

□ Function

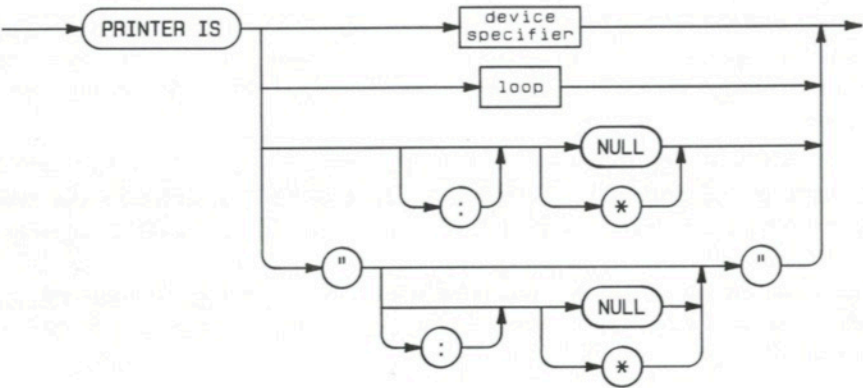
□ Operator

■ Keyboard Execution

□ CALC Mode

■ IF...THEN...ELSE

■ Device Operation



Examples

- PRINTER IS PRINTER(2)

Assigns the second printer device to be the PRINTER IS device.
- PRINTER IS *

Makes the PRINTER IS device be the same as the DISPLAY IS device.
- PRINTER IS NULL

Assigns no PRINTER IS device.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.
loop	See standard definition.	None.

PRINTER IS (continued)

Operation

The `PRINTER IS` statement assigns one HP-IL device as the “printer” device. All subsequent printed output goes to that device. Printed output is generated by the `PRINT` and `PLIST` statements and is formatted according to the line length specified by `PWIDTH`.

Whenever a memory reset condition occurs in the HP-71, the HP-71 automatically performs `PRINTER IS PRINTER`, which sets the first printer device in the loop as the `PRINTER IS` device.

Assigning `*` as the printer device makes the `PRINTER IS` device be the same device as the `DISPLAY IS` device. If you change the `DISPLAY IS` device, the `PRINTER IS` device automatically changes also.

Assigning `NULL` as the printer device assigns *no* `PRINTER IS` device. In this situation, all printed output is lost—it’s sent nowhere.

The HP-71 assumes that the `PRINTER IS` device is a character-oriented device—that is, one that normally operates with individual characters or lines of characters. Examples are printers and video interfaces. You should use care when trying to use a file-oriented device (such as a mass storage device) for receiving printed output—the results may be unpredictable.

The `LOOP` option provides the ability to assign an “unspecified” `PRINTER IS` device in two special situations:

- The HP-71 will prepare one or more devices to receive printed data by making them listeners, and the HP-71 will be designated to send data (it will be a talker). A printing operation will cause the HP-71 to send its data to all listeners simultaneously. (Refer to the `SEND` statement.)
- The HP-71 will operate as an HP-IL device. It can’t specify the device that is to receive data—the controller defines the receiver.

PRINTER IS (continued)

If the HP-71 is operating as an HP-IL device and you want to send printed output, you *must* use the `LOOP` option. The HP-71 can send printed data on HP-IL only after the controller makes it a talker. Program execution is affected by the order of events:

- If the HP-71 is instructed by the controller to send data on HP-IL before it executes a printing operation, the HP-71 holds up loop operation until it executes the printing operation (or any other operation that sends data on HP-IL).
- If the HP-71 executes a printing operation before it is instructed by the controller to send data on HP-IL, it will send and store up to 65 characters in its HP-IL output buffer. These characters are transmitted when the HP-71 is next instructed to send data. If the HP-71 tries to send more than 62 characters, the program waits at the printing statement until the HP-71 is instructed to send data on HP-IL. (`RESET HPIL` clears the output buffer.)

Advanced User's Note: If the HP-71 is operating as an HP-IL device, it expects a printing operation to be terminated by an HP-IL Not Ready For Data message from the receiver. (For example, the HP-71 `ENTER` statement automatically sends this message when the required data has been received.) The HP-71 doesn't automatically send an HP-IL End Of Transmission message after all data has been sent by a printing operation—it sends this message only in response to a Not Ready For Data message.

If the HP-71 is operating as an HP-IL device, the only additional `PRINTER IS` operations it can perform are `PRINTER IS *`, which directs printed output to the HP-71 display, and `PRINTER IS NULL`, which cancels printing operations.

Related Keywords

`DISPLAY IS, OUTPUT.`

Permanently prevents a file from being changed or inspected.

☒ Statement

☐ Function

☐ Operator

☒ Keyboard Execution

☐ CALC Mode

☒ IF...THEN...ELSE

☐ Device Operation



Examples

- `PRIVATE PRGMA:2`

Makes private the file PRGMA in the device at address 2.
- `PRIVATE "FILE5.EXP1"`

Makes private the file FILE5 on the medium with volume label EXP1.

Input Parameters

Item	Description	Restrictions
file specifier	See standard definition.	None.

Operation

The PRIVATE statement provides permanent privacy for a program type file (such as a BASIC, binary, or LEX file). This means that the file can't be changed or inspected—but it *can* be executed, and it can be copied to main RAM only (as a private file).

PRIVATE allows a program file to be executed, but not inspected or duplicated. A private file can also be renamed, secured, unsecured, and purged.

Note: As an added precaution, only an unsecure file can be made private.

PRIVATE (continued)

A private file is displayed in a catalog listing with a “P” in the protection field. If a private file is also secure (using `SECURE`), it has an “E” (for “execute only”) in the protection field. So a private file can have only “P” or “E” in the protection field of a catalog listing.

Refer to the *HP-71 Reference Manual* for information about using files in main RAM (or independent RAM).

Related Keywords

`SECURE`, `UNSECURE`.

PURGE

Deletes a file (if it's not secure).

- Statement
- Function
- Operator

- Keyboard Execution
- CALC Mode
- IF...THEN...ELSE
- Device Operation



Examples

```
PURGE BACKUP:MASSMEM(2)
```

Purges the file BACKUP in the second mass storage device.

```
IF F$=A$ THEN PURGE A$
```

If F\$ equals A\$, purges the file specified by A\$.

Input Parameters

Item	Description	Restrictions
file specifier	See standard definition.	None.

Operation

The PURGE statement deletes an unsecure file from a mass storage medium. It does this by marking the directory entry to indicate a purged file. The directory entry and file information aren't actually erased, although the file is *not* retrievable.

The specified file must exist in the device.

If a file is secure, it can be purged only after you make it unsecure by executing UNSECURE.

Refer to the *HP-71 Reference Manual* for information about using files in main RAM (or independent RAM).

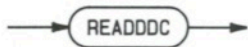
Related Keywords

CAT, CREATE, PACK, PACKDIR, SECURE, UNSECURE.

READDDC

Returns the number of the last HP-IL device-dependent command message received.

- | | |
|--|--|
| <input type="checkbox"/> Statement | <input checked="" type="checkbox"/> Keyboard Execution |
| <input checked="" type="checkbox"/> Function | <input checked="" type="checkbox"/> CALC Mode |
| <input type="checkbox"/> Operator | <input checked="" type="checkbox"/> IF...THEN...ELSE |
| | <input checked="" type="checkbox"/> Device Operation |



Examples

```
X=READDDC
```

Sets *X* to the number of the last device-dependent message.

```
IF BIT(READINTR,0) THEN
  A=READDDC
```

If bit 0 of the interrupt-cause byte is true (meaning a device-dependent message was received), sets *A* to the number of the last device-dependent message.

Operation

The READDDC function returns the number of the last device-dependent command message received by the HP-71. This function also clears the place in memory where this number is stored.

Device-dependent command messages are HP-IL messages that have no universal meanings—the response to these messages varies from device to device, according to the device's design. Device-dependent command messages provide a way to customize communication with a device—to provide a specialized set of instructions for each device that needs it.

Device-dependent command messages are separated into two groups:

- Device Dependent Talker messages, numbered from 0 through 31. These messages are received by a device only if the device is a talker. They often affect the type of data that the device sends.
- Device Dependent Listener messages, numbered from 0 through 31. These messages are received by a device only if the device is a listener. They often affect how a device interprets data that it receives.

READDDC (continued)

READDDC distinguishes between the two device-dependent message groups by offsetting the numbers of Device Dependent Listener messages, as shown in the table below.

READDDC Values

READDDC Value	Device-Dependent Message
Device Dependent Talker Messages:	
0	Device Dependent Talker 0
1	Device Dependent Talker 1
.	.
.	.
.	.
31	Device Dependent Talker 31
Device Dependent Listener Messages:	
32	Device Dependent Listener 0
33	Device Dependent Listener 1
.	.
.	.
.	.
63	Device Dependent Listener 31

If no device-dependent message has been received since READDDC was last used or since the HP-IL module was last installed, READDDC returns a value of -1.

The HP-71 stores the number of only the last device-dependent message it receives. READDDC can therefore retrieve the number of only the last such message.

Receipt of a device-dependent message can be enabled as an HP-IL interrupt event (using ENABLE INTR).

READDDC is useful primarily while the HP-71 is operating as an HP-IL device. The HP-71 doesn't store the number of a device-dependent message sent to other devices. It stores the number of only a message that it's set to receive, such as while it's a listener or talker.

Two expressions may be useful for interpreting the READDDC value. (If you want to use the value more than once, you can save it in a variable.)

BIT(READDDC, 5)

Returns 0 for Device Dependent Talker message and 1 for Device Dependent Listener message.

MOD(READDDC, 32)

Returns the actual message number, 0 through 31.

READDDC (continued)

HP-IL Messages

None.

Related Keywords

ENABLE INTR, ON INTR, READINTR.

READINTR

Returns the value of the interrupt-cause byte, which indicates HP-IL interrupt events that have occurred, and then clears that byte.

- | | |
|--|--|
| <input type="checkbox"/> Statement | <input checked="" type="checkbox"/> Keyboard Execution |
| <input checked="" type="checkbox"/> Function | <input checked="" type="checkbox"/> CALC Mode |
| <input type="checkbox"/> Operator | <input checked="" type="checkbox"/> IF...THEN...ELSE |
| | <input checked="" type="checkbox"/> Device Operation |



Examples

```
I=READINTR
IF BIT(READINTR,4) THEN GOTO
  100
```

Sets I to value of interrupt-cause byte.

If bit 4 of the interrupt-cause byte is true (indicating service request received), branches to line 100.

Operation

The READINTR function returns the decimal value of the interrupt-cause byte and then clears that byte. The interrupt-cause byte stores information about HP-IL interrupt events that have occurred since READINTR was last executed.

The value of the interrupt-cause byte is a decimal value—when expressed in binary form, its eight bits correspond to the HP-IL interrupt events that have occurred. The READINTR value is the sum of the decimal values for the events that have occurred, as described in the following table. (Notice that certain events can occur while the HP-71 is a controller, and that others can occur while the HP-71 is a device.)

READINTR (continued)

Interrupt-Cause Byte

Bit	Decimal Value*	Interrupt Event	Controller†	Device†
7	128	Interface Clear. Received an Interface Clear message that HP-71 didn't send. (Normally clears controller, talker, and listener status.)	X	X
6	64	Listener. Made a listener by HP-IL controller. (Normally prepares device to receive data.)	(X)	X
5	32	Controller. Made active controller by previous controller. (Tells device to take control of HP-IL system.)		X
4	16	Talker. Made a talker by HP-IL controller and instructed to send data (by Send Data message). (Tells device to send data.)	(X)	X
3	8	Service Request. HP-71 detected service request bit set in HP-IL message. (Normally tells controller that a device needs attention.)	X	
2	4	Device Clear. Received a Device Clear or (while a listener) a Selected Device Clear message. (Normally resets device to operational startup conditions. Clears HP-IL input and output buffers in HP-71.)	X	X
1	2	Trigger. Received a Group Execute Trigger message while a listener. (Normally triggers an event at a device.)	X	X
0	1	Device Dependent. Received a Device Dependent Talker message while a talker or else received Device Dependent Listener message while a listener. (Meaning depends upon device.)	X	X

* Interrupt-cause byte is sum of values of all events that have occurred since READINTR last executed.

† An "X" indicates that the interrupt event can occur while the HP-71 is a controller or while the HP-71 is a device. An "(X)" indicates that the event can occur using SEND LISTEN 0 or SEND TALK 0 only.

The HP-71 always watches for the eight HP-IL events that might cause an HP-IL interrupt. Any time one of these events occurs, the HP-71 sets the corresponding bit in the interrupt-cause byte. The interrupt-cause byte is cleared by two operations: READINTR and RESET HPIL.

For example, if READINTR returns a value of 192, then the Interface Clear and Listener events have occurred. If READINTR is executed again (before another HP-IL event occurs), it returns a value of 0 because the interrupt-cause byte was cleared when it was read previously.

READINTR (continued)

The events indicated by the interrupt-cause byte are the same events (and in the same bit positions) as used by the `ENABLE INTR` mask.

`READINTR` must normally be the first statement executed in routines that process HP-IL interrupts. Refer to `ON INTR` for more information.

HP-IL Messages

None.

Related Keywords

`ENABLE INTR`, `OFF INTR`, `ON INTR`.

REMOTE

Enables all HP-IL devices to change to Remote mode and can also set an individual HP-IL device to Remote mode.

- Statement

□ Function

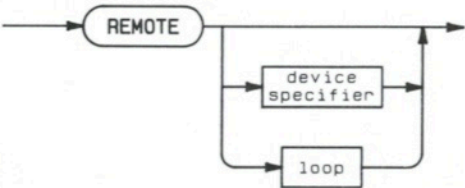
□ Operator

■ Keyboard Execution

□ CALC Mode

■ IF...THEN...ELSE

□ Device Operation



Examples

REMOTE

IF R THEN REMOTE "HP82164A"

Enables all HP-IL devices to change to Remote mode.

If R is true, sets HP-IL/RS-232 interface (device ID "HP82164A") to Remote mode.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.
loop	See standard definition.	None.

Operation

The REMOTE statement always enables all HP-IL devices to change to Remote mode. In addition, it can set an individual HP-IL device to Remote mode. Not all HP-IL devices implement Remote mode—for such devices, REMOTE has no effect. For a device that does implement Remote mode (and Local mode), its response in Remote mode depends upon its design. Typically, a device in Remote mode can respond to instructions received on HP-IL from a “remote” source, such as the HP-71.

REMOTE (continued)

Regarding Remote and Local modes, the HP-IL system can be in either of two conditions:

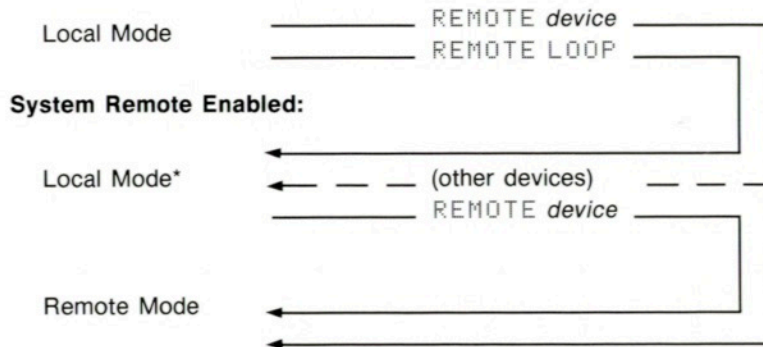
- Not Remote Enabled. Devices may not be in Remote mode—all devices must be in Local mode.
- Remote Enabled. Devices may be in Remote mode or in Local mode, but any device in Local mode automatically changes to Remote mode when it is next instructed to receive data (when it next becomes a listener).

The REMOTE statement has three forms that serve two purposes:

- REMOTE and REMOTE LOOP make the system Remote enabled, but don't change the mode of any device. If a device is in Local mode, it changes to Remote mode when it is next instructed to receive data (when it next becomes a listener).
- REMOTE *device* makes the system Remote enabled *and* puts one device into Remote mode.

The following diagram shows how the REMOTE statement affects a device's mode. It also shows how REMOTE affects the system's condition.

System Not Remote Enabled:



* Changes to Remote mode when next instructed to receive data.

If the HP-71 is operating as an HP-IL device, it can be set to Remote mode by the controller. While the HP-71 is in this mode and isn't busy (executing a program or INPUT statement, or in CALC mode), the HP-71 interprets data that it receives as BASIC instructions. It treats them as though they were entered from the keyboard. In this way, the controller may cause the HP-71 to perform BASIC operations.

Note: If you enter a REMOTE statement using incorrect syntax, it may not generate an error condition. Instead, it may be interpreted as a REM (remark) statement.

REMOTE (continued)

HP-IL Messages

Remote Enable, Listen Address (for *device* only).

Related Keywords

LOCAL, LOCAL LOCKOUT.

RENAME

Changes the name of a file.

☒ Statement
☐ Function
☐ Operator

☒ Keyboard Execution
☐ CALC Mode
☒ IF...THEN...ELSE
☐ Device Operation



Examples

```

RENAME "FILE08:TAPE" TO
"MARGIN"

```

Changes the name of file FILE08 in first mass storage device to MARGIN.

```

RENAME "POINTS" TO
N#&":TAPE(2)"

```

Changes the name of file POINTS in second mass storage device to name specified by N#.

Input Parameters

Item	Description	Restrictions
file specifier	See standard definition.	One specifier may omit the device specifier.

Operation

The **RENAME** statement changes the name of a specified mass storage file to a specified new name.

Only one file specifier must indicate the device that contains the file. If *both* specifiers indicate a device, **RENAME** uses the device indicated by the first specifier.

The new file name must not already exist in the device. If it's already used, the existing file is *not* affected in any way.

RENAME (continued)

Notice that RENAME can't duplicate a file—it affects only the directory entry.

Refer to the *HP-71 Reference Manual* for information about using files in main RAM (or independent RAM).

Related Keywords

CAT, COPY, PURGE.

REQUEST

Defines the HP-71 status byte that is sent when serially polled by an HP-IL controller (while operating as a device).

- | | |
|-------------|----------------------|
| ■ Statement | ■ Keyboard Execution |
| □ Function | □ CALC Mode |
| □ Operator | ■ IF...THEN...ELSE |
| | ■ Device Operation |



Examples

REQUEST 224

Defines the value 224 (a system status byte indicating “request control of loop”) to be sent as the HP-71 status byte when serially polled. Also requests service from controller.

IF FLAG(-61) THEN REQUEST 129

If flag -61 (BAT annunciator on) is true, defines the value 129 (a system status byte indicating “low battery”) to be sent as the HP-71 status byte when serially polled.

Input Parameters

Item	Description	Restrictions
status byte	Numeric expression, which is rounded to an integer.	0 through 255.

Operation

The **REQUEST** statement defines the HP-71 status byte. This status byte is used only when the HP-71 is operating as an HP-IL device. It’s sent when the HP-IL controller requests the status of the HP-71, such as when the controller is conducting a *serial poll*. A serial poll occurs when the controller requests the status of each device, one after another—serially.

Notice that **REQUEST** only *defines* the status byte—it doesn’t cause it to be sent.

REQUEST (continued)

Note: The HP-71 status byte defined by `REQUEST` is for sending to the HP-IL controller by the HP-71 while it's a device (it's also for requesting service). It's analagous to (but differs from) the device status (an external status) returned by `SPOLL` to the HP-71 while it's the controller. It also differs from the HP-IL interface status (an internal status) returned by `STATUS`.

The HP-71 status byte defined by `REQUEST` remains active until changed by one of the following events:

- It's redefined by another `REQUEST` statement.
- It's defined as 0 by a `RESET HPIL` statement.

Although `REQUEST` can define the status byte to be any valid value, you may find it convenient to follow established HP-IL status conventions. (This is especially important if you're interested in compatibility with other HP-IL equipment.) If the status byte is expressed as an eight-bit binary number, the bits have the following meanings:

- Bit 7 indicates the type of status byte.
 - “0” Device status.
 - “1” System status.
- Bit 6 determines if the HP-71 requests service from the controller. (Refer to “Sending Service Requests” below.)
 - “0” Service is not requested.
 - “1” Service is requested.
- Bits 5 through 0 indicate the condition of the HP-71.

The state of bit 7 (the most significant bit) defines two ranges of status bytes:

- System status, which can have values from 128 through 255. The conditions indicated by bits 5 through 0 have been defined by HP-IL conventions.
- Device status, which can have values from 0 through 127. The conditions indicated by bits 5 through 0 are *not* defined in general—they may be defined according to your needs.

`REQUEST` can be executed while the HP-71 is operating as a controller—it merely defines the HP-71 status byte. But the HP-71 can't use this status byte until it begins operating as an HP-IL device, as mentioned above.

System Status Byte. The meaning of a system status byte is defined according to HP-IL conventions. Each status byte indicates only one state or event—usually the one with the highest priority. The following table summarizes defined system status bytes in order from highest to lowest priority.

REQUEST (continued)

System Status Byte

Status Byte Value		Meaning
Service Not Requested	Service Requested	
Events:		
134	198	Self-test failure.
136	200	Powering down.
137	201	External service request.
130	194	Manual intervention required.
131	195	Data error.
135	199	Command error.
133	197	No room for data.
132	196	Device error.
129	193	Low battery.
138	202	Device-dependent service request.
159	223	ASCII display follows.
States:		
160	224	Request control of loop.
162	226	Ready to send data.
161	225	Ready to receive data.
163	227	Not ready to receive or send data.
128	192	All okay.

Device Status Byte. The meaning of a device status byte depends upon the definition that you use—there's no standard definition (except that bit 7 should be "0" and bit 6 indicates the service request condition). Each bit may indicate the occurrence or absence of a particular condition, so that several conditions can be indicated by one byte.

The value defined by `REQUEST` is related to the states of the eight individual bits in the status byte. If a bit is "1", the value of that bit is added to the values of other bits that are "1". If a bit is "0", its value isn't included in the `REQUEST` value. For example, if a status byte is to be expressed in binary form as 01001011, the value required by `REQUEST` is 75:

Bit Number:	7	6	5	4	3	2	1	0
Bit Value:	128	64	32	16	8	4	2	1
Bit Setting:	0	1	0	0	1	0	1	1
		↓			↓		↓	↓
		64			8		2	1 → 75

REQUEST (continued)

Sending Service Requests. The state of bit 6 determines whether or not the HP-71 requests service from the HP-IL controller. If bit 6 of the HP-71 status byte is set by `REQUEST`, the HP-71 begins requesting service from the HP-IL controller. It continues to request service until one of these conditions occurs:

- The HP-IL controller reads the HP-71 status byte. Essentially, the controller satisfies the HP-71 by taking note of its condition. Note that bit 6 remains *set*, but that the HP-71 *stops* requesting service. (It won't request service again until `REQUEST` defines another HP-71 status byte having bit 6 set.)
- Another `REQUEST` statement clears bit 6, stopping the HP-71 from requesting service.

The HP-IL controller may be able to detect a service request indication and realize that some device is requesting service. It may choose either to conduct a serial poll to determine which device requests service or else to ignore the service request.

Advanced User's Note: If the HP-IL system is idle and if devices have been enabled to source "service request" messages, then the HP-71 will send such messages if bit 6 is set.

If the system isn't idle, or if devices haven't been enabled to source "service request" messages, then the HP-71 modifies certain HP-IL messages (to indicate a service request) when it passes them to the next device if bit 6 is set.

HP-IL Messages

None.

Related Keywords

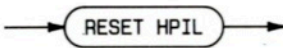
`STATUS`, `SPOLL`.

RESET HPIL

Resets the HP-IL interface to a known condition.

■ Statement
□ Function
□ Operator

■ Keyboard Execution
□ CALC Mode
■ IF...THEN...ELSE
■ Device Operation



Examples

RESET HPIL

Resets the HP-IL interface.

IF LEN(A\$)>L THEN RESET HPIL

If the length of A\$ is greater than L, resets the HP-IL interface.

Operation

The `RESET HPIL` statement resets all conditions maintained by the HP-IL interface and performs a self-test of the interface. (It causes an error if the interface fails the self-test—refer to “Verifying Proper Operation” in appendix A for additional information about the self-test.)

`RESET HPIL` has no effect on the rest of the HP-IL system—no HP-IL messages are sent when `RESET HPIL` is executed. It affects only HP-71 conditions.

The conditions affected by `RESET HPIL` are

- The HP-71 begins behaving as the HP-IL controller.
- HP-IL devices are assigned new addresses at the beginning of the next I/O operation.
- The timeout parameters set by `STANDBY` are set to a timeout period of 60 seconds and a verify interval of 2 seconds.
- The internal HP-IL status byte is set to 32 (reflecting the role of the HP-71 as controller).
- The interrupt mask defined by `ENABLE INTR` is set to 0, disabling all HP-IL interrupts.
- The interrupt-cause byte returned by `READINTR` is set to 0, indicating no pending interrupts.
- The HP-71 status byte defined by `REQUEST` is set to 0.
- The device-dependent command number returned by `READDDC` is set to -1.

RESET HPIL (continued)

HP-IL Messages

None.

Related Keywords

CONTROL ON, RESTORE IO.

RESTORE IO

Enables I/O operations to occur on HP-IL.

- Statement
- Function
- Operator

- Keyboard Execution
- CALC Mode
- IF...THEN...ELSE
- Device Operation



Examples

RESTORE IO

```
IF BIT(STATUS,5) THEN RESTORE
IO
```

Enables I/O operations on HP-IL.

If bit 5 of the HP-IL status byte is true (the HP-71 is controller), sets conditions for I/O operations.

Operation

The `RESTORE IO` statement sets HP-IL conditions for enabling I/O operations. It has three major uses:

- To resume I/O operations that were disabled by `OFF IO`, including the use of the `DISPLAY IS` device and `PRINTER IS` device.
- To restore HP-IL operation and clear “busy” devices after interrupting an I/O operation by pressing `[ATTN]` `[ATTN]`, resulting in an `Aborted` message.
- To assign new addresses to HP-IL devices. This might be done after connecting or removing an HP-IL device or turning a device off and on. (This is also accomplished by turning the HP-71 off and on with flags -21 and -24 clear.)

The conditions affected by `RESTORE IO` are

- Clears the controller, talker, and listener status of all devices (if the HP-71 is controller).
- Activates the `DISPLAY IS` device (if the HP-71 is controller).
- Assigns addresses to HP-IL devices (if the HP-71 is controller).

If the HP-71 is operating as a device, the only effect of `RESTORE IO` is to enable I/O operations that were disabled by `OFF IO`.

RESTORE IO (continued)

HP-IL Messages

Interface Clear, Auto Extended Secondary and Auto Extended Primary (only if flag -22 set and ASSIGN IO not used), Auto Address—all only if controller.

Related Keywords

CONTROL ON, OFF IO, RESET HPIL.

SECURE

Prevents a file from being altered or purged.

- ☒ Statement
- ☐ Function
- ☐ Operator

- ☒ Keyboard Execution
- ☐ CALC Mode
- ☒ IF...THEN...ELSE
- ☐ Device Operation



Examples

`SECURE "RATES:MASSMEM(2)"`

Secures the file `RATES` in the second mass storage device.

`SECURE ROCKIN.ROLL`

Secures the file `ROCKIN` on the medium with volume label `ROLL`.

Input Parameters

Item	Description	Restrictions
file specifier	See standard definition.	Must not be SDATA file.

Operation

The `SECURE` statement provides protection for a file from being purged or altered—that is, the file can't be erased, and information can't be stored in the file. However, the file can be copied, read, and renamed. `SECURE` provides a safeguard against accidentally changing a file.

SECURE (continued)

Any type of file except SDATA file can be protected using `SECURE`. The protection can be removed using the `UNSECURE` statement.

A secure file is displayed in a catalog listing with an “S” in the protection field. If a secure file is also private (using `PRIVATE`), it has an “E” (for “execute only”) in the protection field.

Note: As a precaution, only an unsecure file can be made private.

If you secure a data file that has already been opened by an `ASSIGN #` statement, the file doesn’t become secure until after it’s closed.

Refer to the *HP-71 Reference Manual* for information about using files in main RAM (or independent RAM).

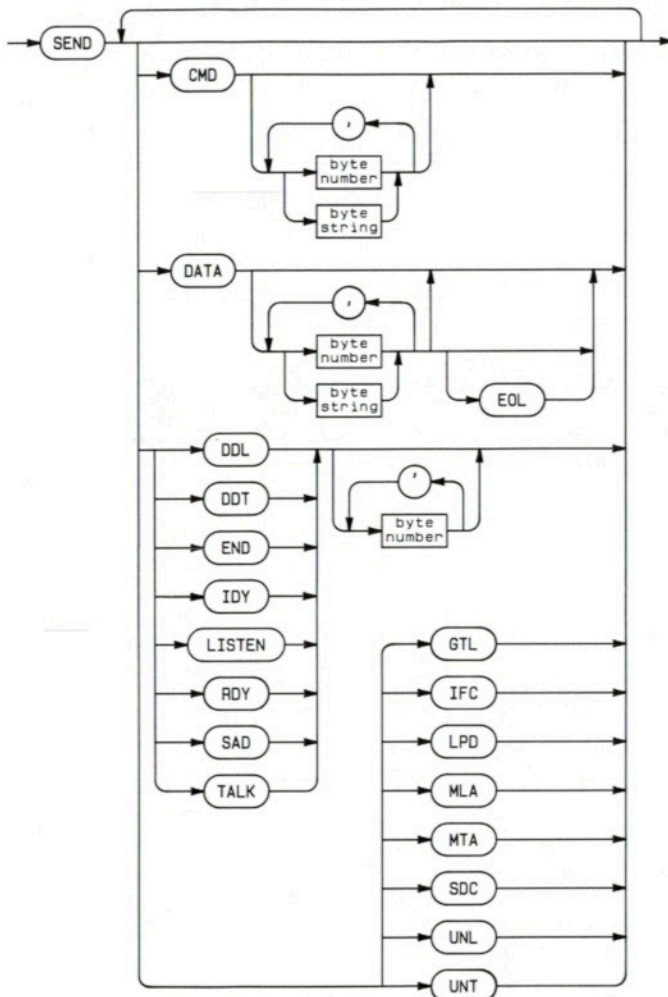
Related Keywords

`PRIVATE`, `UNSECURE`.

SEND

Sends individual HP-IL messages on the loop.

- | | |
|-------------|----------------------|
| ■ Statement | ■ Keyboard Execution |
| □ Function | □ CALC Mode |
| □ Operator | ■ IF...THEN...ELSE |
| | ■ Device Operation |



SEND (continued)

Examples

SEND IFC	Sends an Interface Clear message, clearing all talkers and listeners.
SEND UNT UNL LISTEN 2,6,7 MTA	Sends message sequence that sets up listeners at addresses 2, 6, and 7, and makes the HP-71 a talker. The HP-71 can now send data to three listeners using OUTPUT LOOP.
SEND UNT UNL CMD 146 LISTEN A1 MTA DATA "P4;" UNT UNL CMD 147	Makes the system Remote enabled (command 146), makes device at address A1 a listener (also setting it to Remote mode), sends it the Remote instruction "P4;" (meaning "no parity" to HP-IL/RS-232 interface), then makes system not Remote enabled (command 147).
SEND UNL CMD C\$ LISTEN 1 SAD 2,4 MTA DATA X\$ UNT UNL	Sends HP-IL command message(s) specified by C\$, makes devices at extended addresses 1.03 and 1.05 listeners, makes the HP-71 a talker, and sends data specified by X\$.

Input Parameters

Item	Description	Restrictions
byte number	Numeric expression, which is rounded to an integer. Default: 0.	0 through 255.
byte string	String expression. Default: Byte number 0.	None.

Operation

The SEND statement enables the HP-71 to source individual HP-IL messages. This enables advanced users to control the HP-IL system by sending HP-IL messages one at a time.

Note: Most HP-IL operations can be performed most easily using other statements and functions provided by the HP-IL interface. The SEND statement enables the HP-71 to perform more-specialized operations by controlling the HP-IL system on a message-by-message level. This requires careful planning and a detailed understanding of HP-IL operation.

SEND (continued)

Each HP-IL message is defined by 11 bits: 3 *control* bits and 8 *data* bits. HP-IL messages are separated into four groups according to their control bits:

- **Command group.** These messages convey instructions from the controller and are monitored by *all* HP-IL devices (including idle devices).
- **Ready group.** These messages provide special-purpose communication between the controller and one or more devices, and are generally used to coordinate the transfer of instructions and data.
- **Identify group.** These messages enable devices to request service from the controller. Any device can modify these messages to indicate a service request condition to the controller.
- **Data/end group.** These messages convey data between active devices (possibly including the controller). Any device can modify these messages to indicate a service request condition to the controller.

SEND provides several message indicators and qualifiers that define the HP-IL messages to be sent. The following table describes how they're used.

Message Indicators and Qualifiers

Indicator	Qualifier	Description
Command Group		
CMD	byte number	Command message having data bits set according to byte number. Any command message can be sent this way.
	byte string	Sequence of command messages, each having data bits set according to character code of each character. Any command message can be sent this way.
DDL	byte number	Device-Dependent Listener message having number 0 through 31 indicated by byte number (modulo 32).
DDT	byte number	Device-Dependent Talker message having number 0 through 31 indicated by byte number (modulo 32).
GTL		Go To Local message. Sets all listeners to Local mode (but they return to Remote mode when they next become listeners).
IFC		Interface Clear message. Clears all "busy" devices from controller, listener, and talker status. (If received by another HP-71, that HP-71 then operates as an HP-IL device.)

SEND (continued)**Message Indicators and Qualifiers (continued)**

Indicator	Qualifier	Description
LISTEN	byte number	Listen Address message having address 0 through 31 indicated by byte number (modulo 32). Makes device at address a listener—except that 31 clears all devices (including HP-71) from listener status. (See SAD for extended addressing.)
LPD		Loop Power Down message. Sets all devices with capability to low-power state.
MLA		No message sent. Makes HP-71 a listener.
MTA		Untalk message. Makes HP-71 the only talker.
SAD	byte number	Secondary Address message having address 0 through 31 indicated by byte number (modulo 32). Associates this secondary address with the primary address of the preceding command message, indicating an extended address. Note: Actual HP-IL secondary addresses range from 0 to 30. The secondary part of HP-71 extended addresses ranges from 1 to 31. For example, use SAD 2 to indicate the secondary address of a device with extended address 1.03.
SDC		Selected Device Clear message. Resets the internal conditions of all listeners to operational startup conditions. (For HP-71, clears internal HP-IL input and output buffers.)
TALK	byte number	Talk Address message having address 0 through 31 indicated by byte number (modulo 32). Makes device at address the only talker—except that 31 clears all devices (including HP-71) from talker status. (See SAD for extended addressing.)
UNL		Unlisten message. Clears all devices (including HP-71) from listener status.
UNT		Untalk message. Clears all devices (including HP-71) from talker status.

SEND (continued)**Message Indicators and Qualifiers (continued)**

Indicator	Qualifier	Description
Ready Group		
RDY	byte number	Ready message having data bits set according to byte number.
Identify Group		
IDY	byte number	Identify message having data bits set according to byte number.
Data/End Group		
DATA	byte number	Data Byte message having data bits set according to byte number. Byte number indicates character code of character.
	byte string	Sequence of Data Byte messages that transfer the characters defined by the byte string.
END	byte number	End Byte message having data bits set according to byte number. Byte number indicates character code of character.
EOL		Sequence of Data Byte messages that transfer the end-of-line characters defined by the <code>ENDLINE</code> statement. (May be used only immediately after a <code>DATA</code> byte number or string.)

To use `SEND` successfully, *you must follow HP-IL protocol*. A full discussion of HP-IL protocol is beyond the scope of this manual. Two references for learning about HP-IL protocol are

- Kane, Gerry, et al. *The HP-IL System: An Introductory Guide to the Hewlett-Packard Interface Loop*. Osborne/McGraw-Hill, Berkeley, California, ©1982.
- Hewlett-Packard Company. *The HP-IL Interface Specification*. HP part number 82166-90017, ©1982.

CAUTION

Be sure that the messages sent by `SEND` follow standard HP-IL protocol. Unusual sequences of HP-IL messages may disrupt HP-IL operation and may possibly destroy stored or transferred data.

If the HP-71 is operating as a controller, it can use `SEND` to send any message, although certain messages from the ready group may disrupt HP-IL operation. (Refer to “Controller Operation” below.)

SEND (continued)

If the HP-71 is operating as a device, it can use `SEND` to send only certain messages. (Refer to "Device Operation" below.)

Controller Operation. While the HP-71 is a controller, the `SEND` statement can be used with other HP-IL operations to expand the capabilities of those operations. In particular, `SEND` can be used to define the source or destination(s) of information to be transferred using `OUTPUT`, `ENTER`, and `COPY` (with their `LOOP` options). Consider these examples:

```
SEND UNT UNL LISTEN 2,3,5 MTA
OUTPUT LOOP; X;Y;A$;"END OF
  DATA"
```

Sets up three listeners and makes the HP-71 the talker; then sends specified data to all active listeners.

```
SEND UNT UNL LISTEN 4,6 MTA
COPY "EXPER" TO :LOOP
```

Sets up two listeners and makes the HP-71 the talker; then sends file `EXPER` to all active listeners.

```
SEND UNT UNL MLA LISTEN 4
TALK 3
ENTER LOOP; A$,B$
```

Sets up HP-71 and one other device as listeners and another device as talker; then directs the talker to send data to all active listeners (and stores it in HP-71 variables `A$` and `B$`).

```
SEND UNT UNL MLA LISTEN 2,4
TALK 3
COPY :LOOP TO "FILE1"
```

Sets up HP-71 and two other devices as listeners and another device as talker; then directs the talker to send file data to all active listeners (and stores it in main RAM file `FILE1`).

Notice that in each example the `SEND` statement includes either `MLA` or `MTA`, which prepares the HP-71 either to receive data or to send data—without this provision, the HP-71 wouldn't be able to send or receive the data in the following I/O statement. Notice also that in each example the `SEND` statement includes `UNL` and `UNT` to remove all devices from listener and talker status before defining new listeners and talkers.

`SEND` normally sends only those messages specified in that statement. (The Ready For Command message that must follow each command message is automatically sent after each specified command message.) However, there are certain conditions for which the HP-71 automatically sends other messages:

- If a `DISPLAY IS` device is defined, the HP-71 automatically sends messages that convey keyboard input and deactivate the device—after the `SEND` operation finishes, the HP-71 sends messages that remove all devices from talker and listener status and make the `DISPLAY IS` device a listener.
- If valid addresses have not been assigned (such as after turning off the HP-71 with flag `-21` clear), the HP-71 automatically assigns addresses to HP-IL devices when the first I/O operation is per-

SEND (continued)

formed. If that operation is a `SEND` statement, the addressing messages are sent before the messages specified by `SEND`.

- If `RESET HPIL` has been executed, the HP-71 automatically assigns addresses to HP-IL devices when the next I/O operation is performed. If that operation is a `SEND` statement, the addressing messages are sent before the messages specified by `SEND`.

Refer to “Related Keywords” below for other statements that can cause the HP-71 to send individual HP-IL messages.

You should be careful when using `SEND` to send the following messages from the ready group: `RDY 96` (Send Data), `RDY 97` (Send Status), `RDY 98` (Send Device ID), and `RDY 99` (Send Accessory ID). Each of these messages instructs the talker to send information immediately, but the HP-71 can't receive data because it's still executing the `SEND` statement. The only situation in which you might send one of these messages is to start a data transfer between devices—a transfer that doesn't involve the HP-71. To avoid disrupting such a transfer, the HP-71 must not send any other messages until the transfer is completed—check bit 7 of the HP-IL interface status returned by `STATUS`.

Device Operation. While the HP-71 is operating as a device, it can use only the following message indicators (and qualifiers) with `SEND`:

- `DATA` (all, including `EOL`) only while the HP-71 is a talker.
- `END` (all) only while the HP-71 is a talker.
- `RDY 64` (End Of Transmission—OK) only while the HP-71 is a talker. (End Of Transmission—Error is sent if a data error has occurred.)
- `RDY 65` (End Of Transmission—Error) only while the HP-71 is a talker. (End Of Transmission—OK is sent if no data error has occurred.)

These messages are useful if the controller requires an End Of Transmission message after a data transfer. `OUTPUT` and `PRINT` don't automatically send an End Of Transmission message.

Notice that these messages can be sent only while the HP-71 is a talker. Check bit 4 of the HP-IL interface status returned by `STATUS`.

HP-IL Messages

None (except as specified by `SEND`).

Related Keywords

`CLEAR`, `COPY`, `ENTER`, `LOCAL`, `LOCAL LOCKOUT`, `OUTPUT`, `REMOTE`, `TRIGGER`.

SPOLL

Returns a value that represents one or more status bytes from an HP-IL device.

- ☐ Statement
- ☒ Function
- ☐ Operator
- ☒ Keyboard Execution
- ☒ CALC Mode
- ☒ IF...THEN...ELSE
- ☐ Device Operation



Examples

A=SPOLL(I)

Sets A to the value representing the status of the device at address I.

IF SPOLL("TAPE(2)")=20 THEN 100

If the status of the second mass storage device equals 20 (no tape installed in cassette drive), then branches to line 100.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	Must not be unquoted string.

Operation

The SPOLL function returns a decimal value that represents the status of an HP-IL device. Obtaining status information from one device at a time is often called a *serial poll*.

Note: A device's status (an external status) is returned by SPOLL to the HP-71 while it's the controller. It's analogous to (but differs from) the HP-71 status byte defined by REQUEST, which is to be sent to another HP-IL controller by the HP-71 while it's a device. It also differs from the HP-IL interface status (an internal status) returned by STATUS.

Many devices return one byte of status information—SPOLL returns the decimal value of that byte. Some devices return several bytes of status information—SPOLL accepts up to eight bytes and combines the first four bytes into one decimal value that represents that information. Some devices don't respond to a request for status information—SPOLL returns a value of -1.

SPOLL (continued)

If the specified device isn't in the loop, `SPOLL` returns a value of `-1`.

If `SPOLL` is to be used in `CALC` mode, the argument must be numeric (an address).

The first status byte sent by a device has special significance. If that byte is expressed as an eight-bit binary number, the bits have the following meanings:

- Bit 7 indicates the type of status represented by that byte.
 - “0” Device status.
 - “1” System status.
- Bits 6 through 0 indicate the condition of the device.

The state of bit 7 (the most significant bit) defines two ranges of status bytes:

- System status, which can have values from 128 through 255. The conditions indicated by bits 6 through 0 have been defined by HP-IL conventions.
- Device status, which can have values from 0 through 127. The conditions indicated by bits 6 through 0 are *not* defined in general—they're defined differently for each device.

To determine the meaning of a device's status information, refer to the owner's manual for that device.

System Status Byte. The meaning of a system status byte is defined according to HP-IL conventions. (The state of bit 6 indicates whether or not service is requested from the controller. Bits 5 through 0 indicate the actual condition.) Each status byte indicates only one state or event—usually the one with the highest priority. The following table summarizes defined system status bytes in order from highest to lowest priority.

SPOLL (continued)

System Status Byte

Status Byte Value		Meaning
Service Not Requested	Service Requested	
Events:		
134	198	Self-test failure.
136	200	Powering down.
137	201	External service request.
130	194	Manual intervention required.
131	195	Data error.
135	199	Command error.
133	197	No room for data.
132	196	Device error.
129	193	Low battery.
138	202	Device-dependent service request.
159	223	ASCII display follows.
States:		
160	224	Request control of loop.
162	226	Ready to send data.
161	225	Ready to receive data.
163	227	Not ready to receive or send data.
128	192	All okay.

The BIT function is useful for checking bit 6 for a service request condition (or for checking any other bit). Refer to “Responding to Service Requests” below for additional information about using service requests.

Device Status Byte. The meaning of a device status byte depends upon the the definition used by the device—there’s no standard definition (except that bit 7 should be “0” for the first byte). Each bit may indicate the occurrence or absence of a particular condition, so that several conditions can be indicated by one byte.

SPOLL (continued)

The value returned by `SPOLL` is related to the states of the eight individual bits in the status byte. If a bit is “1”, the value of that bit is added to the values of other bits that are “1”. If a bit is “0”, its value isn’t included in the `SPOLL` value. For example, if a status byte is expressed in binary form as 01001011, the value returned by `SPOLL` is 75:

Bit Number:	7	6	5	4	3	2	1	0
Bit Value:	128	64	32	16	8	4	2	1
Bit Setting:	0	1	0	0	1	0	1	1
		↓			↓		↓	↓
		64		+	8	+	2 + 1	→ 75

The `BIT` function is useful for checking the individual bits of a status byte.

Multiple Status Bytes. Some devices send more than one byte of status information. All additional bytes are *device status bytes*. `SPOLL` combines up to four status bytes into one decimal value. (Four additional bytes can be accepted but are ignored.) The first status byte is used as the “lowest-order” byte—the last status byte is the “highest-order” byte. Each byte has a multiplier that is a power of 256. For example, if a device sends three status bytes, `SPOLL` represents the status using the following scheme.

Third Byte								Second Byte								First Byte							
Bit Number:								Bit Number:								Bit Number:							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Bit Value:								Bit Value:								Bit Value:							
128	64	32	16	8	4	2	1	128	64	32	16	8	4	2	1	128	64	32	16	8	4	2	1
Multiplier: × 256 ²								× 256								× 1							

The following expressions may be useful for interpreting the status of a device that sends more than one status byte:

`MOD(SPOLL(A), 256)`

Returns the value of the first status byte of device at address `A`.

`MOD(SPOLL(A) DIV (256(I-1)), 256)`

Returns the value of the `I`th status byte of device at address `A`.

`BIT(SPOLL(A) DIV (256(I-1)), N)`

Returns the state of bit `N` of the `I`th status byte of device at address `A`.

SPOLL (continued)

Responding to Service Requests. In general, an HP-IL controller may be able to detect a service request indication and realize that some device is requesting service. It may choose either to conduct a serial or parallel poll to determine which device requests service, or else to ignore the service request.

Operating as a controller, the HP-71 can detect a service request indication. The receipt of a service request is an HP-IL event that the HP-71 notes in bit 3 of its HP-IL interface status and in bit 3 of its *interrupt-cause byte*. You can determine the HP-IL interface status by using `STATUS`. The interrupt-cause byte is usually used as part of a subroutine that processes an HP-IL interrupt. An HP-IL interrupt subroutine provides a convenient way to detect and respond to a service request—especially since it's used only when a service request is detected. (Refer to `ON INTR` for more information about interrupt subroutines.)

`SPOLL` is often used with four other keywords to define how the HP-71 responds to service requests:

- `ENABLE INTR` would define the receipt of a service request to be an enabled HP-IL interrupt event.
- `ON INTR` would define the branching that a program takes when an enabled HP-IL interrupt event occurs.
- `READINTR` would be part of the interrupt subroutine. It returns the *interrupt-cause byte*, which indicates the HP-IL event that has occurred. Bit 3 of the interrupt-cause byte indicates the receipt of a service request.
- `SPOLL` would be part of the interrupt subroutine. It returns the status information from a device. This function would be executed for each HP-IL device as part of a search for the device requesting service—a serial poll. It would also indicate the condition of that device so that appropriate action could be taken by the HP-71.

If a device returns a *system status byte*, the state of bit 6 indicates whether or not the device has requested service from the HP-IL controller. A device that requests service usually continues to request service until one of these conditions occurs:

- The HP-IL controller reads the status of that device. Essentially, the controller satisfies the device by taking note of its condition.
- The device's condition changes, clearing the condition that caused the service request.

HP-IL Messages

Send Status.

Related Keywords

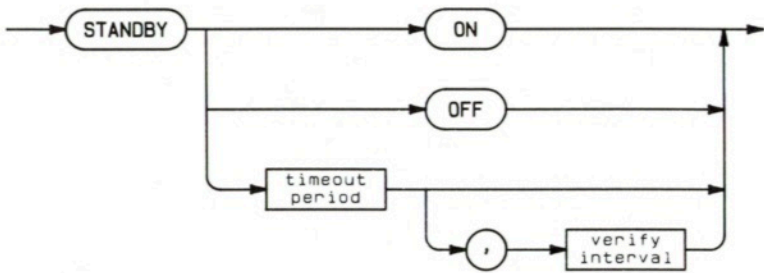
`DEVADDR`, `DEVAID`, `DEVID$`.

STANDBY

Sets the HP-IL timeout period and verify interval.

■ Statement
□ Function
□ Operator

■ Keyboard Execution
□ CALC Mode
■ IF...THEN...ELSE
■ Device Operation



Examples

STANDBY 15

Sets the timeout period and verify interval to 15 seconds.

STANDBY 100,10

Sets the timeout period to 100 seconds and the verify interval to 10 seconds.

IF S THEN STANDBY ON

If S is true, sets the timeout period and verify interval to infinity (HP-71 waits forever for HP-IL response).

Input Parameters

Item	Description	Restrictions
timeout period	Numeric expression, which is rounded to three decimal places.	0.001 through 1048.575.
verify interval	Numeric expression, which is rounded to three decimal places. Default: Same as timeout period.	1/255 of timeout period (must be at least 0.001) through 1048.575.

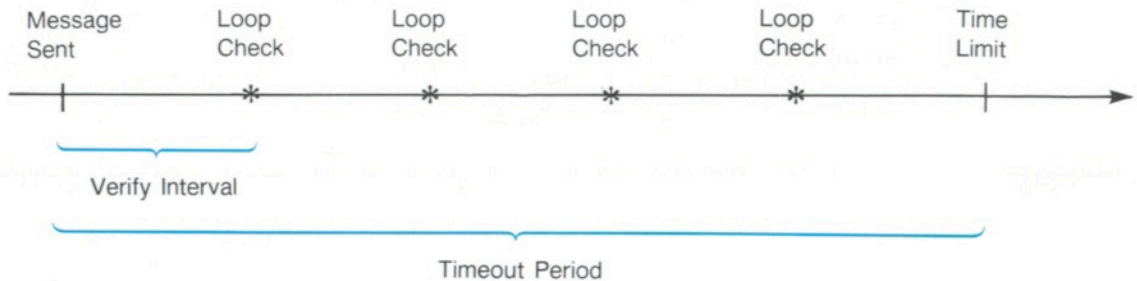
STANDBY (continued)

Operation

The `STANDBY` statement sets two parameters that define how the HP-71 verifies the integrity of the HP-IL system while it's the controller. The two parameters are

- **Timeout period:** defines how long the HP-71 waits for each single HP-IL instruction to travel around the loop and back to the HP-71.
- **Verify interval:** defines how often the HP-71 tests the loop's continuity (by sending an HP-IL Identify message, which travels around the loop quickly). If this parameter isn't specified, it's set equal to the timeout period.

Each parameter is specified in *seconds* (with an implied accuracy of ± 10 percent).



If the timeout period expires and the original message hasn't returned to the HP-71, the operation is cancelled and a `Loop Broken` error (error 255043—timeout) occurs. If any loop check gets no response, the operation is cancelled and a `Loop Broken` error (error 255035—loop not complete) occurs. (The response to the loop check must occur quickly—if your system includes a device that delays all HP-IL messages, you should set the verify interval equal to the timeout period.)

The timeout period is rounded upwards to a multiple of the verify interval. If the timeout period is less than the verify interval, the HP-71 behaves as though the timeout period is as long as the verify interval.

If the `ON` option is specified, the HP-71 enters a “standby” state, in which it waits forever for a message to return and never checks the loop. In effect, the timeout period and verify interval are both set to “infinity.”

If the `OFF` option is specified, the parameters are reset to their original values. The timeout period is set to 60 seconds, and the verify interval is set to 2 seconds.

STANDBY (continued)

You can execute `STANDBY` to set the two parameters while the HP-71 is operating as an HP-IL device. However, the parameters set by the `STANDBY` statement aren't used while the HP-71 is a device. They're used only while the HP-71 is a controller.

HP-IL Messages

None.

Related Keywords

`RESET HPIL.`

STATUS

Returns the HP-IL interface status.

<input type="checkbox"/> Statement	<input checked="" type="checkbox"/> Keyboard Execution
<input checked="" type="checkbox"/> Function	<input checked="" type="checkbox"/> CALC Mode
<input type="checkbox"/> Operator	<input checked="" type="checkbox"/> IF...THEN...ELSE
	<input checked="" type="checkbox"/> Device Operation



Examples

<code>S=STATUS</code>	Sets <code>S</code> to the value of the HP-IL interface status.
<code>IF BIT(STATUS,5) THEN 200</code>	If bit 5 of the HP-IL interface status is true (HP-71 is controller), branches to line 200.

Operation

The `STATUS` function returns the value of the HP-IL interface status. This indicates the current role of the HP-71 in the HP-IL system and gives information about the current state of the system. The status information is maintained internally by the HP-71.

Note: The HP-IL interface status (an internal status) indicates the state of the HP-IL system while the HP-71 is either a controller or a device. It differs from the HP-71 status byte (an internal status) defined by `REQUEST` for sending to the HP-IL controller by the HP-71 while it's a device. It also differs from the device status (an external status) returned by `SPOLL` to the HP-71 while it's the controller.

The value of the HP-IL interface status byte is a decimal value—when expressed in binary form, its eight bits correspond to HP-IL conditions that exist. The `STATUS` value is the sum of the decimal values for the conditions that exist, as described in the following table. (Notice that certain conditions can exist while the HP-71 is a controller, and that others can exist while the HP-71 is a device.)

STATUS (continued)

HP-IL Interface Status Byte

Bit	Decimal Value*	Condition	Controller†	Device‡
7	128	Data Transfer. Data is being transferred between other HP-IL devices.‡	X	
6	64	Listener. HP-71 is a listener. (HP-71 can receive data.)	X	X
5	32	Controller. HP-71 is active controller. (HP-71 controls HP-IL system.)	X	
4	16	Talker. HP-71 is a talker. (HP-71 can send data.)	X	X
3	8	Service Request. HP-71 detected service request bit set in HP-IL message. (Normally tells HP-71 that a device needs attention.)	X	
2	4	Asynchronous Requests. HP-IL devices may source Identify messages to request service from controller. (Loop is idle in this condition.)	X	X
1	2	Remote. HP-71 has been set to Remote mode. (While a device, the HP-71 interprets HP-IL data as BASIC instructions.)	(X)	X
0	1	Local Lockout. HP-IL system has been set to Local Lockout condition. (Has no inherent meaning for HP-71.)	X	X

* HP-IL interface status is sum of values of all conditions that exist.

† An "X" indicates that the interrupt event can occur while the HP-71 is a controller or while the HP-71 is a device. An "(X)" indicates that the event can occur using `SEND LISTEN 0` only.

‡ This condition can occur only during certain operations that are initiated using `SEND`. Refer to `SEND`.

Each bit in the status byte reflects the current HP-IL interface condition—with the exception of bit 3. Bit 3 indicates the service request condition in one of two ways, depending upon the HP-71 situation:

- If the HP-71 is *not* a listener and *not* a talker, bit 3 reflects the service request condition indicated by the *most recent* HP-IL message that can indicate a service request. (In this situation, bit 3 is set or cleared by each Data Byte, End Byte, and Identify message.)
- If the HP-71 is a listener or a talker, bit 3 reflects whether a service request condition was indicated by *any* HP-IL message (not just the most recent one). (In this situation, bit 3 is set by any HP-IL message that indicates a service request—but it isn't cleared by any message.)

STATUS (continued)

For additional information about HP-IL interface conditions, refer to the following keyword entries:

- Controller: CONTROL OFF/ON.
- Service Request: SPOLL.
- Asynchronous Requests: REQUEST.
- Remote: LOCAL and REMOTE.
- Local Lockout: LOCAL LOCKOUT.

HP-IL Messages

None.

Related Keywords

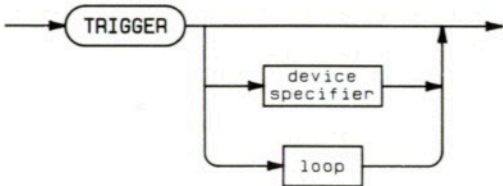
ENABLE INTR.

TRIGGER

Triggers an event at an HP-IL device.

☒ Statement
☐ Function
☐ Operator

☒ Keyboard Execution
☐ CALC Mode
☒ IF...THEN...ELSE
☐ Device Operation



Examples

TRIGGER ":HP3468A"

Triggers the first HP 3468A Multimeter, causing it to make a measurement.

IF T THEN TRIGGER 3

If T is true, triggers the device at address 3.

TRIGGER LOOP

Triggers all devices that are set to listener status.

Input Parameters

Item	Description	Restrictions
device specifier	See standard definition.	None.
loop	See standard definition.	None.

Operation

The TRIGGER statement triggers an event at the specified HP-IL device. The nature of the event depends upon the individual device. For example, the HP 3468A Multimeter can take a voltage, current, or resistance reading each time it's triggered.

TRIGGER (continued)

If a device is specified, only that device is triggered.

If no device is specified, or if the `LOOP` option is used, the `TRIGGER` statement doesn't define which devices are triggered—instead it triggers all devices that are set up as listeners. (You can define listeners using the `SEND` statement.)

HP-IL Messages

Group Execute Trigger.

Related Keywords

`SEND`.

UNSECURE

Cancels security for a file, allowing it to be altered or purged.

- ☒ Statement
☐ Function
☐ Operator
- ☒ Keyboard Execution
☐ CALC Mode
☒ IF...THEN...ELSE
☐ Device Operation



Examples

UNSECURE RATES:TAPE(2)

Cancels security for the file RATES in the second mass storage device.

UNSECURE PARKER.ROLL

Cancels security for the file PARKER on the medium with volume label ROLL.

Input Parameters

Item	Description	Restrictions
file specifier	See standard definition.	None.

Operation

The UNSECURE statement cancels a file’s security (which protects a file from being purged or altered)—that is, new or revised information can be stored in the file, and the file can be erased. (The security is established using the SECURE statement.) Newly created files are unsecure.

An unsecure file is displayed in a catalog listing with no symbol in the protection field. If an unsecure file is private (using PRIVATE), it has a “P” in the protection field.

Refer to the *HP-71 Reference Manual* for information about using files in main RAM (or independent RAM).

Related Keywords

PRIVATE, SECURE.

Appendixes

Care, Warranty, and Service Information

Care of the Interface

The HP-IL interface does not require maintenance. However, there are several precautions listed below that you should observe.

CAUTIONS

- Touch the computer while preparing to install the HP-IL interface to neutralize any electrostatic charge.
- Do not place fingers, tools, or other objects into the HP-IL port. Damage to the computer's internal circuitry may result.
- Turn off the HP-71 before installing or removing the HP-IL interface.
- If the interface jams when inserted into the port, it may be upside down. Attempting to force it further may result in damage to the computer or the interface.
- Handle the interface carefully while it is out of the computer. Always keep the cover on the port when the interface is not installed.

Failure to observe these cautions may result in damage to the interface or to the computer.

In order to maintain product reliability, you should observe the following temperature and humidity limits:

- Operating Temperature: 0° to 45° C (32° to 113° F).
- Storage Temperature: -40° to 55° C (-40° to 131° F).
- Operating and Storage Humidity: 0 to 95 percent relative humidity.

Verifying Proper Operation

If at any time you suspect that the HP-IL interface is not operating properly, you can verify its operation using the following procedure:

1. Turn off the HP-71.
2. Connect a single HP-IL cable between the IN and OUT receptacles of the interface.
3. Turn on the HP-71. If the cursor doesn't appear immediately, press **ATTN** **ATTN** (at least two more times).
 - If the cursor appears, continue with step 4.
 - If the HP-71 won't turn on, check it without the HP-IL interface. If it operates properly only with the interface removed, the interface requires service.
4. Execute **RESET HPIL**. This performs a self-test of the interface's internal circuitry.
 - If the cursor appears, the HP-IL interface is good.
 - If **Self Test Failed** is displayed, the HP-IL interface requires service.
5. Execute **CONTROL ON**. This checks HP-IL continuity.
 - If the cursor appears immediately, HP-IL continuity is good.
 - If **Loop Broken** is displayed after several seconds delay, HP-IL continuity is bad. Try another HP-IL cable. If the HP-IL continuity remains bad, the HP-IL interface requires service.

Limited One-Year Warranty

What We Will Do

The HP-IL interface is warranted by Hewlett-Packard against defects in materials and workmanship for one year from the date of original purchase. If you sell your unit or give it as a gift, the warranty is transferred to the new owner and remains in effect for the original one-year period. During the warranty period, we will repair or, at our option, replace at no charge a product that proves to be defective, provided you return the product, shipping prepaid, to a Hewlett-Packard service center.

What Is Not Covered

The warranty does not apply if the product has been damaged by accident or misuse or as the result of service or modification by other than an authorized Hewlett-Packard service center.

No other express warranty is given. The repair or replacement of a product is your exclusive remedy. **ANY OTHER IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS IS LIMITED TO THE ONE-YEAR DURATION OF THIS WRITTEN WARRANTY.** Some states, provinces, or countries do not allow limitations on how long an implied warranty lasts, so the above limitation

may not apply to you. **IN NO EVENT SHALL HEWLETT-PACKARD COMPANY BE LIABLE FOR CONSEQUENTIAL DAMAGES.** Some states, provinces, or countries do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

This warranty gives you specific legal rights, and you may also have other rights which vary from state to state, province to province, or country to country.

Warranty for Consumer Transactions in the United Kingdom

This warranty shall not apply to consumer transactions and shall not affect the statutory rights of a consumer. In relation to such transactions, the rights and obligations of Seller and Buyer shall be determined by statute.

Obligation to Make Changes

Products are sold on the basis of specifications applicable at the time of manufacture. Hewlett-Packard shall have no obligation to modify or update products once sold.

Warranty Information

If you have any questions concerning this warranty, please contact an authorized Hewlett-Packard dealer or a Hewlett-Packard sales and service office. Should you be unable to contact them, please contact:

- In the United States:

Hewlett-Packard
Personal Computer Group
Customer Support
11000 Wolfe Road
Cupertino, CA 95014
Toll-Free Number: (800) FOR-HPPC (800 367-4772)

- In Europe:

Hewlett-Packard S.A.
150, route du Nant-d'Avril
P.O. Box CH-1217 Meyrin 2
Geneva
Switzerland
Telephone: (022) 83 81 11

Note: Do not send units to this address for repair.

- In other countries:

Hewlett-Packard Intercontinental
3495 Deer Creek Rd.
Palo Alto, California 94304
U.S.A.
Telephone: (415) 857-1501

Note: Do not send units to this address for repair.

Service

Service Centers

Hewlett-Packard maintains service centers in most major countries throughout the world. You may have your unit repaired at a Hewlett-Packard service center any time it needs service, whether the unit is under warranty or not. There is a charge for repairs after the one-year warranty period.

Hewlett-Packard products are normally repaired and reshipped within five (5) working days of receipt at any service center. This is an average time and could vary depending upon the time of year and the work load at the service center. The total time you are without your unit will depend largely on the shipping time.

Obtaining Repair Service in the United States

The Hewlett-Packard United States Service Center for battery-powered computational products is located in Corvallis, Oregon:

Hewlett-Packard Company
Service Department
P.O. Box 999
Corvallis, Oregon 97339, U.S.A.
or
1030 N.E. Circle Blvd.
Corvallis, Oregon 97330, U.S.A.
Telephone: (503) 757-2000

Obtaining Repair Service in Europe

Service centers are maintained at the following locations. For countries not listed, contact the dealer where you purchased your unit.

AUSTRIA

HEWLETT-PACKARD Ges.m.b.H.
Kleinrechner-Service
Wagramerstrasse-Lieblgasse 1
A-1220 Wien (Vienna)
Telephone: (0222) 23 65 11

BELGIUM

HEWLETT-PACKARD BELGIUM SA/NV
Woluwedal 100
B-1200 Brussels
Telephone: (02) 762 32 00

DENMARK

HEWLETT-PACKARD A/S
Datavej 52
DK-3460 Birkerød (Copenhagen)
Telephone: (02) 81 66 40

EASTERN EUROPE

Refer to the address listed under Austria.

FINLAND

HEWLETT-PACKARD OY
Revontulentie 7
SF-02100 Espoo 10 (Helsinki)
Telephone: (90) 455 02 11

FRANCE

HEWLETT-PACKARD FRANCE
Division Informatique Personnelle
S.A.V. Calculateurs de Poche
F-91947 Les Ulis Cedex
Telephone: (6) 907 78 25

GERMANY

HEWLETT-PACKARD GmbH
Kleinrechner-Service
Vertriebszentrale
Berner Strasse 117
Postfach 560 140
D-6000 Frankfurt 56
Telephone: (611) 50041

ITALY

HEWLETT-PACKARD ITALIANA S.P.A.
Casella postale 3645 (Milano)
Via G. Di Vittorio, 9
I-20063 Cernusco Sul Naviglio (Milan)
Telephone: (2) 90 36 91

NETHERLANDS

HEWLETT-PACKARD NEDERLAND B.V.
Van Heuven Goedhartlaan. 121
NL-1181 KK Amstelveen (Amsterdam)
P.O. Box 667
Telephone: (020) 472021

NORWAY

HEWLETT-PACKARD NORGE A/S
P.O. Box 34
Oesterndalen 18
N-1345 Oesteraas (Oslo)
Telephone: (2) 17 11 80

SPAIN

HEWLETT-PACKARD ESPANOLA S.A.
Calle Jerez 3
E-Madrid 16
Telephone: (1) 458 2600

SWEDEN

HEWLETT-PACKARD SVERIGE AB
Skalholtsgatan 9, Kista
Box 19
S-163 93 Spanga (Stockholm)
Telephone: (08) 750 2000

SWITZERLAND

HEWLETT-PACKARD (SCHWEIZ) AG
Kleinrechner-Service
Allmend 2
CH-8967 Widen
Telephone: (057) 31 21 11

UNITED KINGDOM

HEWLETT-PACKARD Ltd.
King Street Lane
GB-Winnersh, Wokingham
Berkshire RG11 5AR
Telephone: (0734) 784 774

International Service Information

Not all Hewlett-Packard service centers offer service for all models of HP products. However, if you bought your product from an authorized Hewlett-Packard dealer, you can be sure that service is available in the country where you bought it.

If you happen to be outside of the country where you bought your unit, you can contact the local Hewlett-Packard service center to see if service is available for it. If service is unavailable, please ship the unit to the address listed above under "Obtaining Repair Service in the United States." A list of service centers for other countries can be obtained by writing to that address. All shipping, reimportation arrangements, and custom costs are your responsibility.

Service Repair Charge

There is a standard repair charge for out-of-warranty repairs. The repair charges include all labor and materials. In the United States, the full charge is subject to the customer's local sales tax. In European countries, the full charge is subject to Value Added Tax (VAT) and similar taxes wherever applicable. All such taxes will appear as separate items on invoiced amounts.

Products damaged by accident or misuse are not covered by the fixed repair charges. In these situations, repair charges will be individually determined based on time and materials.

Service Warranty

Any out-of-warranty repairs are warranted against defects in materials and workmanship for a period of 90 days from date of service.

Shipping Instructions

Should your unit require service, return it with the following items:

- A completed Service Card, including a description of the problem and system setup when the problem occurred.
- A sales receipt or other documentary proof of purchase date if the one-year warranty has not expired.

The product, the Service Card, a brief description of the problem and system configuration, and (if required) the proof of purchase date should be packaged in the original shipping case or other adequate protective packaging to prevent in-transit damage. Such damage is not covered by the one-year limited warranty; Hewlett-Packard suggests that you insure the shipment to the service center. The packaged unit should be shipped to the nearest Hewlett-Packard designated collection point or service center. Contact your dealer for assistance. (If you are not in the country where you originally purchased the unit, refer to "International Service Information" above.)

Whether the unit is under warranty or not, it is your responsibility to pay shipping charges for delivery to the Hewlett-Packard service center.

After warranty repairs are completed, the service center returns the unit with postage prepaid. On out-of-warranty repairs in the United States and some other countries, the unit is returned C.O.D. (covering shipping costs and the service charge).

Further Information

Service contracts are not available. Circuitry and designs are proprietary to Hewlett-Packard, and service manuals are not available to customers.

Should other problems or questions arise regarding repairs, please call your nearest Hewlett-Packard service center.

Potential For Radio/Television Interference (for U.S.A. only)

The HP-IL interface generates and uses radio frequency energy and, if not installed and used properly—that is, in strict accordance with the instructions in this manual—may cause interference to radio and television reception. The interface has been tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J of Part 15 of FCC rules, which are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. In the unlikely event that the interface does cause interference to radio or television reception (which can be determined by removing the interface from the HP-71), you are encouraged to try to correct the interference by one or more of the following measures:

- Reorient the receiving antenna.
- Relocate the HP-71 or HP-IL cables with respect to the receiver.
- Move the HP-IL system away from the receiver.
- Plug the ac adapter into a different ac outlet so that the HP-71 and the receiver are on different branch circuits.

If necessary, you should consult your 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 Number 004-000-00345-4.

When You Need Help

Hewlett-Packard is committed to providing after-sale support of its customers. To this end, our customer support department has established phone numbers that you can call if you have questions about this product.

Product Information. For information about Hewlett-Packard dealers, products, and prices, call:

(800) FOR-HPPC
(800 367-4772)

Technical Assistance. For technical assistance with your product, call the number below:

~~(408) 725-2600~~

503 754 6666

For either product information or technical assistance, you can also write to:

Hewlett-Packard
Personal Computer Group
Customer Support
11000 Wolfe Road
Cupertino, CA 95014

Operating Information

Reset Conditions

There are several HP-IL activities that cause the HP-71 to reset all or part of its conditions. The activities discussed here (in order of increasing complexity) include turning on the HP-71, or executing one of the following statements: `RESTORE IO`, `CONTROL ON`, `RESET HPIL`, or `INIT:3`. (The conditions after executing `INIT:3` are equivalent to the conditions at the initial installation of batteries.)

Turning on the HP-71 causes the `DISPLAY IS` device in the loop to become active (if the HP-71 is the controller). If flag `-21` was clear when the HP-71 was turned off, the HP-71 assigns new addresses to devices in the loop at the next operation that occurs after the HP-71 is turned on. (If flag `-21` was set, the HP-71 assumes the old loop addresses are still valid.)

Executing `RESTORE IO` when the HP-71 is the controller assigns addresses to devices in the loop and causes the `DISPLAY IS` device in the loop to become active. Devices that were "busy" become idle. Executing `RESTORE IO` cancels a previous `OFF IO` statement.

Executing `CONTROL ON` causes the HP-71 to assume control of the loop and assign addresses to devices. The `DISPLAY IS` device in the loop becomes active. Devices that were "busy" become idle. (I/O must be enabled in order to execute `CONTROL ON`; if `OFF IO` is in effect, executing `CONTROL ON` causes an error.)

Executing `RESET HPIL` causes the HP-71 to assume control of the loop. Devices are assigned new addresses at the next operation that occurs. The parameters of the `STANDBY` statement are set to the default values of 60 seconds and 2 seconds.

Performing `INIT:3` causes a total memory reset. The HP-71 assumes control of the loop. Addresses are assigned at the next operation that occurs. The first display device in the loop becomes the `DISPLAY IS` device. The first printer in the loop becomes the `PRINTER IS` device. Assign codes and I/O channel assignments are cancelled, and flags `-21`, `-22`, and `-23` are cleared. The parameters of the `STANDBY` statement are set to the default values of 60 seconds and 2 seconds. I/O operations are enabled.

For both `RESET HPIL` and `INIT:3`, the following are set to zero: the interrupt mask (`ENABLE INTR`), the interrupt-cause byte (`READINTR`), and the device status byte (`REQUEST`). The device-dependent command number (`READDDC`) is set to `-1`.

The following table summarizes the information described above: (A “—” indicates that no change occurs.)

Reset Conditions

	Turn On	RESTORE I/O	CONTROL ON	RESET HPIL	INIT:3
Controller/device	—	—	controller	controller	controller
DISPLAY IS device	active*	active*	active	—	default
PRINTER IS device	active*	active*	active	—	default
Device addresses	next operation*†‡	assigned*	assigned	next operation†	assigned
“Busy” devices	—	cleared	cleared	—	—
Assign codes	—	—	—	—	cancelled
Flags —21,—22, —23,—24	—	—	—	—	cleared
I/O channels	—	—	—	—	cancelled
OFF I/O condition	—	cancelled	(error)§	—	cancelled
Local/Remote mode	—	—	—	Local	Local
STANDBY parameters	—	—	—	60,2	60,2
Interrupt-cause byte	—	—	—	0	0
Interrupt mask	—	—	—	0	0
HP-71 device status	—	—	—	0	0
Device-dependent command number	—	—	—	—1	—1

* Only if HP-71 is controller.

† Only if flag —24 is clear. Occurs immediately if **DISPLAY IS** device is defined.

‡ Only if flag —21 was clear when the HP-71 was turned off.

§ **CONTROL ON** requires I/O enabled.

System Flag Summary

The HP-IL interface uses four HP-71 system flags. Their effects are summarized below.

System Flags

Flag	Purpose	Clear	Set
–21	Inhibit Power Down	HP-71 turns off HP-IL devices (with capability) when turned off; considers old addresses invalid.	HP-71 doesn't affect devices when it's turned off; considers old address still valid.
–22	Extended Addressing	HP-71 uses simple addressing for all devices.	HP-71 uses extended addressing for all devices with capability, and simple addressing for remaining devices. (Uses only simple addressing if ASSIGN IO used.)
–23	End Of Transmission	End Of Transmission doesn't affect ENTER statement.	End Of Transmission terminates variable assignment for ENTER statement while HP-71 is controller.
–24	Inhibit Addressing	HP-71 automatically assigns new addresses to devices if previous addresses become invalid.	HP-71 doesn't automatically assign new addresses to devices. (Only ASSIGN IO, RESTORE IO, and CONTROL ON assign new addresses.)

System Memory Requirements

The following table lists the amount of main memory needed for I/O functions.

Memory Requirements for HP-IL Interface

Item	Memory Requirements for HP-IL Interface
ASSIGN IO codes	64½ bytes for one or more codes (up to 30 codes). If ASSIGN IO *, no memory is used.
I/O channels	35 bytes for each open file, plus 259½ bytes for each file on an external device. (Up to 64 files can be open at once.)
DISPLAY IS device	13 bytes if device specifier is device ID or volume label; 0 bytes otherwise.
PRINTER IS device	13 bytes if device specifier is device ID or volume label; 0 bytes otherwise.
Configuration overhead	14 bytes.

HP-71 Responses as a Device

Section 4 includes a general description of the operation of the HP-71 as a device in the loop. The following table describes how the HP-71 responds to individual HP-IL messages. The messages are divided into Command Group, Ready Group, Identify Group, and Data/End Group. For most applications, you do not need to be aware of this level of operation.

Responses to HP-IL Messages

HP-IL Message	HP-71 Response
Command Group	
Auto Address Unconfigure	Address set to 21.
Device Clear	Clears input and output buffers. (Data in the buffers is lost.)*
Device Dependent Listener 0–31	If listener, stores message number (plus 32) as READDDC number.*
Device Dependent Talker 0–31	If talker, stores message number as READDDC number.*
Enable Asynchronous Requests	When enabled, HP-71 will request service (by sending Identify—Service Request messages) according to device status (set by REQUEST).
Enable Listener Not Ready	If listener, enables the HP-71 to send a Not Ready For Data message whenever it receives a line feed character or End Byte message during an ENTER operation.
Go To Local	If listener, HP-71 set to Local mode (until next made a listener).
Group Execute Trigger	No response.*
Interface Clear	Clears controller, listener, and talker status.*
Listen Address 0–31	If address matches, clears talker status and becomes a listener.* If address is 31, clears listener status.
Local Lockout	If Remote enabled, sets bit in STATUS byte.
Not Remote Enable	Sets HP-71 to Local mode and prevents it from returning to Remote mode. Clears Local Lockout bit and Remote mode bit in STATUS byte.
Parallel Poll Disable	If listener, does not respond to subsequent parallel polls (does not modify Identify messages).
Parallel Poll Enable 0–15	If listener, modifies subsequent Identify messages according to parallel poll conventions. (Refer to page 218.)
Parallel Poll Unconfigure	Does not respond to subsequent parallel polls (does not modify Identify messages).

Responses to HP-IL Messages (continued)

HP-IL Message	HP-71 Response
Remote Enable	Enables HP-71 to begin operating in Remote mode whenever it next becomes a listener.
Secondary Address 0–30	<p>If HP-71 has an extended address, if the secondary address matches, and if it follows a Talk Address or Listen Address message with matching primary address, then HP-71 becomes a talker or listener.*</p> <p>If HP-71 has an extended address, if the secondary address doesn't match, and if it follows a Talk Address message with matching primary address, then HP-71 clears talker status.</p>
Selected Device Clear	If listener, clears input and output buffers. (All data in the buffers is lost.)*
Talk Address 0–31	<p>If address matches,† clears listener status and HP-71 becomes a talker.</p> <p>If address doesn't match, clears talker status.</p>
Unlisten	Clears listener status.
Untalk	Clears talker status.
Ready Group	
Auto Address 0–31	<p>If HP-71 has an earlier assigned address, no response.</p> <p>If message address is 31, no response.</p> <p>If message address is less than 31 and HP-71 does not have an earlier assigned address, sets HP-71 address to message address, increments message address by one, and passes revised message to next device.</p>
Auto Extended Primary 0–31	<p>If HP-71 has earlier assigned address, no response.</p> <p>If message address is 31, no response.</p> <p>If not preceded by Auto Extended Secondary message, no response.</p> <p>If preceded by Auto Extended Secondary 31, no response.</p> <p>If preceded by Auto Extended Secondary less than 31, if message address less than 31, and if HP-71 doesn't have earlier assigned address, then HP-71 primary address is set to message address.</p>
Auto Extended Secondary 0–31	<p>If HP-71 has earlier assigned address, no response.</p> <p>If message address is 31, no response.</p> <p>If message address is less than 31 and HP-71 doesn't have earlier assigned address, HP-71 secondary address is set to message address, increments message address by one, and passes revised message. (Must be followed by Auto Extended Primary message to establish valid device address.)</p>

Responses to HP-IL Messages (continued)

HP-IL Message	HP-71 Response
End Of Transmission—Error	If talker, HP-71 sends this as soon as it detects a data error in loop.
End Of Transmission—OK	If talker, sends this if HP-71 receives a Not Ready For Data message.
Not Ready For Data	If talker, HP-71 makes the previous data byte the last byte sent.
Ready For Command	No response. (Passes message to next device in the loop after executing previous command.)
Send Accessory ID	If talker, sends one byte with value of 3.‡
Send Data	If talker, can send data.*‡
Send Device ID	If talker, sends six ASCII-coded characters (bytes): HP71(CR)(LF).‡
Send Status	If talker, sends status information defined by the REQUEST statement.‡
Take Control	If talker, HP-71 accepts control of the loop.*‡
Identify Group	
Identify Identify—Service Request	If HP-71 is set to respond by Parallel Poll Enable message, modifies message according to parallel poll setup and service request status. (Refer to page 218.)
	If HP-71 requires service, message is modified to Identify—Service Request message.
Data/End Group	
Data Byte	If talker, sends next data byte.‡
Data Byte—Service Request	If listener, accepts data byte and passes it to the next device.
	If HP-71 requires service, message is modified to Data Byte—Service Request message.
End Byte	If talker, sends next data byte.‡
End Byte—Service Request	If listener, accepts data byte and passes it to the next device.
	If HP-71 requires service, message is modified to End Byte—Service Request message.
* These messages can cause an interrupt by setting bits in the ENABLE INTR statement.	
† If HP-71 has extended address, the message address must match the primary address. The response occurs only if the correct Secondary Address message follows.	
‡ Indicates that a message different from the received message is sent to the next device in the loop.	

The HP-71 can be enabled to respond to a parallel poll. A parallel poll allows the HP-IL controller to determine which devices require attention. When the HP-71 receives an HP-IL Parallel Poll Enable message, the HP-71 is assigned one bit number and is set to modify that bit in a particular way during subsequent parallel polls. Up to eight devices can respond to a parallel poll.

The parallel poll consists of an HP-IL Identify 0 message sent by the HP-IL controller. If the HP-71 has been parallel poll enabled, it modifies all Identify messages according to the table below. The HP-71 can only *set* its designated bit in Identify messages. Basically, if the HP-71 is enabled by a Parallel Poll Enable message 0 through 7, setting a bit to “1” means “no service request”; if the bit is not changed, service is requested. If the HP-71 is enabled by a Parallel Poll Enable message 8 through 15, setting a bit to “1” means “service request”; if the bit is not changed, service is not requested. In all cases, a “service request” condition is indicated by placing a “1” in the Service Request control bit in the Identify message.

If the HP-71 receives a Parallel Poll Unconfigure message, or if the HP-71 is a listener and receives a Parallel Poll Disable message, the HP-71 won’t respond to subsequent parallel polls—that is, it doesn’t modify Identify messages.

Parallel Poll Response to Identify Messages

Enable message ...	designates bit ...	and sets that Identify bit if ...
Parallel Poll Enable 0	D ₀	service is not requested*
Parallel Poll Enable 1	D ₁	
Parallel Poll Enable 2	D ₂	
Parallel Poll Enable 3	D ₃	
Parallel Poll Enable 4	D ₄	
Parallel Poll Enable 5	D ₅	
Parallel Poll Enable 6	D ₆	
Parallel Poll Enable 7	D ₇	
Parallel Poll Enable 8	D ₀	service is requested*
Parallel Poll Enable 9	D ₁	
Parallel Poll Enable 10	D ₂	
Parallel Poll Enable 11	D ₃	
Parallel Poll Enable 12	D ₄	
Parallel Poll Enable 13	D ₅	
Parallel Poll Enable 14	D ₆	
Parallel Poll Enable 15	D ₇	

* Otherwise, the designated identify bit isn’t changed. Also, control bit C₀ is set if service is requested.

Appendix C



















































HP-71 Character Set











































The following table shows the HP-71 character set and the ASCII (American Standard Code for Information Interchange) character set. Where keystrokes are shown to the right of an HP-71 character, you can use either those keystrokes or the `CHR$` function to display the character. Where no keystrokes are shown to the right of a character, you can use only the `CHR$` function to display that character. (In most cases, the default display characters in the right side of the table are duplicates of the display characters in the left side of the table—only characters 136, 138, 141, and 155 differ.)

ASCII characters 0 through 31 and character 127 in the following table are defined as special control characters. These characters are used primarily in data communications to control peripheral devices.

Character Code	Binary	ASCII Character	HP-71 Character	HP-71 Keystrokes	Character Code	Binary	Default HP-71 Character*
0	00000000	(NUL)	(none)	CTRL @	128	10000000	(space)
1	00000001	(SOH)	␣	CTRL A	129	10000001	␣
2	00000010	(STX)	␣	CTRL B	130	10000010	␣
3	00000011	(ETX)	␣	CTRL C	131	10000011	␣
4	00000100	(EOT)	␣	CTRL D	132	10000100	␣
5	00000101	(ENQ)	␣	CTRL E	133	00000101	␣
6	00000110	(ACK)	␣	CTRL F	134	10000110	␣
7	00000111	(BEL)	␣	CTRL G	135	10000111	␣
8	00001000	(BS)	(none)†	CTRL H	136	10001000	␣
9	00001001	(HT)	␣	CTRL I	137	10001001	␣
10	00001010	(LF)	(none)†	CTRL J	138	10001010	␣
11	00001011	(VT)	␣	CTRL K	139	10001011	␣
12	00001100	(FF)	␣	CTRL L	140	10001100	␣
13	00001101	(CR)	(none)†	CTRL M	141	10001101	␣
14	00001110	(SO)	␣	CTRL N	142	10001110	␣
15	00001111	(SI)	␣	CTRL O	143	10001111	␣
16	00010000	(DLE)	␣	CTRL P	144	10010000	␣
17	00010001	(DC1)	␣	CTRL Q	145	10010001	␣
18	00010010	(DC2)	␣	CTRL R	146	10010010	␣
19	00010011	(DC3)	␣	CTRL S	147	10010011	␣
20	00010100	(DC4)	␣	CTRL T	148	10010100	␣
21	00010101	(NAK)	␣	CTRL U	149	10010101	␣

Character Code	Binary	ASCII Character	HP-71 Character	HP-71 Keystrokes	Character Code	Binary	Default HP-71 Character*
22	00010110	(SYN)	␣		150	10010110	␣
23	00010111	(ETB)	␣		151	10010111	␣
24	00011000	(CAN)	␣		152	10011000	␣
25	00011001	(EM)	␣		153	10011001	␣
26	00011010	(SUB)	␣		154	10011010	␣
27	00011011	(ESC)	(none)*		155	10011011	␣
28	00011100	(FS)	␣	(none)	156	10011100	␣
29	00011101	(GS)	␣		157	10011101	␣
30	00011110	(RS)	␣		158	10011110	␣
31	00011111	(US)	␣	(none)	159	10011111	␣
32	00100000	(space)	(space)		160	10100000	(space)
33	00100001	!	!		161	10100001	!
34	00100010	"	"		162	10100010	"
35	00100011	#	#		163	10100011	#
36	00100100	\$	\$		164	10100100	\$
37	00100101	%	%		165	10100101	%
38	00100110	&	&		166	10100110	&
39	00100111	'	'		167	10100111	'
40	00101000	((168	10101000	(
41	00101001))		169	10101001)
42	00101010	*	*		170	10101010	*
43	00101011	+	+		171	10101011	+
44	00101100	,	,		172	10101100	,
45	00101101	-	-		173	10101101	-
46	00101110	.	.		174	10101110	.
47	00101111	/	/		175	10101111	/
48	00110000	0	0		176	10110000	0
49	00110001	1	1		177	10110001	1
50	00110010	2	2		178	10110010	2
51	00110011	3	3		179	10110011	3
52	00110100	4	4		180	00110100	4
53	00110101	5	5		181	10110101	5
54	00110110	6	6		182	10110110	6
55	00110111	7	7		183	10110111	7
56	00111000	8	8		184	10111000	8
57	00111001	9	9		185	10111001	9
58	00111010	:	:		186	10111010	:
59	00111011	;	;		187	10111011	;
60	00111100	<	<		188	10111100	<
61	00111101	=	=		189	10111101	=
62	00111110	>	>		190	10111110	>

Character Code	Binary	ASCII Character	HP-71 Character	HP-71 Keystrokes	Character Code	Binary	Default HP-71 Character*
63	00111111	?	?	 	191	10111111	?
64	01000000	@	@	 	192	11000000	@
65	01000001	A	A		193	11000001	A
66	01000010	B	B		194	11000010	B
67	01000011	C	C		195	11000011	C
68	01000100	D	D		196	11000100	D
69	01000101	E	E		197	11000101	E
70	01000110	F	F		198	11000110	F
71	01000111	G	G		199	11000111	G
72	01001000	H	H		200	11001000	H
73	01001001	I	I		201	11001001	I
74	01001010	J	J		202	11001010	J
75	01001011	K	K		203	11001011	K
76	01001100	L	L		204	11001100	L
77	01001101	M	M		205	11001101	M
78	01001110	N	N		206	11001110	N
79	01001111	O	O		207	11001111	O
80	01010000	P	P		208	11010000	P
81	01010001	Q	Q		209	11010001	Q
82	01010010	R	R		210	11010010	R
83	01010011	S	S		211	11010011	S
84	01010100	T	T		212	11010100	T
85	01010101	U	U		213	11010101	U
86	01010110	V	V		214	11010110	V
87	01010111	W	W		215	11010111	W
88	01011000	X	X		216	11011000	X
89	01011001	Y	Y		217	11011001	Y
90	01011010	Z	Z		218	11011010	Z
91	01011011	[[ 	219	11011011	[
92	01011100	\	\	(none)	220	11011100	\
93	01011101]]	 	221	11011101]
94	01011110	^	^	 	222	11011110	^
95	01011111	_	_	(none)	223	11011111	_
96	01100000	`	`	(none)	224	11100000	`
97	01100001	a	a	 	225	11100001	a
98	01100010	b	b	 	226	11100010	b
99	01100011	c	c	 	227	11100011	c
100	01100100	d	d	 	228	11100100	d
101	01100101	e	e	 	229	11100101	e
102	01100110	f	f	 	230	11100110	f
103	01100111	g	g	 	231	11100111	g

Character Code	Binary	ASCII Character	HP-71 Character	HP-71 Keystrokes	Character Code	Binary	Default HP-71 Character*
104	01101000	h	h	 	232	11101000	h
105	01101001	i	i	 	233	11101001	i
106	01101010	j	j	 	234	11101010	j
107	01101011	k	k	 	235	11101011	k
108	01101100	l	l	 	236	11101100	l
109	01101101	m	m	 	237	11101101	m
110	01101110	n	n	 	238	11101110	n
111	01101111	o	o	 	239	11101111	o
112	01110000	p	p	 	240	11110000	p
113	01110001	q	q	 	241	11110001	q
114	01110010	r	r	 	242	11110010	r
115	01110011	s	s	 	243	11110011	s
116	01110100	t	t	 	244	11110100	t
117	01110101	u	u	 	245	11110101	u
118	01110110	v	v	 	246	11110110	v
119	01110111	w	w	 	247	01110111	w
120	01111000	x	x	 	248	11111000	x
121	01111001	y	y	 	249	11111001	y
122	01111010	z	z	 	250	11111010	z
123	01111011	{	{	 	251	11111011	{
124	01111100			(none)	252	11111100	
125	01111101	}	}	 	253	11111101	}
126	01111110	~	~	(none)	254	11111110	~
127	01111111	(DEL)	␣	(none)	255	11111111	␣

* You can use the `CHARSET` statement to change the display character symbol for one or more of the default display characters corresponding to character codes 128 through 255.

† Performs an HP-71 display function.

Mass Storage Compatibility

This appendix presents information about the format of mass storage files used by the HP-71. You may not need to use this information unless you want to share mass storage files between the HP-71 and another computer. The first topic, “Compatibility With Other Hewlett-Packard Computers,” describes how certain types of files can be converted for use with various Hewlett-Packard computers. The second topic, “Mass Storage Format,” defines the format of HP-71 media so that you can evaluate compatibility with other computers, and gives information about storage requirements for storing numeric and string values in mass storage files.

Compatibility With Other Hewlett-Packard Computers

A mass storage medium may be recorded by one computer and later may be accessed by another computer. If the formats used by both computers are compatible, each computer may be able to use files recorded by the other.

The HP-71 provides three types of mass storage data files:

- **TEXT** (or LIF1). This type of file has variable-length records that contain ASCII data. This type is compatible with HP-75 LIF1 files, HP-41 AS (ASCII) files, and LIF (type 1) files from other computers.
- **DATA**. This type of file has fixed-length records that can contain numeric and string values. It allows both sequential and random access. This type isn’t compatible with data files from other Hewlett-Packard computers.
- **SDATA**. This type of file has fixed-length, eight-byte records that can contain numeric values. (The HP-71 can also read six-character ASCII strings from an SDATA file, but it can’t store them in an SDATA file.) This type is compatible with HP-41 DA (data) files.

HP-71 BASIC program files aren’t directly compatible with other computers because the HP-71 has unique coding for its operations. However, you *can* share program files by transforming them into data files—after a computer reads such a data file, it could transform the file back into a program file. For example, the HP-71 can use `TRANSFORM` to transform a BASIC file into a TEXT file, and vice versa.

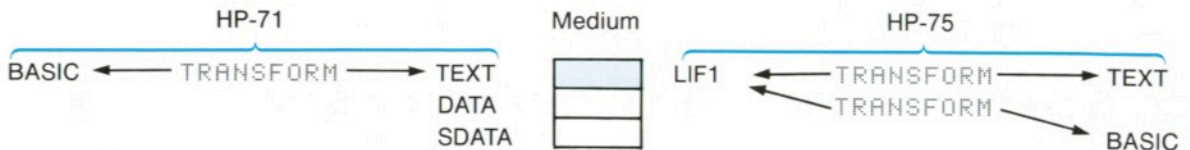
The following topics discuss compatibility with specific Hewlett-Packard computers.

Series 80 Compatibility

Series 80 DATA files aren't compatible with HP-71 DATA files. This is because Series 80 computers use a special directory format for storing DATA files on mass storage media. The file length stored in bytes 28 and 29 of the directory is expressed as *bytes*, whereas the HP-71 requires this parameter to be expressed as *records*. (Refer to "Mass Storage Format" on page 226.) Otherwise, the format of Series 80 DATA files is the same as that of HP-71 DATA files.

HP-75 Compatibility

The HP-71 TEXT file format is compatible with the HP-75 LIF1 file format. These file types enable the HP-71 and HP-75 to share data and programs stored on a medium by using the TRANSFORM statement to convert file types, as illustrated in the following diagram.



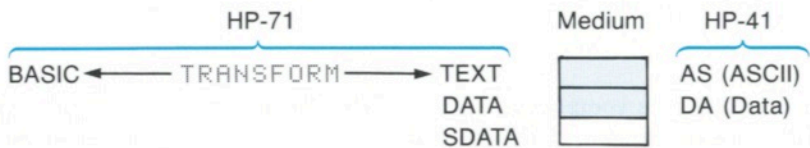
An HP-75 LIF1 file stored on a mass storage medium is treated as a TEXT file by the HP-71, and vice versa.

Note that the HP-75 and HP-71 can't share "data" files. However, the HP-71 TEXT file is a data-type file that the HP-75 can use, as described above. These files require that numeric values be stored and read as strings.

Even though HP-75 LIF1 files have a format that's compatible with HP-71 TEXT files, the HP-75 imposes an additional requirement for making an LIF1 file usable. Each record must start with a four-digit line number, and the line numbers must be stored in increasing order. The HP-75 requires line numbers so that the files can be transformed.

HP-41 Compatibility

The HP-71 TEXT file format is compatible with the HP-41 AS (ASCII) file format (an extended-functions format). The HP-71 SDATA file format is compatible with the HP-41 DA (data) file format. These file types enable the HP-71 and HP-41 to share data stored on a medium. The HP-71 can use the TRANSFORM statement to convert file types, as illustrated in the following diagram.



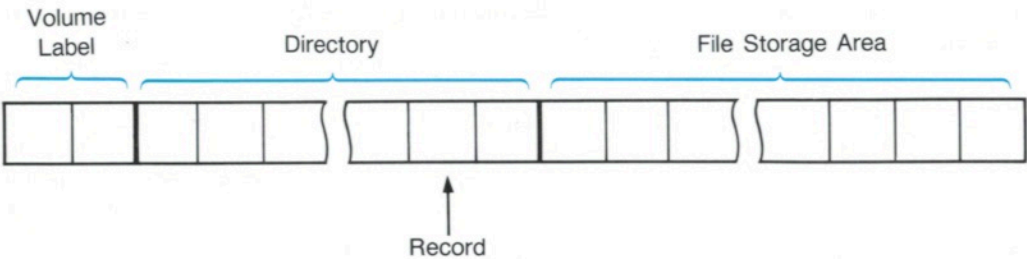
An HP-41 AS file stored on a mass storage medium is treated as a TEXT file by the HP-71, and vice versa. Similarly, an HP-41 DA file is treated as an SDATA file, and vice versa.

Although the HP-71 can read “alpha” (string) data from an SDATA file, the HP-71 can’t store string data in such a file.

Mass Storage Format

The HP-71 stores information on a mass storage medium according to the Logical Interchange Format (LIF) standard. This standard provides for three sections on the medium, each containing a different type of information:

- Volume label.
- Directory.
- File storage area.



Each *record* (or “sector”) on the medium consists of 256 eight-bit bytes, numbered 0 through 255.

The following paragraphs describe the LIF standard and show how the HP-71 uses the sections defined by the standard. Within the volume label and directory, any item that contains more than one byte (as described below) has the most significant byte first (except as noted).

Volume Label. The volume label provides identification information and gives the location of the directory. It occupies the first and second records on the medium—records 0 and 1.

Volume Label

Byte	Definition	Required by HP-71	HP-71 Byte Values
Record 0			
0-1	LIF ID (value=32768)	Yes	128 0
2-7	Label	*	x x x x x x
8-11	Directory Start Record	Yes	0 0 0 2
12-13	System 3000	No	16 0
14-15	Unused (value=0)	No	0 0
16-19	Directory Length	Yes	0 0 x x
20-21	Version	Yes	0 1
22-23	Unused (value=0)	No	0 0
24-27	Tracks per Surface	†	0 0 0 2
28-31	Number of Surfaces	†	0 0 0 1
32-35	Records per Track	†	0 0 1 0
36-41	Date and Time Initialized	No	x x x x x x
42-255	Extensions and Maintenance	No	0 0 ... 0
Record 1			
0-255	System 3000 (values=0)	No	0 0 ... 0
* Required only when volume label is specified in mass storage operation.			
† Required only if version is nonzero.			

Bytes 2 through 7 indicate the label of the medium, which differs from medium to medium.

Bytes 8 through 11 indicate the record at which the directory starts, which is record 2 for HP-71 media.

Bytes 16 through 19 indicate the number of records in the directory, which is fixed when the medium is initialized, but varies from medium to medium.

Bytes 24 through 35 indicate characteristics of the medium: number of tracks per surface, number of surfaces, and number of records per track.

Bytes 36 through 41 indicate the date and time at which the medium was initialized. Each byte represents two binary-coded-decimal digits, each represented by four bits. Bytes 36 through 41 indicate the year, month, date, hour, minute, and second, respectively. Each parameter is a two-digit number.

Directory. The directory provides information for labeling, describing, and locating files stored on the medium. Each record in the directory can contain eight entries, arranged sequentially within the record. The entries start at bytes 0, 32, 64, 96, 128, 160, 192, and 224, with each directory entry containing 32 bytes. The 32 bytes within each entry are defined by the table below.

Directory Entry

Byte	Definition	Required by HP-71	HP-71 Byte Values					
0-9	File Name	Yes	x	x	x	x	...	x
10-11	File Type	Yes	x	x				
12-15	File Start Record	Yes	0	0	x	x		
16-19	File Length	Yes	0	0	x	x		
20-25	Date and Time Created	*	x	x	x	x	x	x
26-27	Volume Flag/Number	Yes	128	1				
28	Implementation	Yes	x					
29	Implementation	Yes	x					
30	Implementation	Yes	x					
31	Implementation	Yes	x					

* Required only for CAT and CAT# operations.

Bytes 0 through 9 give the file name as character codes for up to 10 characters—they must form a valid HP-71 file name (letters in uppercase). Unused characters must be spaces.

Bytes 10 and 11 define the type of file according to the table below, which lists the types of files used by the HP-71.

File Types

Byte 10	Byte 11	Decimal Value*	Type	File Description
0	0	0	—	Purged File
0	1	1	TEXT	TEXT Data File
224	208	−7984	SDATA	SDATA Data File
224	213	−7979	TEXT	Secure TEXT File
224	240	−7952	DATA†	DATA Data File
226	4	−7676	BIN†‡	Binary Program File
226	8	−7672	LEX†	Language Extension File
226	12	−7668	KEY†	Key Definition File
226	20	−7660	BASIC†‡	BASIC Program File
255	255	−1	—	End of Directory

* Decimal value is 2's complement of 16-bit word (bytes 10 and 11).

† Secure file increases byte 11 value and decimal value by 1.

‡ Private file increases byte 11 value and decimal value by 2.

Bytes 12 through 15 and 16 through 19 give the record number at which the file begins and the number of records that the file spans. The file usually doesn't fill the last record entirely.

Bytes 20 through 25 give the time (year, month, date, hour, minute, and second) the file was created. Each byte represents two binary-coded-decimal digits, each represented by four bits. Each parameter is a two-digit number.

Bytes 28 through 31 give various types of information, depending upon the file type, as defined by the following table.

Implementation Bytes				
File Type	Byte 28	Byte 29	Byte 30	Byte 31
TEXT	0	0	0	0
SDATA	← File Length	→ Security		0
DATA	← File Length*	→ ← Record Length*	→	
BIN	← File Length*	→		0
LEX	← File Length*	→		0
KEY	← File Length*	→		0
BASIC	← File Length*	→		0

File length for SDATA file is number of records or HP-41 registers (16-bit number); for DATA file is number of records (16-bit number); for other files is number of nibbles, or half-bytes (24-bit number). Record length is number of bytes in a record (16-bit number).

* Least significant byte is first.

Files. The file type indicated in the directory entry defines the format in which information is stored in the file storage area. The following table summarizes the number of bytes required by the HP-71 to store a numeric or string variable in a mass storage data file.

Mass Storage Data Requirements

File Type	Numeric Variable	String Variable
TEXT	1 byte per character (for number expressed in current round-off setting) plus 2 bytes, plus 1 additional byte if number of bytes is odd.	1 byte per character plus 2 bytes, plus 1 additional byte if number of bytes is odd.
DATA	8 bytes.	1 byte per character plus 3 bytes.
SDATA	8 bytes.	(Not possible.)

Appendix E

Error Messages

The error messages listed in the following tables relate only to HP-IL interface operations. For other error or warning messages, refer to the *HP-71 Reference Manual*.

This appendix contains two listings:

1. An alphabetical listing of error messages with their corresponding error numbers. (Error messages that correspond to more than one number are followed by a brief description of the various types of errors.) You can use this listing to determine the number of any HP-IL error message (so you can look it up in the next listing).
2. A numerical listing of error messages with a description of each message.

An error appears in your display in the form: HPIL ERR: *error message*. (If the error occurs in a program, the error message includes the program line number.) To display the error number of the most recent error message, execute ERRN. Other error-related functions are ERRL (error line) and ERRM\$ (error message). (Refer to the *HP-71 Reference Manual*.)

Alphabetical Message Listing

Message	Error Number
Aborted	255052
ASSIGN IO Needed	255001
Data Type	255054
Device Not Found	255032
Device Not Ready	255034
Device Type	255047
Directory Full	255031
End of File	255027
End of Medium	255017
Excess Chars	255003
File Exists	255030
File Not Found	255022
File Protect	255016
Insufficient Memory	255059
Invalid Arg	255056
Invalid Device Spec	255053

Message	Error Number
Invalid Expr	255006
Invalid Medium	
(Data checksum error detected)	255026
(Mass storage drive motor stalled)	255018
(Medium not initialized)	255024
(Medium not initialized to proper format)	255019
(Opened and closed door of mass storage drive)	255023
(Record number error)	255025
Invalid Mode	255041
Invalid Parm	255005
Loop Broken	
(Loop is not complete)	255035
(Partial message received)	255042
(Timeout period exceeded)	255043
Message Error	
(Message altered during transmission)	255038
(Message lost due to slow retransmission)	255037
(Too many messages received)	255036
(Too many messages received)	255040
Missing Parm	255004
No Loop	255057
No Medium	255020
RESTORE IO Needed	255060
Self Test Failed	255045
Size of File	255028
Syntax	255007
System Error	255044
Unexpected Message	255039

Numerical Message Listing and Descriptions

Parse Errors (255001 through 255007)

Error Number	Message and Condition
255001	<code>ASSIGN IO Needed</code> Attempted to execute a <code>LIST IO</code> statement without having executed an <code>ASSIGN IO</code> statement. Execute <code>ASSIGN IO</code> .
255003	<code>Excess Chars</code> More characters were found in a statement than expected. Check syntax.
255004	<code>Missing Parm</code> A parameter required by the statement is missing. Check syntax.
255005	<code>Invalid Parm</code> A parameter used in the statement is not legal. Check parameters.
255006	<code>Invalid Expr</code> The expression cannot be evaluated due to invalid type (such as a string variable occurring in an expression that requires a numeric value). Check expression.
255007	<code>Syntax</code> The statement is not recognized. Verify spelling and required parameters.

Mass Storage Errors (255016 through 255031)

Error Number	Message and Condition
255016	<code>File Protect</code> The file is secure or private, and an improper operation was attempted. If secure, execute <code>UNSECURE</code> statement.
255017	<code>End of Medium</code> File is too big for available space on medium; medium is full; drive error condition. Check medium; recreate file; pack medium; use another medium.
255018	<code>Invalid Medium</code> Mass storage drive motor stalled. Check for jammed medium.
255019	<code>Invalid Medium</code> Medium is not initialized to proper format. Execute <code>INITIALIZE</code> statement.
255020	<code>No Medium</code> No medium detected in mass storage device. Check that drive door is closed; insert medium.

Mass Storage Errors (continued)

Error Number	Message and Condition
255022	<p>File Not Found</p> <p>The specified file was not found; specified file name differs from directory entry (file name) received from non-mass storage device. Check catalog and file name.</p>
255023	<p>Invalid Medium</p> <p>Opened and closed door of mass storage drive during file operation. Restart operation.</p>
255024	<p>Invalid Medium</p> <p>Medium is not initialized. Execute <code>INITIALIZE</code> statement.</p>
255025	<p>Invalid Medium</p> <p>Record number in directory doesn't match record number on medium. Retry operation. If it fails again, initialize medium (execute <code>INITIALIZE</code>), then recreate file(s).</p>
255026	<p>Invalid Medium</p> <p>Data checksum error detected. Initialize medium; recreate file(s).</p>
255028	<p>Size of File</p> <p>File is too large to copy in or out of mass storage device. Add memory to HP-71 or new medium to mass storage unit.</p>
255030	<p>File Exists</p> <p>The file name specified in a <code>CREATE</code> statement or as the destination in a <code>COPY</code> statement already exists. Purge old file or change name of new or old file.</p>
255031	<p>Directory Full</p> <p>The directory on a medium is full. Purge unwanted files; pack directory or medium.</p>

Loop Errors (255032 through 255060)

Error Number	Message and Condition
255032	<p>Device Not Found</p> <p>A requested device was not found in loop. Check device specifier; check system setup; execute <code>RESTORE IO</code>.</p>
255034	<p>Device Not Ready</p> <p>A device did not respond when expected (such as not sending data or not accepting control). Check device specifier; check device; execute <code>RESTORE IO</code>.</p>
255035	<p>Loop Broken</p> <p>Loop is not complete. Check HP-IL connections; check that devices are turned on.</p>
255036	<p>Message Error</p> <p>Too many messages received by HP-71. Restart operation.</p>

Loop Errors (continued)

Error Number	Message and Condition
255037	<p>Message Error</p> <p>Message lost due to slow retransmission. Restart operation.</p>
255038	<p>Message Error</p> <p>Message altered during transmission. Restart operation.</p>
255039	<p>Unexpected Message</p> <p>HP-IL protocol violation occurred (more than one talker was active in loop at same time); talker indicated transmission error. Restart operation.</p>
255040	<p>Message Error</p> <p>Too many messages received by HP-71. Restart operation.</p>
255041	<p>Invalid Mode</p> <p>Attempted to execute a controller statement while the HP-71 was acting as a device. Check the mode (controller or device) required by the statement.</p>
255042	<p>Loop Broken</p> <p>A partial message was received due to a retransmission error. Restart operation.</p>
255043	<p>Loop Broken</p> <p>A message took longer than the STANDBY timeout period to go around loop. Clear listeners; restart operation.</p>
255044	<p>System Error</p> <p>Device addresses probably invalid (if flag -24 set). Clear flag -24 or assign new addresses (execute RESTORE IO).</p> <p>Internal error related to I/O channels. Execute RUN and restart operation; perform INIT:2; perform INIT:3. If error persists, the HP-71 requires repair service.</p>
255045	<p>Self Test Failed</p> <p>Interface failed internal self-test. Repeat self-test by executing RESET HPIL. If the error persists, the HP-IL interface is defective and requires repair service.</p>
255047	<p>Device Type</p> <p>The specified device is not of a legal type for the statement. Check device type requirements.</p>
255052	<p>Aborted</p> <p>Pressed ATTN ATTN to interrupt operation. Execute RESTORE IO; if necessary, execute RESET HPIL, then RESTORE IO. Check HP-IL connections; check that devices are turned on.</p>
255053	<p>Invalid Device Spec</p> <p>The device specifier is not valid for the statement. Check device specifier.</p>

Loop Errors (continued)

Error Number	Message and Condition
255054	<code>Data Type</code> Specified the wrong type of variable (numeric or string). Change argument to proper variable type.
255056	<code>Invalid Arg</code> An argument is out of the allowable range. Check argument value. Directory entry (start record or length) received during mass storage operation is improper. Re-store file.
255057	<code>No Loop</code> Interface isn't installed. Check system setup.
255059	<code>Insufficient Memory</code> Not enough main RAM to perform required operation. Add memory; delete files or key assignments; reallocate internal RAM.
255060	<code>RESTORE IO Needed</code> Attempted an I/O operation after executing <code>OFF IO</code> . Execute <code>RESTORE IO</code> .

Glossary

This glossary defines terms that are used throughout this manual. Refer also to the *HP-71 Reference Manual* for descriptions of other terms that relate to the HP-71.

Although some terms can be applied to several types of interfacing systems, in this glossary the terms are defined primarily in relation to HP-IL concepts.

A

Accessory ID: 1. A numeric identifier that you can use to specify a device in an HP-IL system. It is a numeric expression that should match the accessory ID (see definition 2) of the desired device. It's preceded by a % symbol, and may be followed by an optional sequence number. 2. A numeric identifier that a device sends when requested by the controller. A device need not have an accessory ID.

Acoustic coupler: A device that converts between standard interface signals (such as from HP-IL) and acoustical signals that are used by a standard telephone handset. It may provide two-way communication. See also *Modem*.

Address: 1. A number that you can use to specify a device in an HP-IL system. It is a numeric expression that's rounded to two decimal places—an integer indicates a simple address, a non-integer indicates an extended address. 2. An identifying number that's usually assigned to a device by the controller. (The HP-71 automatically does this.) By assigning a different address to each device (the usual case), the controller can designate individual devices for HP-IL operations. See also *Simple addressing* and *Extended addressing*.

ASCII: American Standard Code for Information Interchange, a convention that associates characters (including control characters) with numeric character codes.

Assign code: A code that you can use to specify a device in an HP-IL system. It is a string expression (or unquoted string) that evaluates to either one letter or else a letter followed by a letter or digit. (Because an assign code may look like a variable name, it's good practice to avoid using it as an unquoted string.)

Asynchronous service request: A service request that is indicated when a device sends an HP-IL message on an idle HP-IL system—that is, the device initiates the message. This differs from the normal service request situation in which the device modifies a message that the controller or talker has sent. An asynchronous service request condition must be set up by the controller.

B

Bit: The smallest unit of information—one binary digit. Two possible values, “0” and “1”, can represent the presence or absence of a condition or that a state is true or false.

Busy device: A peripheral device that’s been set by the controller as a talker or listener—it’s “busy” with an information transfer. Some devices indicate this condition with a BUSY light.

Byte: A fundamental unit of information, which is usually represented as either eight binary digits or a decimal number (from 0 through 255). For example, each HP-71 character, each ASCII character, and each HP-IL Data Byte or End Byte message represents one byte of data.

C

Channel: A path through which the HP-71 stores and retrieves information from a data file. An I/O buffer is associated with each channel that accesses a mass storage data file.

Character code: A numeric value that corresponds to a character. Each HP-71 display character has a corresponding character code (values 0 through 255); each ASCII character has a corresponding character code (values 0 through 127). (The two character sets are *not* identical.)

Character-oriented device: A device that treats data as individual characters or lines of characters and interprets that data as information to be processed (printed, displayed, or transmitted). (Differs from a file-oriented device).

Close (a file): To update a mass storage data file according to data in its corresponding I/O buffer and cancel the I/O channel for that file.

Control character: A subset of the ASCII character set that is frequently used in data communications to control peripheral devices (rather than to produce a displayed or printed character, for example). These characters correspond to character codes 0 through 31 and 127.

Controller: The HP-IL unit that’s responsible for issuing instructions to other units in the HP-IL system. Only one unit can be the controller—remaining units are often referred to as *devices*. The role of controller can be transferred to another device, but there can be only one controller at any time.

D

Data file: A file that contains numeric or string information: a DATA file, an SDATA file, or a TEXT (LIF1) file.

Device: 1. A peripheral device—a device that's *not* the controller, but rather is controlled by the controller. 2. Any unit connected in the HP-IL system; a general term that includes peripheral devices and controllers.

Device-dependent message: An HP-IL message whose meaning depends upon the design of the device receiving it. Many devices don't respond to device-dependent messages.

Device ID: 1. An identifying string that you can use to specify a device in an HP-IL system. It is a string expression (or unquoted string) that should match the device ID (see definition 2) of the desired peripheral device. It may be followed by an optional sequence number. 2. An identifying string that a device sends when instructed by the controller. A device need not have a device ID.

Device specifier: Any of several identifiers that you can use to specify a device in an HP-IL system: accessory ID, address, assign code, device ID, device word, and volume label.

Device status: Status information that expresses a device's condition using conventions defined for that device. The most significant bit of the first device status byte must be "0". See also *Status* and *System status*.

Device word: A reserved HP-71 word that you can use to specify a device in an HP-IL system. It may be followed by an optional sequence number.

Directory: The portion of a mass storage medium that contains information about files stored on the medium. Information about each file is contained in its directory entry.

Directory entry: A 32-byte field, located in the directory portion of a mass storage medium, that gives detailed information about an individual file, such as names, sizes, and locations of the files on the medium.

Display device: A character-oriented device designed to produce video output—specifically, a device with an accessory ID in the range 48 to 63 (although some devices may not have accessory ID's).

Displayed output: Data sent by the HP-71 to the DISPLAY IS device, regardless of whether that device is a standard display device or any other type of device. For example, such output is generated by the DISP, CAT, and LIST statements.

E

End-of-line sequence: A series of characters that is sent at the end of a line of data. They're used to indicate the end of the line or to cause a device to perform a function, such as to control the location of the next output line on a printer. It's specified by the `ENDLINE` statement.

Extended addressing: A method of assigning addresses to peripheral devices in which the address of each device consists of a primary part and a secondary part. Although each part is an integer, the HP-71 expresses an extended address as a fractional number *pp.ss*. (The secondary part of an HP-71 extended address is 1 greater than the number of the HP-IL Secondary Address message.)

F

Field: A sequence of one or more symbols (characters) in a format string that defines how one output or input item is to be expressed when it's sent or received.

File: A collection of program lines or data that can be manipulated as a single unit and has a single name.

File-oriented device: A device that treats data as encoded file information (directory information and file contents) and merely stores and retrieves that information—it doesn't attempt to interpret it. (Differs from a character-oriented device.)

File storage area: The portion of a mass storage medium that's used for storing the contents of files.

Format string: A string of symbols (characters) in an `IMAGE` statement or `USING` option that defines how data is to be expressed when it's sent or received. The string defines a series of fields, each of which corresponds to one output or input item.

Function: A routine built into the HP-71 that may operate on arguments and produces a single numeric or string value.

G

GPIO: General-Purpose Input/Output, an interfacing system that uses parallel transmission of data and control signals between two devices.

H

HP-IB: Hewlett-Packard Interface Bus, a standardized interfacing system that uses parallel transmission of data and control signals in a parallel configuration. It is Hewlett-Packard's implementation of IEEE Standard 488.

HP-IL: Hewlett-Packard Interface Loop, an interfacing system that uses bit-serial transmission of information around a loop configuration.

I

Idle device: A device that's not a talker or listener—that is, it's not involved in a data transfer. Although a device may be idle, it still responds to instructions issued by the controller and still passes HP-IL messages around the loop.

Initialize: To format a mass storage medium so that it's ready to store information. For the HP-71, this operation includes erasing the entire medium, recording the volume label, and allocating space for the directory and file storage area.

Interface device: A device that converts signals used by two interface systems. For example, it could convert between HP-IL signals and RS-232 signals, between HP-IL and HP-IB signals, or between HP-IL and telephone signals (modem). It may provide two-way communication.

Interrupt: An event that the HP-71 recognizes when it occurs and can cause program execution to branch in a predefined way. HP-IL interrupt events cause branching only at the ends of program lines, whereas timer and error interrupts cause branching after any statement.

Interrupt-cause byte: A number maintained by the HP-71 that indicates which HP-IL interrupt events have occurred.

Interrupt mask: A number maintained by the HP-71 that specifies which HP-IL interrupt events are able to cause interrupt branching.

I/O: Input/output, which refers to processes that use an interface (such as HP-IL) to interact with other devices, especially the processes of receiving and sending data.

K

Keyword: A word having a predefined meaning to the HP-71—usually a statement, function, or operator.

L

LIF standard: Logical Interchange Format standard, a Hewlett-Packard standard that defines a format for encoding and storing information on mass storage media.

Listener: A device that's been assigned by the controller to receive data from the talker. Several devices can be listeners at one time. The controller can be a listener.

Local Lockout: An operating condition of a peripheral device in which the device can't respond to its "local" controls—keys or switches. The device receives all instructions from the controller. This condition can exist only while the device is Remote enabled. Some devices don't implement the Local Lockout condition.

Local mode: An operating condition of a peripheral device in which the device responds to its “local” controls—keys or switches. Certain interface devices reserve Local mode for transferring data between systems. Some devices don’t implement Local mode (or Remote mode).

Loop: The HP-IL system, in which the cables connect all devices in a loop (or ring) configuration.

M

Mass storage device: A file-oriented device designed to store and retrieve information on a medium—specifically, a device with an accessory ID in the range 16 to 31 (although some devices may not have accessory ID’s).

Medium: A (usually) removable component, used in a mass storage device, that is capable of storing data. For example, digital cassettes and discs are mass storage media.

Message (HP-IL): The fundamental element of communication among HP-IL devices. It consists of 11 bits that convey one instruction or one byte of data. Most operations require that numerous messages (instructions and data bytes) be sent on HP-IL.

Modem: A device that converts between standard interface signals (such as from HP-IL) and standard electrical telephone signals. It may provide two-way communication. See also *Acoustic coupler*.

O

Open (a file): To associate an I/O channel number with a data file and establish an I/O buffer for transferring data to or from a mass storage file.

P

Parallel poll: A method of fetching operating information from peripheral devices in which the controller receives only service request information from each device. All of the information is contained in one message.

Pass control: To give up the responsibility for controlling the HP-IL system and turn over that responsibility to another device.

Peripheral (device): A device that’s connected to other devices using a standard interface (such as HP-IL) and that’s controlled by one of those devices (the controller).

Printer device: A character-oriented device designed to produce “paper-copy” output—specifically, a device with an accessory ID in the range 32 to 47 (although some devices may not have accessory ID’s).

Printed output: Data sent by the HP-71 to the `PRINTER IS` device, regardless of whether that device is a standard printer or any other type of device. Such output is generated by the `PRINT` and `PLIST` statements.

Private file: A file that can only be executed and copied (into main RAM), but that can't be changed or inspected.

Protocol: A set of rules that define proper communication procedures. A violation of protocol may disrupt communication.

R

Record: 1. A “physical” record (or sector) on a mass storage medium is a 256-byte storage area. 2. A “logical” record on a mass storage medium is the area set aside in a data file for storing each data item or sequence of data items. It isn't necessarily the same as a “physical” record.

Remote enabled: An operating condition of a peripheral device in which the device may be in Remote mode or in Local mode (unless Local Lockout is in effect). In this condition, when a device becomes a listener, it changes to Remote mode. If a device is *not* Remote enabled, it may be only in Local mode. Some devices don't implement Remote and Local modes, and therefore aren't affected by the Remote enabled condition.

Remote mode: An operating condition of a peripheral device in which the device responds to instructions received from a “remote” source—another device connected to HP-IL. Some devices don't implement Remote mode (and Local mode).

Round-off setting: Any one of four settings that determine how the computer displays and sends numbers—STD, FIX, SCI, ENG.

RS-232: An interfacing system developed by the Electronic Industries Association. It uses bit-serial transmission of information between two devices, often in combination with control signals on separate lines.

S

Secure file: A file that can be executed, copied, read, and renamed, but that can't be changed or purged.

Sequence number: A numeric expression enclosed in parentheses that indicates the *n*th occurrence of the preceding device specifier. It's an optional parameter.

Serial poll: A method of fetching operating information from peripheral devices in which the controller receives complete status information from each device, one after another.

Service request: A special indication from a device to the controller. It tells the controller that the device requires attention from the controller for an unspecified reason. Certain HP-IL messages can convey a service request in addition to their normal information.

Simple addressing: A method of assigning addresses to peripheral devices in which the address of each device is an integer.

Statement: An instruction that can begin a line of a BASIC program or be executed from the keyboard.

Status: Information that conveys the operating condition of a device.

Syntax: Rules that govern the construction of statements and functions and govern the spelling of keywords, variable names, and other instruction components.

System status: Status information that expresses a device's condition as one of several conditions defined by HP-IL conventions. The most significant bit of a system status byte must be "1". See also *Status* and *Device status*.

T

Talker: An HP-IL device that's been assigned by the controller to send data to other devices. Only one device at a time can be a talker. The controller can be a talker.

Timeout period: The length of time the HP-71 waits for each single HP-IL message to travel around the loop and back to the HP-71. Messages that convey instructions usually are delayed by a device that must respond to the instructions.

Trigger (a device): To cause an unspecified event to occur at one or more devices. The nature of the event depends upon the individual device. Some devices don't have a response to being triggered.

V

Verify interval: The time interval at which the HP-71 checks the loop's continuity.

Volume label: 1. An identifying string that you can use to specify a device in an HP-IL system. It is a string expression (or unquoted string) that should match the volume label (see definition 2) of the medium in the desired device (although a medium need not have a volume label). It must begin with a letter and may have up to six letters or digits. It is preceded by a . (period). 2. A sequence of characters stored in the volume label portion (see definition 3) of a mass storage medium. It serves as a "label" for that medium. 3. The portion of a mass storage medium that stores the volume label string and other information about the medium format and directory.

Subject Index

Page numbers in **bold type** indicate primary references; page numbers in regular type indicate secondary references.

A

Accessory ID

- as device specifier, 17, **20–21**
- definition, **74, 238**
- fetching from device, **44, 109–110**
- finding, **20, 109–110**
- of HP-71 device, **56, 62, 217**
- relates to device word, **19**

Accessory type, definition, **74**

Acoustic couplers, **25, 238**

Address, HP-IL

- as device specifier, 17, **18, 39**
- definition, **76, 238**

Addresses. *See also* Extended addresses; Simple addresses

- affected by flag –21, **16**
- affected by flag –22, **39–40**
- affected by flag –23, **214**
- affected by flag –24, **41**
- assigned when restoring operation, **177**
- assigned when taking control, **54, 99**
- assigning, **18, 39–41**
- associated with assign codes, **80–81**
- effect of assigning on HP-IL messages, **186–187**
- effects of reset operations, **175, 212–213**
- finding, **18, 40, 107–108**
- listing with assign codes, **129**
- of HP-71 device, **55–56**
- schemes for assigning, **39–41**
- usually assigned at turn on, **15–16**

Addressing, **39–41**. *See also* Addresses

American Standard Code for Information Interchange. *See* ASCII

AND binary function, **69, 82–83**

ASCII

- character code conventions, **68**
- character set, **220–223**
- data stored in TEXT files, **93**
- definition, **238**

Assign codes

- as device specifiers, 17, **18–19**
- affect addressing, **40**
- cancelling, **81**
- definition, **74, 238**
- listing, **19, 129**
- memory requirements, **214**
- operation using, **80–81**
- suspending use, **135**

Asynchronous service requests

- affect interface status, **197**
- definition, **238**
- enabling, **215**
- purpose, **49–50**
- sent by HP-71 device, **174**

ATTN **ATTN** keystroke, **16–17, 177, 205**

Auto addresses. *See* Simple addresses

B

BASIC commands, receiving from controller, **55, 59–62, 167**

BASIC files

- compatibility, **224–226**
- format of directory entry, **228–230**
- listed in catalog, **93**
- making private, **157**

Binary data, image symbols, **123, 145**

Binary files

- format of directory entry, **228–230**
- listed in catalog, **93**
- making private, **157**

Binary numbers, using, **68–70**

Bit binary function, **69, 90–91**

Branching. *See* Interrupt branching

Buffer, I/O. *See* I/O buffer

C

Cables, HP-IL, connecting, 13

CALC mode
 doesn't use `DISPLAY IS` device, 113
 executing keywords in, 72, 108, 109, 189
 prevents receiving BASIC commands, 60, 167

Catalog
 operation for mass storage files, 92-93
 string function operation, 94-95

Channel, I/O. *See* I/O channel

Character codes
 definition, 239
 HP-71 and ASCII, 220-223
 represent characters, 68

Character-oriented devices
 definition, 239
 receive and send data, 36, 38
 used for printed and displayed output, 25, 113, 155

Characters
 ASCII set, 220-223
 comma, 141
 control, 220, 239
 for numeric input, 119, 125
 HP-71 set, 220-223
 line feed, 119, 121, 123, 125, 125, 141, 215
 semicolon, 141

Classes of devices. *See* Device word; Accessory ID

Clearing controller condition. *See* Controller

Clearing devices. *See* Devices

Codes, assign. *See* Assign codes

Codes, character. *See* Character codes

Comma, used in free-field output, 141, 142-143

Compatibility, mass storage, 224-230

Complement binary function, 69, 84-85

Computer, remote, interfacing with HP-IL, 38-39

Concatenating statements, 73

Connecting the HP-IL system, 12-13

Continuity of system, checking, 46, 194, 205

Control of HP-IL
 acquiring, 53-54
 acquiring is interrupt event, 116
 giving up, 52-53, 98-99, 152-153
 passing, 51-54, 152-153, 243
 requesting, 54
 taking, 54, 98-99, 152-153, 175

Control characters, 220, 239

Controller. *See also* Controller, HP-71
 accessory ID's, 62
 clearing, 52, 99, 177
 controls system, 15, 239
 instructing HP-71 device, 55, 59-62
 passing responsibility of, 52, 152-153

Controller, HP-71. *See also* Controller
 affects interface status, 197
 clearing, 98
 clearing affects interrupt-cause byte, 164
 establishing, 17, 98-99
 becoming is interrupt event, 116
 normal condition, 34, 35, 52, 55, 98, 153
 operation, 35-51
 reset condition, 175

Copying files. *See* Files, mass storage

Current file, default for copying, 101, 102

D

Data. *See also* Numeric data; String data
 affects interface status, 197
 affects interrupt-cause byte, 164
 conveyed by HP-IL messages, 14
 fetching, 38-39, 118-126
 formatted, 36, 38, 56, 58, 121-124, 141, 143-147
 interrupt events, 116
 receiving by HP-71 device, 58-59, 118-126
 sending, 35-37, 140-147
 sending by HP-71 device, 56-58, 140-147
 terminating entry, 119-120, 125
 transferring between devices, 187, 197

Data files, mass storage. *See also* Files, mass storage
 accessing, 31
 closing, 30, 31, 78, 79, 180
 closing, definition, 239
 compatibility, 224-230
 creating, 30, 104-106
 definition, 240
 listed in catalog, 93
 opening, 30, 36, 38, 77-79, 105
 reading from, 30, 38, 78
 record length, 30
 restoring pointer, 30
 securing and closing, 180
 size can't be increased, 30
 storage requirements, 230
 types, 30
 writing to, 30, 36, 78

DATA files. *See also* Data files, mass storage
 compatibility, 224-230
 data storage requirements, 230
 format of directory entry, 228-230

Destination, for copying files, 101

Device, HP-71
 automatically processes messages, 55
 clearing affects interrupt-cause byte, 164
 clearing is interrupt event, 116

Device, HP-71 (*continued*)

- copying files, 56, 58
- establishing, 98–99
- executing keywords, 72
- interaction with controller, 59
- operation, 34, 55–68
- responses to HP-IL messages, 215–218
- sending asynchronous service requests, 174
- sending device information, 62–63
- status, 62, 171–174, 175, 212–213, 217
- taking address, 55–56

Device-dependent messages

- definition, 240
- interrupt event, 116
- number of, 65, 160–162, 175, 212–213, 215
- receipt affects interrupt-cause byte, 164
- receipt is interrupt event, 47, 64, 161
- sending, 183

Device ID

- as device specifier, 17, 20
- definition, 74, 240
- fetching from device, 44, 111
- finding, 20, 111
- of HP-71 device, 56, 62, 217

Device specifiers

- definition, 75, 240
- types, 17–21, 39

Device status

- definition, 240
- meaning, 189, 190–191
- relation to HP-71 device, 172–173

Device type, definition, 75

Device word

- definition, 75, 240
- related to accessory ID, 109–110
- as device specifier, 17, 19–20

Devices

- accessory ID's of, 109–110
- addresses of, 107–108
- changing modes, 42
- classes of, 109–110
- clearing, 17, 42, 96–97, 183
- connecting, 13
- controlling conditions, 41–44
- definition, 240
- device ID's of, 20, 111
- don't control system, 15
- fetching information from, 44–46
- HP-IL status of, 97, 99
- interaction with HP-71, 14–15
- passing control to, 152–153
- role when idle, 15
- setting to Local Lockout condition, 132–133

- setting to Local mode, 130–131
- setting to Remote mode, 166–168
- specifying, 17–21. *See also* Device specifiers
- status of, 44, 48, 52, 63, 68, 188–192, 191, 192
- transferring data between, 187
- triggering, 42, 44, 199–200
- turning on and off, 14, 16, 183, 214

Directory

- definition, 240
- determining size, 29
- packing, 32, 148–149, 150–151
- section of medium, 29, 226, 228–230
- setting up on medium, 128

Directory entries

- how operations affect, 159, 170
- checked before copying files, 101, 102–103
- definition, 240
- format on medium, 228–230
- relation to stored files, 150

Display, HP-71

- duplicating, 23, 113
- sending output to, 135, 156

DISPLAY IS device. *See also* Display operations

- activating, 99, 177
- assigning, 23–24, 112–114
- cancelling, 81, 112–113
- changing, 23
- choosing, 23, 25
- effect on HP-IL messages, 186
- effects of reset operations, 212–213
- memory requirements, 214
- relating to passing control, 153
- startup assignment, 26
- suspending use, 135
- three categories, 113
- using, 24–25, 113–114, 129

Display operations, 22–26. *See also*

- DISPLAY IS device
- using only HP-71 display, 23
- using PRINTER IS device, 23

E

End Byte messages

- affect interface status, 197
- convey service requests, 48
- during data transfer, 119, 121, 125, 141, 215
- sending, 185
- terminate data entry, 119
- terminate variable assignment, 121

End Of Transmission messages

- don't terminate data entry, 120
- response of HP-71 device, 217
- sending, 187

use affected by flag -23, 214
 during data transfer, 68, 125-126, 142, 156
 End-of-line branching. *See* Interrupt branching
 End-of-line sequence
 definition, 241
 output image symbol, 146
 sending using HP-IL messages, 185
 while sending data, 36, 141, 143
 Entering data, operation, 118-126
 Erasing a medium, 128
 Error line, 232
 Error messages, 232-237
 Error numbers, 232-237
 Examples
 Binary operations on interrupt mask, 69-70
 Catalog of files on printer, 24
 Checking cassette drives for tapes, 45-46
 Copying files between HP-71's, 59
 Data file operations, 30
 Decimal to binary function, 70
 Device-dependent message service requests,
 66-67
 DISPLAY IS device assignments, 23-24
 Fetching data from remote computer, 38-39
 File operations, 29-30
 Key definitions to control display, 25
 Passing control in a program, 53
 PRINTER IS device assignments, 22-23
 Printing on HP-IB printer, 37
 Program listed on printer, 24
 Requesting control in a program, 54
 Responding to interrupts from multimeter,
 48-49
 Sending data to RS-232 device, 42-43
 Service requests for sending and receiving data,
 64
 Setting print width, 45
 Simple and extended addressing, 41
 Status of HP-71 device, 63
 Transferring data between HP-71's, 57-58
 Triggering multimeter readings, 43-44
 Using BASIC commands to copy files between
 HP-71's, 61-62
 Using BASIC commands to transfer data be-
 tween HP-71's, 61
 Exclusive-OR binary function, 69, 86-87
 Execute-only files, listed in catalog, 93
 Extended addresses
 affected by flag -22, 214
 definition, 241
 finding, 107-108
 for HP-71 device, 55, 216
 setting using HP-IL messages, 183
 using, 40-41

F

File creation, 93, 95
 File name
 changing, 169-170
 definition, 75
 for creating file, 30, 105
 format on medium, 228
 in catalog, 92, 95
 usually required for copying, 102
 File-oriented devices
 definition, 241
 not for printed and displayed output, 25, 113,
 155
 used for mass storage operations, 36, 38, 78,
 102, 149, 151
 File protection
 for private files, 157
 for secure files, 179-180, 201
 in catalog, 93, 95
 File record length, 30, 102, 105
 File size, 30, 93, 95, 102, 105
 File specifier, definition, 76
 File storage area
 data storage requirements, 230
 packing, 148-149, 151
 relates to directory entries, 150
 section of medium, 29, 226, 230
 File type
 for creating file, 30, 105
 for copied file, 102
 format on medium, 228-230
 in catalog, 93, 95
 Files, main-RAM, 29, 30, 31, 78, 79, 93, 95,
 100-103, 106, 157, 158, 159, 170, 180, 201
 Files, mass storage. *See also* Data files, mass
 storage
 creating while copying, 102
 definition, 241
 catalog, 30, 92-93, 94-95
 copying, 29, 36, 38, 56, 100-103
 copying as HP-71 device, 56
 defining number on medium, 128
 execute-only, 157
 format standard, 224-230
 purging, 31, 148, 150, 159
 relate to directory entries, 150
 renaming, 169-170
 security, 29, 30, 157-158, 159, 179-180, 201
 sending and receiving, 29, 36, 38, 56, 100-103
 transforming, 224
 Flag -21, power-down, 16, 41, 50, 81, 177, 186,
 212-213, 214

Flag —22, addressing, 39–40, 50–51, 212–213, 214
 Flag —23, End Of Transmission, 119, 124–126, 212–213, 214
 Flag —24, readdressing, 41, 177, 214
 Format, mass storage, 224–230
 Format string
 definition, 241
 for entering data, 119, 121
 for sending data, 141, 143
 repeating, 124, 147
 Formatted data
 entering, 38, 58, 121–124
 sending, 36, 56, 141, 143–147
 Free-field input, 119, 120–121, 125
 Free-field output, 141, 142–143
 Functions, class of keywords, 72

H

Hewlett-Packard Interface Loop. *See* HP-IL
 HP 3468A Multimeter, 43, 48, 199
 HP 82161A Digital Cassette Drive, 28, 45, 128
 HP 82162A Thermal Printer, 45, 97
 HP 82163 Video Interface, 24
 HP 82164A HP-IL/RS-232-C Interface, 38–39, 41, 42–43
 HP 82169A HP-IL/HP-IB Interface, 37
 HP 82905B Printer, 38–39, 43, 44
 HP Series 80, mass storage compatibility, 225
 HP-41
 AS file compatibility, 105, 224, 226
 DA file compatibility, 105, 224, 226
 mass storage compatibility, 226
 registers, 105
 HP-75
 LIF1 file compatibility, 105, 224, 225
 mass storage compatibility, 225
 HP-IB printer, using with HP-IL, 37
 HP-IB system, interfacing with HP-IL, 37
 HP-IL, definition, 241
 HP-IL address. *See* Address; Addresses
 HP-IL cables, connecting, 13
 HP-IL interface
 care of, 204
 checking operation, 205
 connecting, 12–13
 installing, 12–13
 resetting, 175–176
 self-test, 175, 205
 HP-IL interrupts. *See* Interrupts, HP-IL
 HP-IL messages. *See* Messages, HP-IL

HP-IL operation
 restored by assign codes, 81
 restoring, 15, 81, 135, 177–178
 suspending, 15, 99, 135–136, 177
 suspending prevents taking control, 99
 HP-IL port (in HP-71), 12
 HP-IL protocol. *See* Protocol, HP-IL
 HP-IL system
 checking response, 46
 controlling conditions, 41–44
 operation, 14–15
 resetting, 16–17
 turning on and off, 15–16
 HP-IL timeout. *See* Timeout period; Verify interval

I

I/O buffers, used for mass storage data files, 31, 78
 I/O channels, access mass storage data files, 31, 77–79, 214, 239
 I/O operations. *See* HP-IL operations
 Identify messages, 48, 185, 197
 Idle devices, role in system, 15
 Image field, 121, 143
 Image symbols, 121–124, 143–147
 Inclusive-OR binary function, 69, 88–89
 Initializing a medium, 28–29, 127–128, 242
 Input buffer
 clearing, 116, 183, 215, 216
 clearing affects interrupt-cause byte, 164
 clearing using HP-IL messages, 183
 interaction with BASIC commands, 60
 used during data entry, 120
 Input operations. *See also* HP-IL operations
 entering data, 34, 38–39, 118–126
 may depend upon previous operation, 114
 of HP-71 device, 58–59, 118–126
 Instructions, conveyed by HP-IL messages, 14
 Interface Clear messages, 116, 164, 183, 215
 Interface devices, 19, 25, 110
 Interface status (of HP-71 interface)
 affected by service requests, 52, 192
 checking during data transfer, 187
 indicates controller, 52
 reading, 48, 65, 196–198
 set at reset, 175
 Interrupt branching
 can respond to interrupt events, 47, 65
 can turn on HP-71, 47
 cancelling, 47, 65, 134, 138
 caused by enabled interrupt events, 47, 64, 65
 defining, 47, 65, 137–139
 only within program, 138
 responding to service requests, 192

- subroutine guidelines, 139
- unexpected, 139
- Interrupt-cause byte
 - affected by service requests, 192
 - clearing, 163, 175
 - definition, 242
 - effects of reset operations, 212–213
 - in example, 49, 66–67
 - meaning, 163–165
 - operation, 138–139
 - reading value of, 47, 65, 163–165
- Interrupt events
 - affect interrupt branching, 47, 65, 137–139
 - affect interrupt-cause byte, 163, 165
 - disabled at reset, 175
 - disabled by branching, 116, 139
 - enabling, 47, 115–117
 - indicates service request, 52
 - list, 47, 64–65, 116
 - recognized by HP-71 controller, 47
 - recognized by HP-71 device, 64
 - related to interrupt branching, 134
 - responding to service requests, 192
 - types of events, 47, 54, 64–65, 116, 161
- Interrupt mask
 - defining, 115–117, 138
 - definition, 242
 - effects of reset operations, 212–213
 - enables interrupt events, 115–117
 - related to interrupt-cause byte, 138, 165
 - related to interrupt branching, 138–139
 - resetting, 116, 138, 175
- Interrupting an operation, 16
- Interrupts, error, relation to other interrupts, 139
- Interrupts, HP-IL. *See also* Interrupt branching; Interrupt-cause byte; Interrupt events
 - definition, 242
 - events causing, 47, 64–65, 116
 - operation, 47–50, 64–67, 138–139
 - operation for HP-71 device, 64–67, 138–139
 - processing, 138–139
 - relation to other interrupts, 139
 - subroutine guidelines, 139
- Interrupts, timer, relation to other interrupts, 139

K

- Key definition files, 93, 229–230
- Key definitions, display example, 25
- Keyboard, executing keywords from, 72
- Keyboard, receiving instructions from, 55
- keys file, copying, 102

L

- Language extension files, 93, 157, 229–230
- LEX files, 93, 157, 229–230
- LIF
 - definition, 242
 - standard for mass storage format, 28, 226–230
 - used by TEXT files, 93, 105
 - used for copying files, 102
- LIF1 files. *See* TEXT files
- Line feed character
 - during data entry, 119, 121, 125, 215
 - end-of-line character, 141
 - input image symbol, 123
 - not using for data entry, 125
- Listener, HP-71
 - affects interface status, 197
 - affects interrupt-cause byte, 164
 - becoming, 215
 - becoming is interrupt event, 116
- Listeners, 15, 183, 242
- Listing assign codes, operation, 129
- Local Lockout condition
 - affects interface status, 197
 - definition, 242
 - response of HP-71 device, 197, 215
 - setting devices, 42, 132–133
- Local mode
 - definition, 243
 - for HP-71 device, 60, 215
 - in example, 43, 44, 49, 61, 62
 - interaction with Local Lockout, 132
 - related to Remote mode, 167
 - setting devices to, 42, 130–131
 - setting using HP-IL messages, 183
- Logical Interchange Format standard. *See* LIF
- Loop, definition, 76. *See also* HP-IL system
- Loop errors, 235–237
- LOOP option
 - for copying files, 101, 103
 - for entering data, 38, 120
 - for passing control, 153
 - for printed output, 155–156
 - for sending data, 36, 141–142
 - for sending HP-IL messages, 186

M

- Main-RAM files, 29, 30, 31, 78, 79, 93, 95, 100–103, 106, 157, 158, 159, 170, 180, 201
- Mask, interrupt. *See* Interrupt mask
- Masking bits, using binary functions, 69
- Mass storage compatibility, 224–230
- Mass storage devices, identifying, 21, 28
- Mass storage errors, 234–235

Mass storage operations, 17, 28–32

Medium

- definition, 243
- erasing, 128
- information format, 28, 226–230
- initializing, 28–29, 127–128
- packing, 31–32, 148–149
- relation to volume label, 17
- three sections, 28–29, 226–230

Memory requirements, summary, 214

Messages, HP-IL

- checking transmission, 46
- command group, 183–184
- control bits, 183
- convey information, 14
- convey service requests, 48
- data bits, 183
- data/end group, 183, 185
- definition, 243
- effect on data entry, 119, 120
- identify group, 183, 185
- ready group, 183, 185, 187
- sending by HP-71 device, 67–68
- sending individually, 50–51, 181–187
- sent during keyword execution, 73
- with HP-71 device, 55, 67–68, 181–187

Model number of device, relates to device ID, 20

Modems, 25, 243

N

Not Remote enabled condition, 130–131, 167, 215

Numeric characters, during data entry, 119, 120

Numeric data

- entering, 120, 124, 125
- input image symbols, 122
- output image symbols, 144–145
- sending free-field, 142–143

O

Ones-complement binary function, operation, 84–85

OR binary function, 69, 86–89

Output buffer

- clearing, 116, 183, 215, 216
- clearing affects interrupt-cause byte, 164
- clearing using HP-IL messages, 183
- used during output operations, 142, 156

Output operations. *See also* HP-IL operations

- may depend upon previous operation, 114
- of HP-71 device, 56–58, 140–147
- sending data, 34, 35–37, 56–58, 140–147
- using `PRINTER IS` device, 155

Overflow condition, for formatted data, 147

P

Packing a directory, 32, 150–151

Packing a medium, 31–32, 148–149

Parallel poll

- definition, 243
- indicates service request, 63
- responding to service requests, 192
- response of HP-71 device, 215, 217, 218

Parse errors, 234

Passing control of HP-IL. *See* Control of HP-IL

Peripheral devices, 243. *See also* Devices

Poll. *See* Parallel poll; Serial poll

Port, HP-IL (in HP-71), 12

Primary addresses. *See* Extended addresses

`PRINTER IS` device. *See also* Printer operations

- assigning, 22–23, 154–156
- cancelled by assign codes, 81
- cancelling, 26, 155, 156
- changing, 22
- choosing, 22, 25
- effects of reset operations, 212–213
- identifying in program, 45
- memory requirements, 214
- need not be a printer, 22
- restoring use, 177
- setting to `DISPLAY IS` device, 155
- startup assignment, 26
- suspending use, 135
- using, 24–25, 35

Printer operations. *See also* `PRINTER IS` device

- performing, 22–26, 35, 155
- suppressing printing, 22
- using `DISPLAY IS` device, 22

Private files

- definition, 244
- listed in catalog, 93
- making private, 157–158
- security, 180, 201

Program execution

- affected by interrupt branching, 138
- during data output, 142, 156
- during data entry, 120
- finding addresses during, 108

Program files, 157–158, 224

Programmable keywords, 73

Protocol, HP-IL

- definition, 244
- governs messages sent, 68
- messages must follow, 50
- references, 50, 68, 185

Q

Quote marks, single versus double, 74

R

- Radio/television interference, 210
- Random access, provided by DATA files, 105
- Receiving data. *See* Data, receiving
- Record length
 - for creating file, 30, 105
 - for copied file, 102
- Remote enabled condition
 - relation to Local and Remote modes, 131, 167
 - definition, 244
 - required for Local Lockout, 132
 - response of HP-71 device, 216
- Remote mode
 - affects interface status, 197
 - definition, 244
 - for HP-71 device, 60, 216
 - in example, 43, 49, 61, 62
 - interaction with Local Lockout, 132
 - related to Local mode, 130, 131
 - setting devices to, 42, 166–168
- Repair information, 207–210
- Requests, service. *See* Service requests
- Reset
 - effect on HP-IL messages, 187
 - mentioned, 16, 35, 52, 55, 98, 113, 117, 131, 138, 142, 153, 155, 156, 164, 172
 - of a “stuck” system, 16
 - operation, 175–176
 - runs self-test, 205
 - summary of effects, 212–213
- Roles of devices, 15
- Round-off setting, 142, 244
- RS-232 system, interfacing with HP-IL, 38–39, 42–43

S

- SDATA files. *See also* Data files, mass storage
- can't be secure, 180
 - compatibility, 224–230
 - creating, 104–106
 - data storage requirements, 230
 - format of directory entry, 229–230
 - listed in catalog, 93
- Secondary addresses. *See* Extended addresses
- Secure files
 - effects of security, 102, 159, 179–180
 - definition, 244
 - listed in catalog, 93
 - making private, 157
 - securing, 179–180
- Self-test, HP-IL interface, 175, 205
- Semicolon, used in free-field output, 141, 142–143
- Sending data. *See* Data, sending
- Sending files. *See* Files, mass storage
- Sequence number, 19, 20, 21, 76, 244
- Serial access, provided by DATA files, 105
- Serial poll
 - conducting, 188–192
 - definition, 244
 - response of HP-71 device, 171, 174
 - used with service requests, 63, 192
- Service (repair) information, 207–210
- Service requests
 - asynchronous, 49–50, 174, 197, 215
 - by HP-71 device, 63–64, 172, 174, 217
 - can request control, 52
 - definition, 244
 - effects on interrupt-cause byte, 138, 164
 - indicated by parallel poll, 218
 - indicated by system status, 189–190
 - receipt affects interface status, 197
 - receipt is interrupt event, 47, 116
 - responding to, 48–50, 192
- Simple addresses
 - affected by flag –22, 214
 - always used with assign codes, 81
 - definition, 244
 - finding, 107–108
 - for HP-71 device, 55
 - using, 39–41
- Source, for copying files, 101
- Specifying devices, 17–21. *See also* Device specifiers
- Standard terms, definitions, 74–76
- Startup conditions, setting devices to, 97
- Statements, class of keywords, 72, 73
- Status. *See also* Device status; System status
 - definition, 245
 - fetching from devices, 44, 188–192
 - HP-IL interface. *See* Interface status
 - multiple bytes, 191
 - of devices. *See* Devices, status of
 - of HP-71 device, 62, 63, 171–174
- Stream data files. *See* SDATA files
- String data
 - entering, 121, 124, 125
 - input image symbols, 123
 - output image symbols, 145
 - sending free-field, 142–143
- “Stuck” system, resetting, 16
- Syntax, definition, 245
- Syntax diagrams, reading, 73–74
- System status
 - affected by service requests, 192
 - definition, 245

System status (*continued*)

- meaning, 173, 189–190
- priorities, 173, 189–190
- relation to HP-71 device, 172, 173
- used by some devices, 44

T

Talker, HP-71

- affects interface status, 197
- becoming, 216
- interrupt-cause byte, 164
- interrupt event, 116

Talkers, 15, 183, 245

Temperature limits, for interface, 204

TEXT files

- compatibility, 224–230
- creating, 104–106
- data storage requirements, 230
- format of directory entry, 229–230
- listed in catalog, 93

Timeout period

- definition, 245
- effects of reset operations, 212–213
- operation, 193–195
- set at reset, 175
- setting, 46, 193–195

Transforming files, 224

Triggering devices

- affects interrupt-cause byte, 164
- definition, 245
- is interrupt event, 47, 65, 116

- operation, 42, 44, 199–200

Turn on, can be caused by interrupt event, 47

Turn on and off, affects system, 15–16

U

Unsecure files, purging, 159

USING option

- for entering data, 38, 58, 119
- for HP-71 device, 56, 58
- for sending data, 36, 56, 141

V

Verify interval

- definition, 245
- effects of reset operations, 212–213
- operation, 193–195
- set at reset, 175
- setting, 46, 193–195

Verifying interface operation, 205

Volume label

- definition, 76, 245
- section of medium, 28, 226, 227–228
- specifier for mass storage operations, 17, 21, 29
- writing on medium, 128

W

Warranty information, 205–207

workfile file, copying, 102

Keyword Index and Summary

Keyword	Page		Description
	Part I	Part II	
System Setup			
ASSIGN IO	18	80	Associates assign codes with HP-IL devices.
LIST IO	19	129	Lists all defined assign codes and their HP-IL addresses.
OFF IO	15	135	Suspends I/O operation.
RESET HPIL	16	175	Resets the HP-IL interface to a known condition.
RESTORE IO	15	177	Enables I/O operations to occur on HP-IL.
Printer and Display Operations			
DISPLAY IS	23	112	Assigns one HP-IL device to be the display device.
PRINTER IS	22	154	Assigns one HP-IL device to be used for all printing operations.
Mass Storage Operations			
ASSIGN #	30	77	Associates an I/O channel number with a file and opens the file.
CAT	29	92	Gives a catalog of file information.
CAT\$	29	94	Returns a string containing catalog information.
COPY	29	100	Copies a file from one location to another.
CREATE	30	104	Creates a data file.
INITIALIZE	28	127	Initializes a mass storage medium.
PACK	32	148	Packs directory and storage space on a medium.
PACKDIR	32	150	Packs only directory space on a medium.
PRIVATE	29	157	Permanently prevents a file from being changed or inspected.
PURGE	29	159	Deletes a file.
RENAME	29	169	Changes the name of a file.
SECURE	29	179	Prevents a file from being altered or purged.
UNSECURE	29	201	Cancels security for a file.
General I/O Operations—Data Transfer			
ENTER	38	118	Reads data from HP-IL into numeric and string variables.
OUTPUT	35	140	Sends data from numeric and string expressions to HP-IL.

Keyword**Page**
Part I Part II**Description****General I/O Operations—HP-IL Interaction**

CLEAR	17	96	Clears an individual HP-IL device or all HP-IL devices.
DEVADDR	18	107	Returns the address of a device.
DEVAID	20	109	Returns the accessory ID of a device.
DEVID\$	20	111	Returns a string containing the device ID of a device.
ENABLE INTR	47	115	Specifies the events that can cause an HP-IL interrupt.
LOCAL	42	130	Sets an individual HP-IL device or all HP-IL devices to Local mode.
LOCAL LOCKOUT	42	132	Sets all HP-IL devices to Local Lockout condition.
OFF INTR	47	134	Cancels HP-IL interrupt branching.
ON INTR	47	137	Defines how a program branches when an enabled HP-IL interrupt event occurs.
READDDC	65	160	Returns the number of the last HP-IL device-dependent command message received.
READINTR	47	163	Returns the value of the interrupt-cause byte.
REMOTE	42	166	Enables all HP-IL devices to change to Remote mode and can also set an individual HP-IL device to Remote mode.
REQUEST	63	171	Defines the HP-71 status byte that is sent when serially polled by an HP-IL controller.
SEND	50	181	Sends individual HP-IL messages on the loop.
SPOLL	44	188	Returns a value that represents one or more status bytes from an HP-IL device.
STANDBY	46	193	Sets the HP-IL timeout period and verify interval.
STATUS	48	196	Returns the HP-IL interface status.
TRIGGER	42	199	Triggers an event at an HP-IL device.

General I/O Operations—Passing Control

CONTROL OFF/ON	54	98	Sets the controller status of the HP-71.
PASS CONTROL	52	152	Gives control of the HP-IL system to another device.

Binary Functions

BINAND	69	82	Returns the value of the binary AND operation.
BINCMPL	69	84	Returns the value of the binary complement.
BINEOR	69	86	Returns the value of the binary exclusive-OR operation.
BINIOR	69	88	Returns the value of the binary inclusive-OR operation.
BIT	69	90	Returns the value of one bit of an integer.

How To Use This Manual (page 9)

Part I: Overview

- 1: Getting Started (page 12)**
- 2: Printer and Display Operations (page 22)**
- 3: Mass Storage Operations (page 28)**
- 4: General I/O Operations (page 34)**

Part II: Keyword Dictionary

Organization (page 72)

Keyword Entries (pages 77-201)

Appendixes

- A: Care, Warranty, and Service Information (page 204)**
- B: Operating Information (page 212)**
- C: HP-71 Character Set (page 220)**
- D: Mass Storage Compatibility (page 224)**
- E: Error Messages (page 232)**
- F: Glossary (page 238)**



**HEWLETT
PACKARD**

Portable Computer Division
1000 N.E. Circle Blvd., Corvallis, OR 97330, U.S.A.

European Headquarters
150, Route Du Nant-D'Avril
P.O. Box, CH-1217 Meyrin 2
Geneva-Switzerland

HP-United Kingdom
(Pinewood)
GB-Nine Mile Ride, Wokingham
Berkshire RG11 3LL