ManualsPlus.com

HP-IB Programming Guide

For the HP 8711 RF Network Analyzer

HP Part No. 08711-90109
Printed in USA July 1993

ManualsPlus.com

© Copyright 1992, 1993 Hewlett-Packard Company

ManualsPlus.com

9a

HP-IB Programming

This document is an introduction to programming the HP 8711 RF network analyzer over the Hewlett-Packard Interface Bus (HP-IB). Its purpose is to provide concise information about the operation of the instrument under HP-IB control. It provides some background information on the HP-IB and a tutorial introduction using programming examples to demonstrate the remote operation of the HP 8711. The examples are provided on two disks that are included with the *HP 8711 Operating Manual*. Both disks contain the same examples, only the disk format is different.

- *Example Programs Disk —DOS Format* has examples written in HP Instrument BASIC (IBASIC). These programs can be run on the instrument's internal controller (option 1C2) or on a PC-compatible running Instrument BASIC for Windows (HP model no. E2200A).
- *Example Programs Disk — LIF Format* also has examples written in IBASIC. These programs can be run on the instrument's internal controller or on an HP series 300 workstation running HP BASIC (version 3.0 or higher).

The reader should become familiar with the operation of the HP 8711 network analyzer before controlling it over HP-IB. This document is not intended to teach programming or to discuss HP-IB theory except at an introductory level. Related information can be found in the following references:

- Information on the HP 8711's HP-IB command mnemonics is available in the *HP-IB Command Reference*.
- Information on making measurements with the HP 8711 is available in the *HP 8711 Operating Manual* and the *HP 8711 User's Guide*.
- Information on HP Instrument BASIC is available in the *HP Instrument BASIC User's Handbook* and *Using HP Instrument BASIC with the HP 8711*.
- Information concerning HP Instrument BASIC for Windows is available in *Installing and Using HP Instrument BASIC for Windows*.
- Information on HP BASIC programming is available in the manual set for the BASIC revision being used. For example: *BASIC 5.0 Programming Techniques* and *BASIC 5.0 Language Reference*.
- Information on using the HP-IB is available in the *Tutorial Description of the Hewlett-Packard Interface Bus* (HP literature no. 5952-0156).

ManualsPlus.com

Table of Contents

ManualsPlus.com

HP-IB Overview

HP-IB — the Hewlett-Packard Interface Bus — is a high-performance bus that allows individual instruments and computers to be combined into integrated test systems. The bus and its associated interface operations are defined by the IEEE 488.1 standard. The IEEE 488.2 standard defines the interface capabilities of instruments and controllers in a measurement system, including some frequently used commands.

HP-IB cables provide the physical link between devices on the bus. There are eight data lines on each cable that are used to send data from one device to another. Devices that send data over these lines are called **Talkers**. **Listeners** are devices that receive data over the same lines. There are also five control lines on each cable that are used to manage traffic on the data lines and to control other interface operations. **Controllers** are devices that use these control lines to specify the talker and listener in a data exchange.

When an HP-IB system contains more that one device with controller capabilities, only one of the devices is allowed to control data exchanges at any given time. The device currently controlling data exchanges is called the **Active Controller**. Also, only one of the controller-capable devices can be designated as the **System Controller**, the one device that can take control of the bus even if it is not the active controller. The HP 8711 can act as a talker, listener, active controller or system controller at different times.

HP-IB addresses provide a way to identify devices on the bus. The active controller uses HP-IB addresses to specify which device talks and which device listens during a data exchange. This means that each device's address must be unique. A device's address is set on the device itself, using either a front-panel key sequence or a rear-panel switch.

To set the HP-IB address on the HP 8711 network analyzer use the `HP8711 Address` softkey located in the (SYSTEM OPTIONS) `HP-IB` menu. The factory default address for the HP 8711 is 16.

Bus Structure

Data Bus

The data bus consists of eight lines that are used to transfer data from one device to another. Programming commands and data sent on these lines is typically encoded in the ASCII format, although binary encoding is often used to speed up the transfer of large arrays. Both ASCII and binary data formats are available to the HP 8711. In addition, every byte transferred over HP-IB undergoes a **handshake** to ensure valid data.

ManualsPlus.com

Handshake Lines

A three-line handshake scheme coordinates the transfer of data between talkers and listeners. This technique forces data transfers to occur at the speed of the slowest device, and ensures data integrity in multiple listener transfers. With most computing controllers and instruments, the handshake is performed automatically, which makes it transparent to the programmer.

Control Lines

The data bus also has five control lines that the controller uses both to send bus commands and to address devices.

IFC Interface Clear. Only the system controller uses this line. When this line is true (low) all devices (addressed or not) unaddress and go to an idle state.

ATN Attention. The active controller uses this line to define whether the information on the data bus is a **command** or is **data**. When this line is true (low) the bus is in the command mode and the data lines carry bus commands. When this line is false (high) the bus is in the data mode and the data lines carry device-dependent instructions or data.

SRQ Service Request. This line is set true (low) when a device requests service: the active controller services the requesting device. The analyzer can be enabled to pull the SRQ line for a variety of reasons.

REN Remote Enable. Only the system controller uses this line. When this line is set true (low) the bus is in the remote mode and devices are addressed either to listen or talk. When the bus is in remote and a device is addressed, it receives instructions from HP-IB rather than from its front panel (pressing the `Return to Local` softkey returns the device to front panel operation). When this line is set false (high) the bus and all devices return to local operation.

EOI End or Identify. This line is used by a talker to indicate the last data byte in a multiple byte transmission, or by an active controller to initiate a parallel poll sequence. The analyzer recognizes the EOI line as a terminator and it pulls the EOI line with the last byte of a message output (data, markers, plots, prints, error messages). The analyzer does not respond to parallel poll.

ManualsPlus.com

Sending Commands

Commands are sent over the HP-IB via a controller's language system, such as IBASIC, QuickBASIC or C. The keywords used by a controller to send HP-IB commands vary among systems. When determining the correct keywords to use, keep in mind that there are two different kinds of HP-IB commands:

- Bus management commands, which control the HP-IB interface.
- Device commands, which control analyzer functions.

Language systems usually deal differently with these two kinds of HP-IB commands. For example, HP BASIC uses a unique keyword to send each bus management command, but always uses the keyword `OUTPUT` to send device commands.

The following example shows how to send a typical device command:

```
OUTPUT 716;"CALCULATE:MARKER:MAXIMUM"
```

This sends the command within the quotes (`CALCULATE:MARKER:MAXIMUM`) to the HP-IB device at address 716. If the device is an HP 8711, the command instructs the analyzer to set a marker to the maximum point on the data trace.

HP-IB Requirements

Number of Interconnected Devices:	15 maximum
Interconnection Path/ Maximum Cable Length:	20 meters maximum or 2 meters per device, whichever is less.
Message Transfer Scheme:	Byte serial/ bit parallel asynchronous data transfer using a 3-line handshake system.
Data Rate:	Maximum of 1 megabyte per second over limited distances with tri-state drivers. Actual data rate depends on the transfer rate of the slowest device involved.
Address Capability:	Primary addresses: 31 talk, 31 listen. A maximum of 1 talker and 14 listeners at one time.
Multiple Controller Capability:	In systems with more than one controller (like the analyzer system), only one can be active at a time. The active controller can pass control to another controller, but only the system controller can assume unconditional control. Only one system controller is allowed. The system controller is hard-wired to assume bus control after a power failure.

ManualsPlus.com

Interface Capabilities

The HP 8711 has the following interface capabilities, as defined by the IEEE 488.1 standard:

SH1	full Source handshake capability
AH1	full Acceptor handshake capability
T6	basic Talker, Serial Poll, no Talk Only, unaddress if MLA
TE0	no Extended Talker capability
L4	basic Listener, no Listen Only, unaddress if MTA
LE0	no Extended Listener capability
SR1	full Service Request capability
RL1	full Remote/Local capability
DC1	full Device Clear capability
C1	System Controller capability
C2	send IFC and take charge Controller capability
C3	sen REN Controller capability
C4*	respond to SRQ
C6*	send IFC, receive control, pass control, parallel poll, pass control to self
C10*	send IFC, receive control, pass control, parallel poll
C12**	send IF messages, receive control, pass control
E2	tri-state drivers

* only when an HP Instrument BASIC program is running

** only when an HP Instrument BASIC program is not running

ManualsPlus.com

Programming Fundamentals

This section includes specific information that is needed to program an HP 8711 network analyzer. It includes how the analyzer interacts with a controller, how data is transferred between the analyzer and a controller, and how to use the analyzer's status register structure to generate service requests.

Controller Capabilities

The HP 8711 can be configured as an HP-IB system controller or as a talker/listener on the bus. To configure the analyzer, select either the `System Controller` or the `Talker/Listener` softkey in the SYSTEM OPTIONS `HP-IB` menu.

The HP 8711 is not usually configured as the system controller unless it is the only controller on the bus. This setup would be used if the analyzer only needed to control printers or plotters. It would also be used if HP Instrument BASIC was being used to control other test equipment.

When the analyzer is used with another controller on the bus, it is usually configured as a talker/listener. In this configuration, when the analyzer is passed control it can function as the active controller.

Response to Bus Management Commands

The HP-IB contains an attention (ATN) line that determines whether the interface is in command mode or data mode. When the interface is in command mode (ATN TRUE) a controller can send bus management commands over the bus. Bus management commands specify which devices on the interface can talk (send data) and which can listen (receive data). They also instruct devices on the bus, either individually or collectively, to perform a particular interface operation.

This section describes how the HP 8711 responds to the HP-IB bus management commands. The commands themselves are defined by the IEEE 488.1 standard. Refer to the documentation for your controller's language system to determine how to send these commands.

Device Clear (DCL)

When the analyzer receives this command, it:

- Clears its input and output queues.
- Resets its command parser (so it is ready to receive a new program message).
- Cancels any pending `*OPC` command or query.

ManualsPlus.com

The command does not affect:

- Front panel operation.
- Any analyzer operations in progress (other than those already mentioned).
- Any instrument settings or registers (although clearing the output queue may indirectly affect the Status Byte's Message Available (MAV) bit).

Go To Local (GTL)

This command returns the analyzer to local (front-panel) control. All keys on the analyzer's front-panel are enabled.

Interface Clear (IFC)

This command causes the analyzer to halt all bus activity. It discontinues any input or output, although the input and output queues are not cleared. If the analyzer is designated as the active controller when this command is received, it relinquishes control of the bus to the system controller. If the analyzer is enabled to respond to a Serial Poll it becomes Serial Poll disabled.

Local Lockout (LLO)

This command causes the analyzer to enter the local lockout mode, regardless of whether it is in the local or remote mode. The analyzer only leaves the local lockout mode when the HP-IB's Remote Enable (REN) line is set FALSE.

Local Lockout ensures that the analyzer's remote softkey menu (including the `Return to LOCAL` softkey) is disabled when the analyzer is in the remote mode. When the key is enabled, it allows a front-panel operator to return the analyzer to local mode, enabling all other front-panel keys. When the key is disabled, it does not allow the front-panel operator to return the analyzer to local mode.

Parallel Poll

The HP 8711 ignores all of the following parallel poll commands:

- Parallel Poll Configure (PPC).
- Parallel Poll Unconfigure (PPU).
- Parallel Poll Enable (PPE).
- Parallel Poll Disable (PPD).

Remote Enable (REN)

REN is a single line on the HP-IB. When it is set TRUE, the analyzer will enter the remote mode when addressed to listen. It will remain in remote mode until it receives the Go to Local (GTL) command or until the REN line is set FALSE.

ManualsPlus.com

When the analyzer is in remote mode and local lockout mode, all front-panel keys are disabled. When the analyzer is in remote mode but not in local lockout mode, all front-panel keys are disabled except for the softkeys. The remote softkey menu includes seven keys that are available for use by a program. The eighth softkey is the `Return to LOCAL` key which allows a front-panel operator to return the analyzer to local mode, enabling all other front-panel keys.

Selected Device Clear (SDC)

The analyzer responds to this command in the same way that it responds to the Device Clear (DCL) command.

When the analyzer receives this command it:

- Clears its input and output queues.
- Resets its command parser (so it is ready to receive a new program message).
- Cancels any pending `*OPC` command or query.

The command does not affect:

- Front-panel operation.
- Any analyzer operations in progress (other than those already mentioned).
- Any analyzer settings or registers (although clearing the output queue may indirectly affect the Status Byte's MAV bit).

Serial Poll

The analyzer responds to both of the serial poll commands. The Serial Poll Enable (SPE) command causes the analyzer to enter the serial poll mode. While the analyzer is in this mode, it sends the contents of its Status Byte register to the controller when addressed to talk.

When the Status Byte is returned in response to a serial poll, bit 6 acts as the Request Service (RQS) bit. If the bit is set, it will be cleared after the Status Byte is returned.

The Serial Poll Disable (SPD) command causes the analyzer to leave the serial poll mode.

Take Control Talker (TCT)

If the analyzer is addressed to talk, this command causes it to take control of the HP-IB. It becomes the active controller on the bus. The analyzer automatically passes control back when it completes the operation that required it to take control. Control is passed back to the address specified by the `*PCB` command (which should be sent prior to passing control).

If the analyzer does not require control when this command is received, it immediately passes control back.

ManualsPlus.com

Message Exchange

The HP 8711 communicates with the controller and other devices on the HP-IB using program messages and response messages. Program messages are used to send commands, queries, and data to the analyzer.

Response messages are used to return data from the analyzer. The syntax for both kinds of messages is discussed in the *HP-IB Command Reference*.

There are two important things to remember about the message exchanges between the analyzer and other devices on the bus:

- The analyzer only talks after it receives a terminated query (see "Query Response Generation" later in this section).
- Once it receives a terminated query, the analyzer expects to talk before it is told to do something else.

HP-IB Queues

Queues enhance the exchange of messages between the HP 8711 and other devices on the bus. The analyzer contains:

- An input queue.
- An error queue.
- An output queue.

Input Queue. The input queue temporarily stores the following until they are read by the analyzer's command parser:

- Device commands and queries.
- The HP-IB END message (EOI asserted while the last data byte is on the bus).

The input queue also makes it possible for a controller to send multiple program messages to the analyzer without regard to the amount of time required to parse and execute those messages. The queue holds up to 128 bytes. It is cleared when:

- The analyzer is turned on.
- The Device Clear (DCL) or Selected Device Clear (SDC) command is received.

Error Queue. The error queue temporarily stores up to 20 error messages. Each time the analyzer detects an error, it places a message in the queue. When you send the `SYST:ERR?` query, one message is moved from the error queue to the output queue so it can be read by the controller. Error messages are delivered to the output queue in the order they were received.

The error queue is cleared when:

- All the error messages are read using the `SYST:ERR?` query.
- The analyzer is turned on.
- The `*CLS` command is received.

ManualsPlus.com

Output Queue. The output queue temporarily stores a single response message until it is read by a controller. It is cleared when:

- The message is read by a controller.
- The analyzer is turned on.
- The Device Clear (DCL) or Selected Device Clear (SDC) command is received.

Command Parser

The command parser reads program messages from the input queue in the order they were received from the bus. It analyzes the messages to determine what actions the analyzer should take.

One of the parser's most important functions is to determine the position of a program message in the analyzer's command tree (described in the *HP-IB Command Reference*). When the command parser is reset, the next command it receives is expected to arise from the base of the analyzer's command tree.

The parser is reset when:

- The analyzer is turned on.
- The Device Clear (DCL) or Selected Device Clear (SDC) command is received.
- A colon immediately follows a semicolon in a program message. (For more information see "Sending Multiple Commands" in the *HP-IB Command Reference*.)
- A program message terminator is received. A program message terminator can be an ASCII carriage return ($^{C}_{R}$) or newline character or the HP-IB END message (EOI set true).

Query Response Generation

When the HP 8711 parses a query, the response to that query is placed in the analyzer's output queue. The response should be read immediately after the query is sent. This ensures that the response is not cleared before it is read. The response is cleared when one of the following message exchange conditions occurs:

- Unterminated condition — the query is not properly terminated with an ASCII carriage return character or the HP-IB END message (EOI set true) before the response is read.
- Interrupted condition — a second program message is sent before the response to the first is read.
- Buffer deadlock — a program message is sent that exceeds the length of the input queue or that generates more response data than fits in the output queue.

ManualsPlus.com

Synchronization

The IEEE 488.2 standard provides tools that can be used to synchronize the analyzer and a controller. Proper use of these tools ensures that the analyzer is in a known state when you send a particular command or query.

Device commands can be divided into two broad classes:

- Sequential commands.
- Overlapped commands.

Most of the HP 8711's commands are processed sequentially. A sequential command holds off the processing of subsequent commands until it has been completely processed.

Some commands do not hold off the processing of subsequent commands; they are called overlapped commands.

Overlapped commands

Typically, overlapped commands take longer to process than sequential commands. For example, the `:INITIATE:IMMEDIATE` command restarts a measurement. The command is not considered to have been completely processed until the measurement is complete. This can take a long time with a narrow system bandwidth or when averaging is enabled.

The HP 8711 has the following overlapped commands:

```
:ABORt
:CALibration:ZERO:AUTO
:CONFigure[1|2]
:DIAGnostic:CCONstants:LOAD
:DIAGnostic:CCONstants:STORe:DISK
:DIAGnostic:CCONstants:STORe:EEPRom
:DIAGnostic:DITHer
:DIAGnostic:SPUR:AVOid
:HCOPy[:IMMediate]
:INITiate[1|2]:CONTinuous
:INITiate[1|2][:IMMediate]
:MMEMory:LOAD:STATe
:OUTPut[:STATe]
:PROGram[:SELected]:EXECute
:SENSe[1|2]:AVERage:CLEar
:SENSe[1|2]:AVERage:COUNt
:SENSe[1|2]:AVERage[:STATe]
:SENSe[1|2]:BWIDth[:RESolution]
:SENSe[1|2]:CORRection:COLLect[:ACQuire]
:SENSe[1|2]:CORRection:COLLect:ISTate[:AUTO]
:SENSe[1|2]:CORRection:COLLect:METHod
:SENSe[1|2]:CORRection:COLLect:SAVE
:SENSe[1|2]:CORRection:CSET[:SELect]
```

ManualsPlus.com

```
:SENSe[1|2]:CORRection[:STATe]
:SENSe:COUPle
:SENSe[1|2]:DETector[:FUNCtion]
:SENSe[1|2]:FREQuency:CENTer
:SENSe[1|2]:FREQuency:SPAN
:SENSe[1|2]:FREQuency:STARt
:SENSe[1|2]:FREQuency:STOP
:SENSe[1|2]:FUNCtion
:SENSe:ROSCillator:SOURce
:SENSe[1|2]:STATe
:SENSe[1|2]:SWEep:POINts
:SENSe[1|2]:SWEep:TIME
:SENSe[1|2]:SWEep:TIME:AUTO
:SENSe:SWEep:TRIGger:SOURce
:SOURce[1|2]:POWer[:LEVel][:IMMediate][:AMPLitude]
:SYSTem:PRESet
:TRACe[:DATA]
:TRIGger[:SEQuence]:SOURce
```

The analyzer uses a No Pending Operation (NPO) flag to keep track of overlapped commands. The NPO flag is reset to 0 when an overlapped command has not completed (still pending). It is set to 1 when no overlapped commands are pending. The NPO flag cannot be read directly but all of the following common commands take some action based on the setting of the flag.

- `*WAI` — Holds off the processing of subsequent commands until the NPO flag is set to 1. This ensures that commands in the analyzer's input queue are processed in the order received.

  The program continues to run and additional commands are received and parsed by the analyzer (but not executed) while waiting for the NPO flag to be set. Use of the `*WAI` command is demonstrated in the `SETUP` example program.

- `*OPC?` — Places a 1 in the analyzer's output queue when the NPO flag is set to 1. If the program is designed to read the output queue before it continues, this effectively pauses the controller until all pending overlapped commands are completed. Use of the `*OPC?` command is demonstrated in the `TRANCAL` and `REFLCAL` example programs.

- `*OPC` — Sets bit 0 of the Standard Event Status event register to 1 when the NPO flag is set to 1. The analyzer's status registers can then be used to generate a service request when all pending overlapped commands are completed. This synchronizes the controller to the completion of an overlapped command, but also leaves the controller free to perform other tasks while the command is executing.

Note

`*OPC` only informs you when the NPO flag is set to 1. It does not hold off the processing of subsequent commands. No commands should be sent to the analyzer between sending the `*OPC` command and receiving the service request. Any command sent will be executed and may affect how the instrument responds to the previously sent `*OPC`.

ManualsPlus.com

The *CLS and *RST commands cancel any preceding *OPC command or query. Pending overlapped commands are still completed, but their completion is not reported in either the status register or the output queue. Two HP-IB bus management commands — Device Clear (DCL) and Selected Device Clear (SDC) — also cancel any preceding *OPC command or query.

Note

Use *WAI, *OPC? or *OPC whenever overlapped commands are used. A recommended technique is to send *WAI at the end of each group of commands.

Caution

ALWAYS trigger an individual sweep (using *OPC? and waiting for the reply) before reading data over the bus or executing a marker function. The analyzer has the ability to process the commands it receives faster than it can make a measurement. If the measurement is not complete when the data is read or a marker search function is executed the results are invalid.

The command to use (in an IBASIC OUTPUT statement) is:

```
OUTPUT @Hp8711;"ABOR;:INIT:CONT OFF;:INIT;*OPC?
ENTER @Hp8711;Opc_done
```

or another form of the :INITiate[1|2][:IMMediate] command combined with the *OPC? query.

ManualsPlus.com

Passing Control

The HP 8711 requires temporary control of the HP-IB to complete some commands (such as print/plot or external disk operations). After sending such a command, the active controller must pass control to the analyzer. When the analyzer completes the command, it automatically passes control of the bus back to the controller.

For smooth passing of control, take steps that ensure the following conditions are met:

- The analyzer must know the controller's address so it can pass control back.
- The controller must be informed when the analyzer passes control back.

The following is a procedure for passing control:

1. Send the controller's HP-IB address to the analyzer with the `*PCB` command.
2. Clear the analyzer's status registers with the `*CLS` command.
3. Enable the analyzer's status registers to generate a service request when the Operation Complete bit is set. (Send `*ESE` with a value of 1 and `*SRE` with a value of 32.)
4. Enable the controller to respond to the service request.
5. Send the command that requires control of the bus followed by the `*OPC` command.
6. Pass control to the analyzer and wait for the service request. The service request indicates that the command has been completed and control has been passed back to the controller.

Note

For this procedure to work properly, only the command that requires control of the bus should be pending. Other overlapped commands should not. For more information on overlapped commands, see "Synchronization" in this chapter.

An example program, `PASSCTRL`, demonstrates passing control to the analyzer. In this example program control is passed so the analyzer can control a printer for hardcopy output.

ManualsPlus.com

Transferring Data

Data is transferred between the HP 8711 and a controller via the HP-IB data lines, DIO1 through DIO8. Such transfers occur in a byte-serial (one byte at a time), bit-parallel (8 bits at a time) manner. This section discusses the following aspects of data transfer:

- The different data types used during data transfers.
- Data encoding used during transfers of numeric block data.

Data Types

The HP 8711 uses a number of different data types during data transfers. Data transfer occurs in response to a query. The data type used is determined by the parameter being queried. The different parameter types are described in the "Parameter Types" section of the *HP-IB Command Reference*. Data types described in this section are:

- Numeric Data.
- Character Data
- String Data
- Expression Data
- Block Data

Numeric Data

The analyzer returns three types of numeric data in response to queries:

- NR1 data — Integers (such as `+1`, `0`, `-1`, `123`, `-12345`). This is the response type for boolean parameters as well as some numeric parameters.
- NR2 data — Floating point numbers with an explicit decimal point (such as `12.3`, `+1.234`, `-0.12345`).
- NR3 data — Floating point numbers in scientific notation (such as `+1.23E+5`, `+123.4E-3`, `-456.789E+6`).

Character Data

Character data consists of ASCII characters grouped together in mnemonics that represent specific instrument settings (such as `MAXimum` , `MINimum` or `MLOGarithmic`). The analyzer always returns the short form of the mnemonic in upper-case alpha characters.

String Data

String data consists of ASCII characters. The string must be enclosed by a delimiter, either single quotes (`'This is string data.'`) or double quotes (`"This is also string data."`). To include the delimiter as a character in the string it must be typed twice without any characters in between. The analyzer always uses double quotes when it returns string data.

ManualsPlus.com

Expression Data

Expression data consists of mathematical expressions that use character parameters. When expression data is sent to the analyzer it is always enclosed in parentheses (such as `(IMPL/CH1SMEM)` or `(IMPL)`). The analyzer returns expression data enclosed in double quotes.

Block Data

Block data are typically used to transfer large quantities of related data (like a data trace). Blocks can be sent as definite length blocks or indefinite length blocks — the instrument will accept either form. The analyzer always returns definite length block data in response to queries.

Definite Block Length. The general form for a definite block length transfer is:

```
#<num_digits><num_bytes><data_bytes>
```

In the definite length block, two numbers must be specified. The single decimal digit `<num_digits>` specifies how many digits are contained in `<num_bytes>`. The decimal number `<num_bytes>` specifies how many data bytes will follow in `<data_bytes>`. An example IBASIC (or HP BASIC) statement to send `ABC+XYZ` as a definite block length parameter is shown, note that the data block contains seven bytes (7) and only one digit is needed to describe the block length `1`.

```
OUTPUT 716;"#17ABC+XYZ"
```

Indefinite Block Length. The general form for an indefinite block length transfer is:

```
#0<data_bytes><^C_R><EOI>
```

Note that a mandatory $^{C}_{R}$ (carriage return or newline) `EOI` sequence immediately follows the last byte of block data in an indefinite length block. This forces the termination of the program message. An example IBASIC (or HP BASIC) statement to send `ABC+XYZ` as an indefinite block length parameter is shown, note that `,END` is used to properly terminate the message.

```
OUTPUT 716;"#0ABC+XYZ",END
```

Data Encoding for Large Data Transfers

The `FORMat:DATA` command selects the type of data and the type of data encoding that is used to transfer large blocks of numeric data between the analyzer and a controller. There are two specifiers:

- `REAL` specifies the block data type. Either the definite or indefinite length syntax can be used. The block is transferred as a series of binary-encoded floating-point numbers. Data transfers of the `REAL,64` data type are demonstrated in the `REALDATA` example program.
- `INTeger` specifies the block data type. Either the definite or indefinite length syntax can be used. The block is transferred as an array of binary-encoded data with each point represented by a set of three 16-bit integers. This is the instrument's internal format — it

ManualsPlus.com

should only be used for data that will be returned to the instrument for later use. Data transfers of the `INTeger,16` data type are demonstrated in the `INTDATA` and `LOADCALS` example programs.

- `ASCii` specifies the numeric data type (NR1, NR2 or NR3 syntax). The data is transferred as a series of ASCII-encoded numbers separated by commas. `ASCii` formatted data transfers are demonstrated in the `ASCDATA` example program.

Blocks that contain mixed data — both numbers and ASCII characters — ignore the setting of `FORMat:DATA`. These blocks always transfer as either definite length or indefinite length block data. The following commands transfer blocks of mixed data:

```
PROGram[:SELected]:DEFine
SYSTem:SET
```

ASCII Encoding

The ANSI X3.4-1977 standard defines the ASCII 7-bit code. When an ASCII-encoded byte is sent over the HP-IB, bits 0 through 6 of the byte (bit 0 being the least significant bit) correspond to the HP-IB data lines DIO1 through DIO7. DIO8 is ignored.

When ASCII encoding is used for large blocks of data, the number of significant digits to be returned for each number in the block can be specified. For example, the following command returns all numbers as NR3 data with 7 significant digits.

```
FORMat:DATA ASCii,7
```

Binary Encoding

When binary encoding is used for large blocks of data, all numbers in the block are transferred as 32-bit or 64-bit binary floating point numbers or as an array of 16-bit integers. The binary floating-point formats are defined in the IEEE 754-1985 standard.

`FORMat:DATA REAL,32` selects the IEEE 32-bit format (not supported by IBASIC or HP BASIC).

`FORMat:DATA REAL,64` selects the IEEE 64-bit format.

`FORMat:DATA INTeger,16` selects the 16-bit integer format.

Byte Swapping

PC compatibles frequently use a modification of the IEEE floating point formats with the byte order reversed. To reverse the byte order for data transfer into a PC, the `FORMat:BORDer` command should be used.

`FORMat:BORDer SWAPped` selects the byte-swapped format.

`FORMat:BORDer:NORMal` selects the standard format.

ManualsPlus.com

Using the Status Registers

The HP 8711's status registers contain information about the condition of the network analyzer and its measurements. This section describes the registers and their use in HP-IB programming.

Example programs using the status registers are included in the "Example Programs" section later in this chapter. These programs include `AVERAGE` and `GRAPHICS` which use of service request interrupt routines, `PASSCTRL` which uses the status byte to request control of the HP-IB and `LIMITEST` which uses the Limit Fail condition register.

General Status Register Model

The HP 8711's status system is based on the general status register model shown in Figure 9a-1. Most of the analyzer's register sets include all of the registers shown in the model, although commands are not always available for reading or writing a particular register. The information flow within a register set starts at the condition register and ends at the register summary bit (see Figure 9a-2). This flow is controlled by setting bits in the transition and enable registers.

Two register sets — the Status Byte and the Standard Event Status Register — are 8-bits wide. All others are 16-bits wide, but the most significant bit (bit 15) in the larger registers is always set to 0.

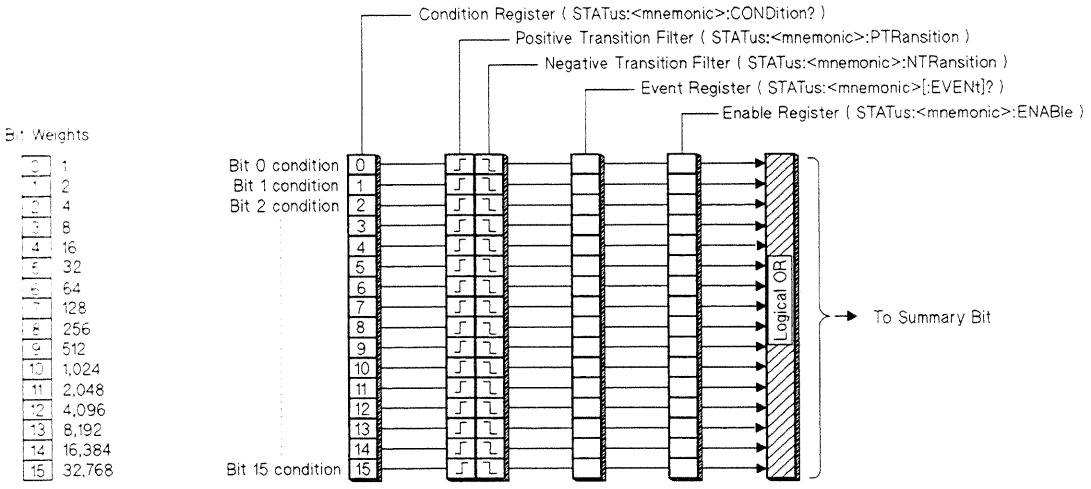


Figure 9a-1. General Status Register Model

ManualsPlus.com

Condition Register

Condition registers continuously monitor the instrument's hardware and firmware status. Bits in a condition register are not latched or buffered, they are updated in real time. When the condition monitored by a specific bit becomes true, the bit is set to 1. When the condition becomes false the bit is reset to 0. Condition registers are read-only.

Transition Registers

Transition registers control what type of change in a condition register will set the corresponding bit in the event register. Positive state transitions (0 to 1) are only reported to the event register if the corresponding positive transition bit is set to 1. Negative state transitions (1 to 0) are only reported if the corresponding negative transition bit is set to 1. Setting both transition bits to 1 causes both positive and negative changes to be reported. Transition registers are read-write, and are unaffected by `*CLS` (clear status) or queries. They are reset to instrument default conditions at power up and after `*RST` and `SYSTem:PRESet` commands.

Event Register

Event registers latch any reported condition changes. When a transition bit allows a condition change to be reported, the corresponding event bit is set to 1. Once set, an event bit is no longer affected by condition changes. It remains set until the event register is cleared. Event registers are read-only.

An event register is cleared when you read it. All event registers are cleared when you send the `*CLS` (clear status) command.

Enable Register

Enable registers control the reporting of events (latched conditions) to the register summary bit. If an enable bit is set to 1 the corresponding event is included in the logical ORing process that determines the state of the summary bit. (The summary bit is only set to 1 if one or more enabled event bits are set to 1.) Summary bits are recorded in the instrument's status byte. Enable registers are read-write and are cleared by `*CLS` (clear status).

ManualsPlus.com

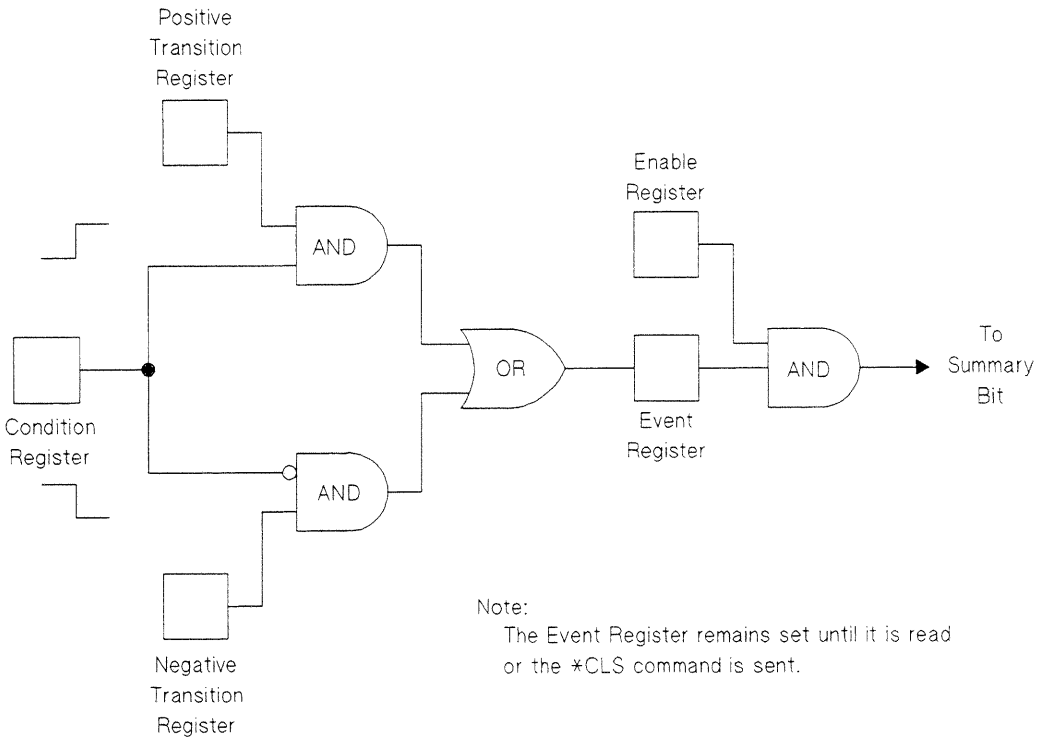


Figure 9a-2. Flow of information within a register set

How to Use Registers

There are two methods of accessing the information in status registers:

- The direct-read method.
- The service request (SRQ) method.

In the direct-read method the analyzer is passive. It only tells the controller that conditions have changed when the controller asks the right question. In the SRQ method, the analyzer is more active. It tells the controller when there has been a condition change without the controller asking. Either method allows you to monitor one or more conditions.

The following steps are used to monitor a condition with the direct read method:

1. Determine which register contains the bit that monitors the condition.
2. Send the unique HP-IB query that reads that register.
3. Examine the bit to see if the condition has changed.

The direct-read method works well when it is not necessary to know about changes the moment they occur. It does not work well if immediate knowledge of the condition change is needed. A program that used this method to detect a change in a condition would need to continuously read the registers at very short intervals. The SRQ method is better suited for that type of need.

ManualsPlus.com

The Service Request Process

The following steps are used to monitor a condition with the SRQ method:

1. Determine which bit monitors the condition.
2. Determine how that bit reports to the request service (RQS) bit of the Status Byte.
3. Send HP-IB commands to enable the bit that monitors the condition and to enable the summary bits that report the condition to the RQS bit.
4. Enable the controller to respond to service requests.

When the condition changes, the analyzer sets its RQS bit and the HP-IB's SRQ line. The controller is informed of the change as soon as it occurs. The time the controller would otherwise have used to monitor the condition can now be used to perform other tasks. The controller's response to the SRQ is determined by the program being run.

Generating a Service Request

A service request is generated using the Status Byte. As shown in Figure 9a-3, the HP 8711's other register sets report to the Status Byte. Some of them report directly while others report indirectly through other register sets.

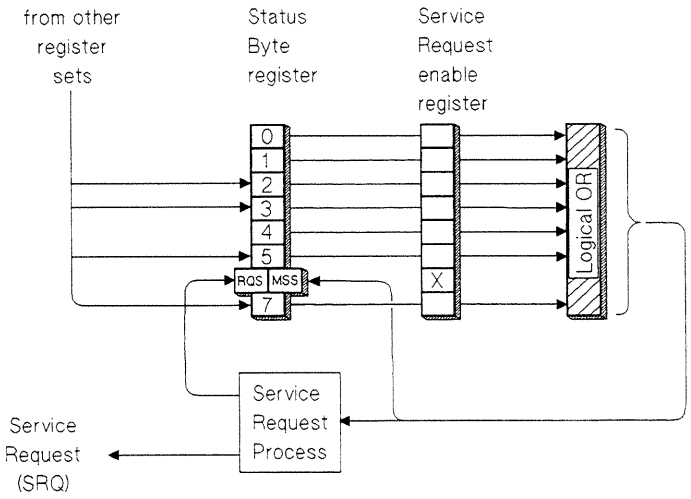


Figure 9a-3. Generating a Service Request

The process of preparing the analyzer to generate a service request, and the handling of that interrupt when it is received by a program, are demonstrated in the `AVERAGE` example program.

ManualsPlus.com

When a register set causes its summary bit in the Status Byte to change from 0 to 1, the analyzer can initiate the service request (SRQ) process. If both the following conditions are true the process is initiated:

- The corresponding bit of the Service Request enable register is also set to 1.
- The analyzer does not have a service request pending. (A service request is considered to be pending between the time the analyzer's SRQ process is initiated and the time the controller reads the Status Byte register with a serial poll).

The SRQ process sets the HP-IB's SRQ line true and sets the Status Byte's request service (RQS) bit to 1. Both actions are necessary to inform the controller that the HP 8711 requires service. Setting the SRQ line informs the controller that some device on the bus requires service. Setting the RQS bit allows the controller to determine that the HP 8711 was the device that initiated the request.

When a program enables a controller to detect and respond to service requests, it should instruct the controller to perform a serial poll when the HP-IB's SRQ line is set true. Each device on the bus returns the contents of its Status Byte register in response to this poll. The device whose RQS bit is set to 1 is the device that requested service.

Note

When the analyzer's Status Byte is read with a serial poll, the RQS bit is reset to 0. Other bits in the register are not affected.

As implied in Figure 9a-3, bit 6 of the Status Byte register serves two functions; the request service function (RQS) and the master summary status function (MSS). Two different methods for reading the register allow you to access the two functions. Reading the register with a serial poll allows you to access the bit's RQS function. Reading the register with `*STB` allows you to access the bit's MSS function.

ManualsPlus.com

The HP 8711's Status Register Sets

The HP 8711 uses eight register sets to keep track of instrument status:

- Status Byte — `*STB?` and `*SRE`
- Device Status — `STATus:DEVice`
- Limit Fail — `STATus:QUEStionable:LIMit`
- Questionable Status — `STATus:QUEStionable`
- Standard Event Status — `*ESR?` and `*ESE`
- Measuring Status — `STATus:OPERation:MEASuring`
- Averaging Status — `STATus:OPERation:AVERaging`
- Operational Status — `STATus:OPERation`

Their reporting structure is summarized in Figure 9a-4. They are described in greater detail in the following section.

Note

Register bits not explicitly presented in the following sections are not used by the HP 8711. A query to one of these bits returns a value of 0.

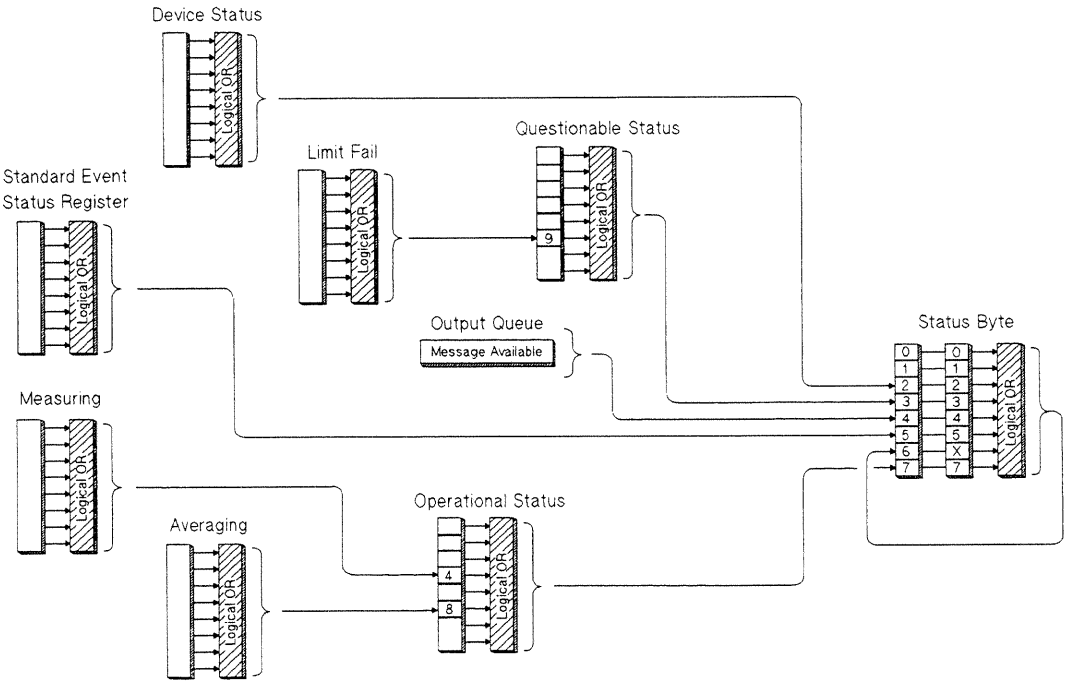


Figure 9a-4. HP 8711 Register Sets

ManualsPlus.com

Status Byte

The Status Byte register set summarizes the states of the other register sets and monitors the analyzer's output queue. It is also responsible for generating service requests (see "Generating Service Requests" earlier in this chapter). See Figure 9a-5

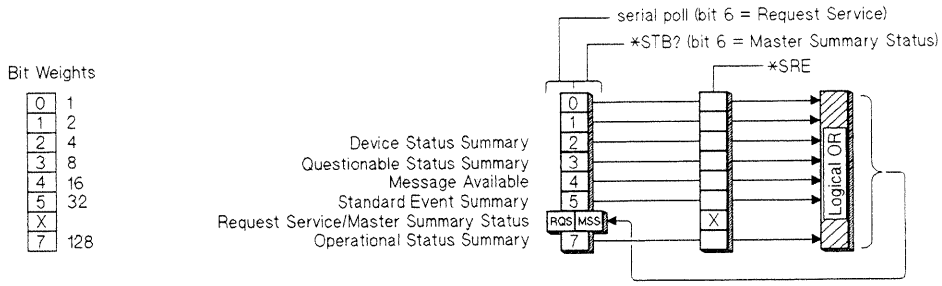


Figure 9a-5. The Status Byte Register Set

The Status Byte register set does not conform to the general status register model described at the beginning of this chapter. It contains only two registers: the Status Byte register and the Service Request enable register. The Status Byte register behaves like a condition register for all bits except bit 6. The Service Request enable register behaves like a standard enable register except that bit 6 is always set to 0.

Bits in the Status Byte register are set to 1 under the following conditions:

- **Device Status Summary** (bit 2) is set to 1 when one or more enabled bits in the Device Status event register are set to 1.
- **Questionable Status Summary** (bit 3) is set to 1 when one or more enabled bits in the Questionable Status event register are set to 1.
- **Message Available** (bit 4) is set to 1 when the output queue contains a response message.
- **Standard Event Status Summary** (bit 5) is set to 1 when one or more enabled bits in the Standard Event Status event register are set to 1.
- **Master Summary Status** (bit 6, when read by `*STB`) is set to 1 when one or more enabled bits in the Status Byte register are set to 1.
- **Request Service** (bit 6, when read by serial poll) is set to 1 by the service request process (see "Generating a Service Request" earlier in this chapter).
- **Operational Status Summary** (bit 7) is set to 1 when one or more enabled bits in the Operational Status event register are set to 1.

ManualsPlus.com

The commands used to read and write the Status Byte registers are listed below:

- `SPOLL` — an IBASIC (or HP BASIC) command used in the service request process to determine which device on the bus is requesting service.
- `*STB?` reads the value of the instrument's status byte. This is a non-destructive read, the Status Byte is cleared by the `*CLS` command.
- `*SRE <num>` sets bits in the Service Request Enable register. The current setting of the Service Request Enable register is stored in non-volatile memory.
- `*SRE?` reads the current state of the Service Request Enable register.

Device Status Register Set

The Device Status register set monitors the state of device-specific parameters.

Bits in the Device Status condition register are set to 1 under the following conditions:

- **Key Pressed** (bit 0) is set to 1 when one of the HP 8711's front panel keys has been pressed.

Limit Fail Register Set

The Limit Fail register set monitors limit test results for both measurement channels.

Bits in the Limit Fail condition register are set to 1 under the following conditions:

- **Channel 1 Limit Failed** (bit 0) is set to 1 when limit testing in enabled and any point on channel 1 fails the limit test.
- **Channel 2 Limit Failed** (bit 1) is set to 1 when limit testing in enabled and any point on channel 2 fails the limit test.

Questionable Status Register Set

The Questionable Status register set monitors conditions that affect the quality of measurement data.

Bits in the Questionable Status condition register are set to 1 under the following conditions:

- **Limit Fail** (bit 9) is set to 1 when one or more enabled bits in the Limit Fail event register are set to 1.
- **Data Questionable** (bit 10) is set to 1 when a change in the analyzer's configuration requires that new measurement data be taken.

ManualsPlus.com

Standard Event Status Register Set

The Standard Event Status register set monitors HP-IB errors and synchronization conditions. See Figure 9a-6

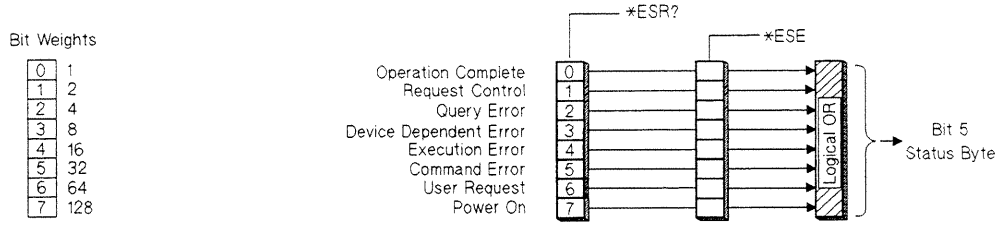


Figure 9a-6. The Standard Event Status Register Set

The Standard Event Status register set does not conform to the general status register model described at the beginning of this section. It contains only two registers: the Standard Event Status event register and the Standard Event Status enable register. The Standard Event Status event register is similar to other event registers, but behaves like a register set that has a positive transition register with all bits set to 1. The Standard Event Status enable register is the same as other enable registers.

- **Operation Complete** (bit 0) is set to one when the following two events occur (in the order listed):
  - The `*OPC` command is sent to the analyzer.
  - The analyzer completes all pending overlapped commands.
- **Request Control** (bit 1) is set to 1 when both of the following conditions are true:
  - The analyzer is configured as a talker/listener for HP-IB operation.
  - The analyzer is instructed to do something (such as plotting or printing) that requires it to take control of the bus.
- **Query Error** (bit 2) is set when the command parser detects a query error.
- **Device Dependent Error** (bit 3) is set to 1 when the command parser detects a device-dependent error.
- **Execution Error** (bit 4) is set to 1 when the command parser detects an execution error.
- **Command Error** (bit 5) is set to 1 when the command parser detects a command error.
- **Power On** (bit 7) is set to 1 when you turn on the analyzer.

ManualsPlus.com

The commands used to read and write the Standard Event Status registers are listed below:

- `*ESR?` reads the value of the standard event status register.
- `*ESE <num>` sets bits in the standard event status enable register. The current setting of the standard event statue enable register is stored in non-volatile memory.
- `*ESE?` reads the current state of the standard event status enable register.

Measuring Status Register Set

The Measuring Status register set monitors conditions in the analyzer's measurement process.

Bits in the Measuring Status condition register are set to 1 under the following conditions:

- **Channel 1 Measuring** (bit 0) is set to 1 while the analyzer is collecting measurement data on channel 1.
- **Channel 2 Measuring** (bit 1) is set to 1 while the analyzer is collecting measurement data on channel 2.

Averaging Status Register Set

The Averaging Status register set monitors conditions in the analyzer's measurement process when the trace averaging function is in use.

Bits in the Averaging Status condition register are set to 1 under the following conditions:

- **Channel 1 Averaging** (bit 0) is set to 1 while the analyzer is sweeping on channel 1 and the number of sweeps completed (since "average restart") is less than the averaging factor.
- **Channel 2 Averaging** (bit 1) is set to 1 while the analyzer is sweeping on channel 2 and the number of sweeps completed (since "average restart") is less than the averaging factor.

Operational Status Register Set

The Operation Status register set monitors conditions in the analyzer's measurement process, disk operations, and printing/plotting operations. It also monitors the state of the current HP Instrument BASIC program.

Bits in the Operational Status condition register are set to 1 under the following conditions:

- **Calibrating** (bit 0) is set to 1 while the instrument is zeroing the broadband diode detectors.
- **Settling** (bit 1) is set to 1 while the measurement hardware is settling.
- **Measuring** (bit 4) is set to 1 when one or more enabled bits in the Measuring Status event register are set to 1.
- **Correcting** (bit 7) is set to 1 while the analyzer is performing a calibration function.

ManualsPlus.com

- **Averaging** (bit 8) is set to 1 when one or more enabled bits in the Averaging Status event register are set to 1.
- **Hardcopy Running** (bit 9) is set to 1 while the analyzer is performing a hardcopy (print or plot) function.
- **Test Running** (bit 10) is set to 1 when one of the analyzer's internal service tests is being run.
- **Program Running** (bit 14) is set to 1 while an HP Instrument BASIC program is running on the analyzer's internal controller.

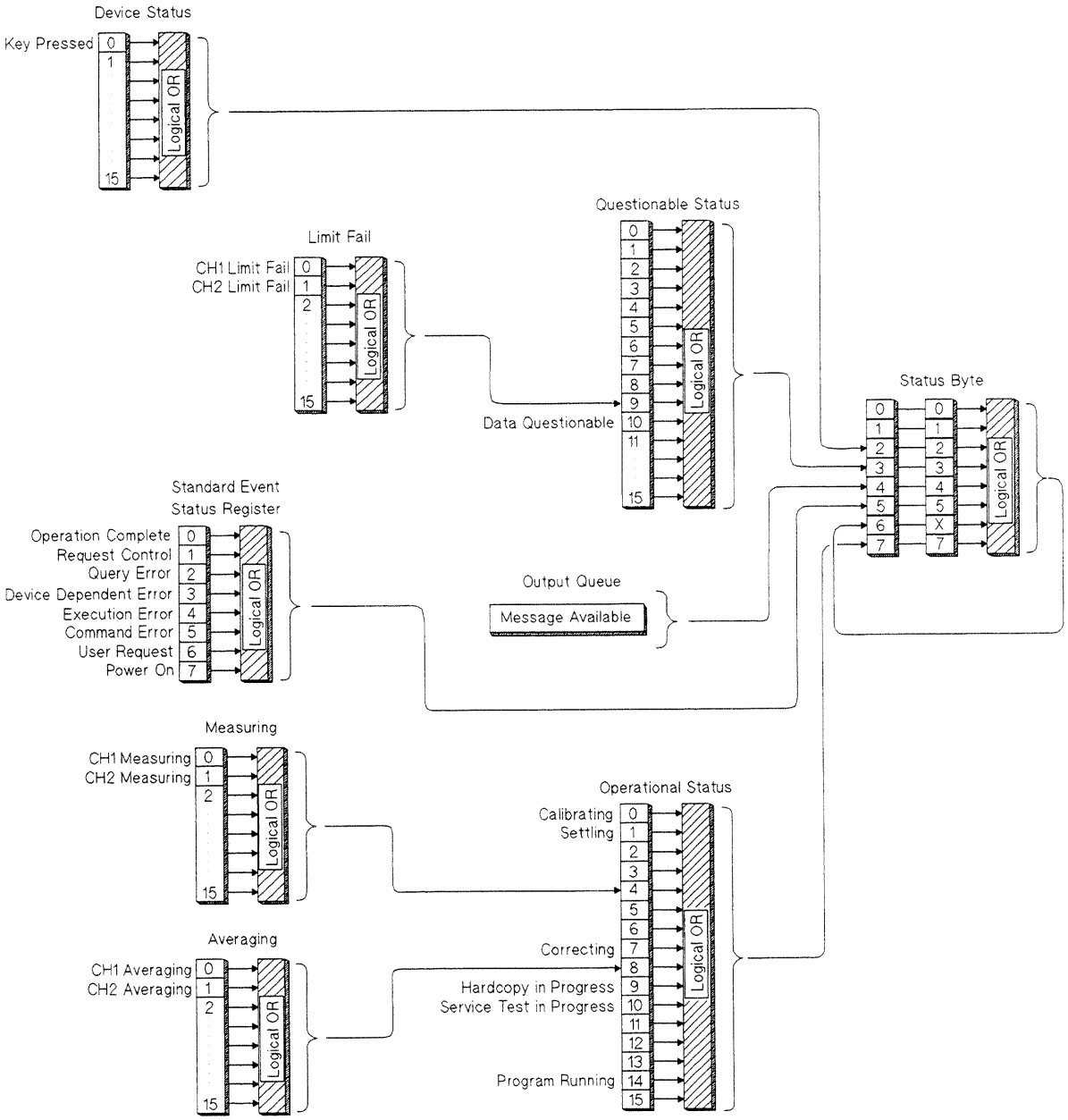
STATus:PRESet Settings

Executing the `STATus:PRESet` command changes the settings in the enable (`ENAB`), positive transition (`PTR`) and negative transition (`NTR`) registers. The table below shows the settings after the command is executed.

Register Set	ENABle	PTRansition	NTRansition
STATus:DEVice	all 0s	all 1s	all 0s
STATus:QUEStionable:LIMit	all 1s	all 1s	all 0s
STATus:QUEStionable	all 0s	all 1s	all 0s
STATus:OPERation:MEASuring	all 1s	all 0s	all 1s
STATus:OPERation:AVERaging	all 1s	all 0s	all 1s
STATus:OPERation	all 0s	all 1s	all 0s

ManualsPlus.com

HP 8711 Register Set Summary



ManualsPlus.com

User Graphics

The HP 8711 has a set of user graphics commands that can be used to create graphics and messages on the display. The GRAPHICS example program uses some of these commands to draw a simple setup diagram. These commands, listed below, are of the form:

```
DISPlay:WINDow[1|2|10]:GRAPhics:<mnemonic>.
```

The number specified in the WINDow part of the command selects where the graphics are to be written.

- WINDow1 draws the graphics to the channel 1 measurement screen. (This is the default if no window is specified in the mnemonic.)
- WINDow2 draws the graphics to the channel 2 measurement screen.
- WINDow10 draws the graphics to an IBASIC display partition. (This window is only available on instruments with IBASIC — option 1C2.)

Note

When graphics commands are used to write directly to a measurement screen they write to the static graphics plane (the same plane where the graticule is drawn). There is no sweep-to-sweep speed penalty once the graphics have been drawn.

Unless otherwise specified, the graphics commands listed below start at the current pen location. All sizes are dimensioned in pixels.

```
DISPlay:WINDow[1|2|10]:GRAPhics:CIRCle <diameter>
DISPlay:WINDow[1|2|10]:GRAPhics:CLEar
DISPlay:WINDow[1|2|10]:GRAPhics:COLor <pen>
  color choices are: 0 for erase, 1 for bright, 2 for dim
DISPlay:WINDow[1|2|10]:GRAPhics[:DRAW] <new_x>,<new_y>
DISPlay:WINDow[1|2|10]:GRAPhics:LABel <string>
DISPlay:WINDow[1|2|10]:GRAPhics:LABel:FONT <font>
  font choices are: SMALl, HSMall, NORMal, HNORmal, BOLD, HBOLd, SLANt, HSLant
  (H as the first letter of the font name indicates highlighted text – inverse video).
DISPlay:WINDow[1|2|10]:GRAPhics:MOVE <new_x>,<new_y>
DISPlay:WINDow[1|2|10]:GRAPhics:RECTangle <width>,<height>
DISPlay:WINDow[1|2|10]:GRAPhics:STATe?
```

ManualsPlus.com

Window Geometry

Even though there are only three graphics windows, these windows can have different sizes and locations.

The size and location of the graphics window are determined by the display configuration currently in use — split screen measurements, full screen measurements, and full or partial IBASIC display partitions will affect the dimensions of the graphics window in use.

The sizes of the different graphics windows are listed below.

- Channel 1 or channel 2 full screen measurement: width=801 pixels, height=321 pixels.
- Channel 1 or channel 2 split screen measurement: width=801 pixels, height=161 pixels.
- IBASIC full screen display: width=861 pixels, height=352 pixels.
- IBASIC upper display: width=861 pixels, height=160 pixels.
- IBASIC lower display: width=861 pixels, height=158 pixels.

There is a set of queries that can be used to determine the size and location of the display window in use.

These queries, listed below, return the width and height of the window or the absolute location of its lower left or upper right corners. All the coordinates and sizes are dimensioned in pixels.

```
DISPlay:WINDow[1|2|10]:GEOMetry:LLEFt?
DISPlay:WINDow[1|2|10]:GEOMetry:SIZE?
DISPlay:WINDow[1|2|10]:GEOMetry:URIGht?
```

Note

The origin of EVERY graphics window is its lower left corner. The locations returned in response to the `LLEFt` and `URIGht` are relative to the ABSOLUTE origin of the entire display, NOT to the graphics window.

The Graphics Buffer

The analyzer has a graphics buffer that is used to refresh the graphics display if needed. When the buffer is full, additional graphics can still be drawn — BUT they will not be refreshed. The graphics buffer can be turned on and off using the following command (which is used in the `GRAPHICS` example program).

```
DISPlay:WINDow:GRAPhics:BUFFer[:STATe] <ON|OFF>
```

The graphics buffer will hold up to:

500 lines
40 circles
40 rectangles
50 strings (60 characters long)

ManualsPlus.com

Front Panel Keycodes

Each key on the front panel of the HP 8711 produces a "key code" when it is pressed. It is possible to trap key presses using the Device Status register. Demonstrations of this capability are included in the `REFLCAL` and `GRAPHICS` example programs. Information about the key press can be obtained using the following commands.

```
SYSTem:KEY:QUEue:CLEar
SYSTem:KEY:COUNt?
SYSTem:KEY:QUEue:MAXimum?
SYSTem:KEY:QUEue[:STATe]
SYSTem:KEY:QUEue[:STATe]?
SYSTem:KEY:TYPE?
  returns: NONE, RPG (knob), KEY (front panel), ASC (external keyboard)
SYSTem:KEY[:VALue]?
  returns: knob count (RPG), key code (KEY), ASCII code (ASC)
```

Key Group	Key Label	Key Code
Softkeys	Softkey 1 (top key)	0
	Softkey 2	1
	Softkey 3	2
	Softkey 4	3
	Softkey 5	4
	Softkey 6	5
	Softkey 7	6
	Softkey 8 (bottom key)	7
Numeric Keys	0 (zero)	10
	1 (one)	11
	2 (two)	12
	3 (three)	13
	4 (four)	14
	5 (five)	15
	6 (six)	16
	7 (seven)	17
	8 (eight)	18
	9 (nine)	19
	ENTER	20
	. (decimal)	21

Key Group	Key Label	Key Code
Numeric Keys	— / ← (minus/backspace)	22
	⇑ (step up)	23
	⇓ (step down)	24
Feature Keys	BEGIN	40
	CHAN 1	41
	CHAN 2	42
	POWER	43
	MENU	44
	FREQ	45
	SWEEP	46
	CAL	47
	DISPLAY	48
	SCALE	49
	AVG	50
	FORMAT	51
	MARKER	52
	SAVE/RECALL	53
	SYSTEM/OPTIONS	54
	HARD/COPY	55
	PRESET	56

ManualsPlus.com

Example Programs

The example programs listed in this manual are all written in IBASIC (HP Instrument BASIC). An optional internal controller can be purchased with an HP 8711 RF Network Analyzer (option 1C2). This controller runs IBASIC directly on the analyzer. It controls the analyzer over an internal interface bus that operates the same way as the external HP-IB interface. For more information about IBASIC refer to the *HP Instrument BASIC User's Handbook* or *Using HP Instrument BASIC with the HP 8711*.

IBASIC is a programming language that developed from HP BASIC. Because of this relationship, programs written for IBASIC can be run on external controllers that run HP BASIC. In addition to running on external workstations, the same programs can be run on PC-compatibles using IBASIC for Windows.

The example programs are provided on two disks that are included with the *HP 8711 Operating Manual*. Both disks contain the same examples written in IBASIC, only the disk format is different. Because the HP 8711's internal floppy disk drive is designed to be both DOS and LIF compatible, both disks can be used to supply programs for the analyzer's internal IBASIC controller.

- *Example Programs Disk -DOS Format* has programs that can be run on the instrument's internal controller or on a PC-compatible running IBASIC for Windows.
- *Example Programs Disk - LIF Format* has programs can be run on the instrument's internal controller or on an HP series 300 workstation running HP BASIC (version 3.0 or higher).

ManualsPlus.com

Following is a list of all the example programs included on both disks.

SETUP — Setting up a basic measurement. The example also demonstrates the use of the `*WAI` command.

MARKERS — Transferring data using markers. The example also demonstrates the use of the query form of command mnemonics.

LIMITEST — Performing automatic PASS/FAIL testing with limit lines. The example also demonstrates some methods of combining mnemonics for more efficient programming.

ASCDATA — Transferring data using the ASCII format. The section also lists the different data arrays (and their mnemonics) that can be accessed.

REALDATA — Transferring data using the IEEE 64-bit floating point REAL format. The example also demonstrates block data transfers of both indefinite length and definite length syntax. Also demonstrated is access to the swapped-byte data format designed for PCs.

INTDATA — Transferring data using the 16-bit INTEGER format.

TRANCAL — Performing a transmission calibration. The calibration is **User Defined** (performed over the instruments current source settings). This example also demonstrates the use of the `*OPC?` command.

REFLCAL — Performing a reflection calibration. The calibration is **Full Band** (performed over the instrument's preset source settings). This example also demonstrates the detection of front panel key presses, the use of softkeys, and the use of the `*OPC?` command.

LOADCALS — Uploading and downloading correction arrays. The data transfer is performed in the 16-bit integer format. The arrays must be dimensioned properly for both the number of data points and the format of the data being transferred.

LEARNSTR — Using the learn string to upload and download instrument states.

SAVERCL — Saving and recalling instrument states, calibrations and data. The example also demonstrates saving data in an ASCII file that includes both magnitude and frequency information.

PRINTPLT — Using the serial and parallel ports for hardcopy output. The example also demonstrates plotting test results to an HPGL file.

PASSCTRL — Using pass control and the HP-IB for hardcopy output. The example uses an HP-IB printer.

AVERAGE — Generating a service request interrupt. The example uses the status reporting structure to generate an interrupt as soon as averaging is complete.

GRAPHICS — Using graphics and softkeys to create customized procedures. The example demonstrates the use of some of the user graphics commands including the one to erase a previously drawn line. It also demonstrates use of the softkeys and detecting a front panel keypress with the service request interrupt process.

CALKIT — Instrument state file for downloading User Defined cal kit definitions. This example is NOT a program. It is an instrument state file example. This type of file enables the user to calibrate the analyzer for use with connector types that are not in the firmware. To see an example of using this feature, refer to "To Calibrate with Other Cal Kits" in chapter 6 of the *Operating Manual*.

ManualsPlus.com

Because the examples are designed to run in different environments, the setup at the beginning of each program must determine the operating environment and properly set the analyzer's HP-IB address. In these examples, the internal IBASIC controller uses the address 800 when communicating with the analyzer (the internal HP-IB is at select code 8). The default address of 716 is used when the programs are being run on an external controller.

A version of the following lines is included in all of the example programs. The use of the `Isc` variable (for addressing commands to the bus itself) and the `Internal` (internal-controller) flag vary due to differences in the programs needs.

10	`IF SYSTEM$("SYSTEM ID")="HP 8711" THEN`	Identify the operating system.
20	`  ASSIGN @Hp8711 TO 800`	If internal, set address to 800.
30	`  Isc=8`	Set interface select code to 8 (internal).
35	`  Internal=1`	Set internal-control flag to 1.
40	`ELSE`	
50	`  ASSIGN @Hp8711 TO 716`	If external, set address to 716.
60	`  Isc=7`	Set interface select code to 7 (external).
65	`  Internal=0`	Set internal-control flag to 0.
70	`  ABORT 7`	Abort all bus transactions and give active control of the bus to the computer.
80	`  CLEAR 716`	Send a selected device clear (SDC) to the analyzer – this clears all HP-IB errors, resets the HP-IB interface and clears syntax errors. (It does not affect the status reporting system.)
90	`END IF`	

ManualsPlus.com

SETUP — Setting up a basic measurement

This program demonstrates how to set up the analyzer to make a basic measurement. The *WAI command is used extensively throughout this program. This has the effect of making sure that the commands are executed in the order they are received. More information about making measurements with the HP 8711 is available in the *HP 8711 Operating Manual*.

Lines 10-70 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

```
10   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
20     ASSIGN @Hp8711 TO 800
30   ELSE
40     ASSIGN @Hp8711 TO 716
50     ABORT 7
60     CLEAR 716
70   END IF
80   OUTPUT @Hp8711;"SYST:PRES;*WAI"
90   OUTPUT @Hp8711;"CONF 'FILT:TRAN';*WAI"
100  OUTPUT @Hp8711;"ABOR;:INIT:CONT OFF;*WAI"
110  OUTPUT @Hp8711;"SENS2:STAT ON; *WAI"
120  OUTPUT @Hp8711;"SENS2:FUNC 'XFR:POW:RAT 1,0';
     DET NBAN;*OPC?"
125  ENTER @Hp8711;Opc
130  INPUT "Enter Start Frequency (MHz):",Start_f
140  INPUT "Enter Stop Frequency (MHz):",Stop_f
150  OUTPUT @Hp8711;"SENS2:FREQ:STAR";Start_f;
     "MHZ;STOP";Stop_f;"MHZ;*WAI"
160  OUTPUT @Hp8711;"INIT;*OPC?"
170  ENTER @Hp8711;Opc
180  OUTPUT @Hp8711;"DISP:WIND1:TRAC:Y:PDIV 10 dB;
     RLEV 0 dB;RPOS 8"
190  OUTPUT @Hp8711;"DISP:WIND2:TRAC:Y:PDIV 5 DB;
     RLEV 0 dB;RPOS 8"
200  DISP "Start =";Start_f;"MHz     Stop =";
     Stop_f;"MHz"
210  END
```

Preset the instrument.

Configure the analyzer to measure transmission of a filter on channel 1 – this is the command for the (BEGIN) **Filter** **Transmissn** key sequence.

Put the instrument in trigger hold mode.

Turn on channel 2.

Configure channel 2 to measure reflection – this is the command for the (CHAN 2) **Reflection** key sequence.

Wait for the previous commands to complete execution.

Input a start frequency.

Input a stop frequency.

Set the start and stop frequencies of the analyzer to the values entered.

Trigger a single sweep.

Wait for the sweep to complete.

Set up the scale and reference parameters for channel 1.

Set up the scale and reference parameters for channel 2.

Display the current start and stop frequencies.

ManualsPlus.com

MARKERS — Transferring data using the markers

This program demonstrates how to transfer measurement data by using the markers. Before any data is read over the HP-IB a controlled sweep should be taken. The analyzer has the ability to process and execute commands very quickly when they are received over the HP-IB. This speed can lead to commands (such as marker searches) being executed before any data has been taken. To ensure that the sweep has completed and the data is present before it is read, the command for a single sweep is used before data is requested. Note that `*WAI` is sent with that command. More information about making measurements with the HP 8711 is available in the *HP 8711 Operating Manual*.

Lines 10-70 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

```
10   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
20     ASSIGN @Hp8711 TO 800
30   ELSE
40     ASSIGN @Hp8711 TO 716
50     ABORT 7
60     CLEAR 716
70   END IF
80   OUTPUT @Hp8711;"SENS1:STAT ON;FREQ:
     STAR 10 MHZ;STOP 400 MHZ;*WAI"
```

Turn on channel 1 and set up start and stop frequencies for the example – these frequencies were chosen for the demonstration filter that is shipped with the analyzer.

```
90   OUTPUT @Hp8711;"SENS1:FUNC 'XFR:POW:RAT 2,0';
     DET NBAN;*WAI"
```

Configure a transmission measurement on channel 1 using the narrowband detection mode.

```
100  OUTPUT @Hp8711;"ABOR;:INIT:CONT OFF;
     :INIT;*WAI"
```

Take a single controlled sweep and have the analyzer wait until it has completed before executing the next command.

```
110  OUTPUT @Hp8711;"CALC1:MARK1 ON"
```

Turn on the first marker.

```
120  OUTPUT @Hp8711;"CALC1:MARK1:X 175 MHZ"
```

Set marker 1 to a frequency of 175 MHz.

```
130  OUTPUT @Hp8711;"CALC1:MARK1:Y?"
```

Query the amplitude of the signal at 175 MHz.

```
140  ENTER @Hp8711;Data1
```

Read the data – the data is in the NR3 format.

```
150  DISP "Marker 1 (175 MHz) = ";Data1
160  WAIT 5
170  OUTPUT @Hp8711;"CALC1:MARK2 ON;MARK2:MAX"
```

Turn on the second marker and use a marker search function to find the maximum point on the data trace.

ManualsPlus.com

`180 OUTPUT @Hp8711;"CALC1:MARK2:X?;Y?"`	Query the frequency and amplitude of the maximum point – note that the two queries can be combined into one command.
`190 ENTER @Hp8711;Freq_2,Data_2`	Read the data.
`200 DISP "Max = ";Data_2;"dB at";Freq_2/1.E+6; "MHz"`	Display the results of the marker search.
`210 OUTPUT @Hp8711;"INIT:CONT ON;*WAI"`	Put the analyzer into its continuously sweeping mode – this mode works very well for tuning applications.
`220 OUTPUT @Hp8711;"CALC1:MARK2:FUNC:TRAC ON"`	Turn on the marker search tracking function. This function causes the marker 2 to track the maximum value each time the analyzer takes a sweep.
`230 WAIT 5`	
`240 OUTPUT @Hp8711;"CALC1:MARK2 OFF"`	Turn off marker 2.
`250 OUTPUT @Hp8711;"ABOR;:INIT:CONT OFF; :INIT;*WAI"`	Take a single controlled sweep.
`260 OUTPUT @Hp8711;"CALC1:MARK:BWID -3;FUNC:RES?"`	Perform a search for the -3 dB bandwidth of the filter. This function uses several markers to find four key values.
`270 ENTER @Hp8711;Bwid,Center_f,Q,Loss`	Read the four values – the bandwidth, center frequency, Q and the insertion loss.
`280 DISP "BW: ";Bwid,Center_f,Q,Loss`	Display the bandwidth search values on the screen.
`290 OUTPUT @Hp8711;"CALC1:MARK:AOFF"`	Turn off all the markers.
`300 END`	

ManualsPlus.com

LIMITEST — Performing automatic PASS/FAIL testing with limit lines

This program demonstrates how to set up and use limit lines over the HP-IB. The example device used in this program is the demonstration filter that is shipped with the analyzer. The program sets up the basic measurement, downloads the limit lines and uses the status registers to determine of the device passes its specifications. For more information about limit lines, refer to the *HP 8711 Operating Manual*. For information about using the status registers, refer to the previous section "Using the Status Registers."

This example also demonstrates how multiple command mnemonics can be combined together. The easiest commands to combine are ones that are closely related on the command tree (such as the start and stop frequency of a limit segment). For more information of command mnemonics, refer to the *HP-IB Command Reference*.

Lines 20-80 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

```
10   DIM Title$[30]
20   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
30     ASSIGN @Hp8711 TO 800
40   ELSE
50     ASSIGN @Hp8711 TO 716
60     ABORT 7
70     CLEAR 716
80   END IF
90   OUTPUT @Hp8711;"SYST:PRES;*WAI"
```

Perform a system preset – this clears the limit table.

```
100  OUTPUT @Hp8711;"SENS1:FREQ:STAR 10 MHZ;
     STOP 400 MHZ;*WAI"
```

Set up the source frequencies for the measurement.

```
110  OUTPUT @Hp8711;"SENS1:FUNC 'XFR:POW:RAT 2,0';
     DET NBAN;*WAI"
```

Set up the receiver for the measurement parameters (Transmission in this case).

```
120  OUTPUT @Hp8711;"DISP:WIND1:TRAC:Y:PDIV 10 DB;
     RLEV 0 DB;RPOS 9"
```

Configure the display so measurement results are easy to see.

```
130  OUTPUT @Hp8711;"DISP:ANN:YAX OFF"
```

Reduce the distractions on the display by getting rid of notation that won't be used in this example.

```
140  OUTPUT @Hp8711;"DISP:WIND1:TRAC:GRAT:
     GRID OFF"
```

Erase the graticule grid for the same reason.

```
150  OUTPUT @Hp8711;"CALC1:LIM:SEGM1:TYPE LMAX;
     STAT ON"
```

Create and turn on the first segment for the new limit lines – this one is a maximum limit.

```
160  OUTPUT @Hp8711;"CALC1:LIM:SEGM1:AMPL:
     STAR -70;STOP -70"
```

Set the amplitude limits for the first limit segment.

```
170  OUTPUT @Hp8711;"CALC1:LIM:SEGM1:FREQ:
     STAR 10 MHZ;STOP 75 MHZ"
```

Set the frequency range of the first limit segment.

ManualsPlus.com

```
180  OUTPUT @Hp8711;"CALC1:LIM:SEGM2:TYPE LMAX;
     STAT ON"
```
Create and turn on a second maximum limit segment.

```
190  OUTPUT @Hp8711;"CALC1:LIM:SEGM2:AMPL:STAR 0;
     STOP 0"
```
Set the amplitude limits for segment 2.

```
200  OUTPUT @Hp8711;"CALC1:LIM:SEGM2:FREQ:
     STAR 145 MHZ;STOP 200 MHZ"
```
Set the frequency range for segment 2.

```
210  OUTPUT @Hp8711;"CALC1:LIM:SEGM3:TYPE LMIN;
     STAT ON"
```
Create and turn on a third limit segment – this one is a minimum limit.

```
220  OUTPUT @Hp8711;"CALC1:LIM:SEGM3:AMPL:STAR -6;
     STOP -6"
```
Set the amplitude limits for segment 3.

```
230  OUTPUT @Hp8711;"CALC1:LIM:SEGM3:FREQ:
     STAR 150 MHZ;STOP 195 MHZ"
```
Set the frequency range for segment 3.

```
240  OUTPUT @Hp8711;"CALC1:LIM:SEGM4:TYPE LMAX;
     STAT ON"
```
Create and turn on a fourth limit segment – another maximum segment.

```
250  OUTPUT @Hp8711;"CALC1:LIM:SEGM4:AMPL:
     STAR -60;STOP -60"
```
Set the amplitude limits for segment 4.

```
260  OUTPUT @Hp8711;"CALC1:LIM:SEGM4:FREQ:
     STAR 290 MHZ;STOP 400 MHZ"
```
Set the frequency range for segment 4.

```
270  OUTPUT @Hp8711;"*OPC?"
```
Send an operation complete query to ensure that all overlapped commands have been executed.

```
280  ENTER @Hp8711;Opc
```
Wait for the reply.

```
290  OUTPUT @Hp8711;"CALC1:LIM:DISP ON"
```
Turn on display of the limit lines.

```
300  OUTPUT @Hp8711;"CALC1:LIM:STAT ON"
```
Turn on the pass/fail testing –watch the analyzer's display for the pass/fail indicator.

```
310  OUTPUT @Hp8711;"ABOR;:INIT1:CONT OFF;
     :INIT1;*WAI"
```
Take a controlled sweep to ensure that there is real data present for the limit test.

```
320  OUTPUT @Hp8711;"STAT:QUES:LIM:COND?"
```
Query the limit fail condition register to see if there is a failure.

```
330  ENTER @Hp8711;Fail_flag
```
Read the register's contents.

```
340  IF BIT(Fail_flag,0)=1 THEN
```
Bit 0 is the test result for channel 1 while bit 1 is the result for channel 2 testing.

```
350    Title$="Limit Test FAIL - Tune device"
```
In case of a failure give additional direction to the operator using the title strings.

```
360    OUTPUT @Hp8711;"DISP:ANN:TITL1:DATA '"
       &Title$&"';STAT ON"
```
Turn on the title string.

```
370    OUTPUT @Hp8711;"INIT1:CONT ON;*WAI"
```
Turn on continuous sweep mode for tuning.

```
380    LOOP
```
Loop while the tuning is taking place.

```
390      OUTPUT @Hp8711;"STAT:QUES:LIM:COND?"
```
Monitor the status of the limit fail condition register.

```
400      ENTER @Hp8711;Fail_flag
410    EXIT IF BIT(Fail_flag,0)=0
```
Check the limit fail bit. Exit if the device has been tuned to pass the test.

ManualsPlus.com

```
420     END LOOP
430   END IF
440   OUTPUT @Hp8711;"DISP:ANN:TITL1 OFF;
      :INIT1:CONT ON;*WAI"

450   END
```

Turn off the prompt to the operator and return the analyzer to the continuously sweeping mode.

ManualsPlus.com

ASCDATA — Transferring data using the ASCII format

This program demonstrates how to read data arrays from the analyzer and write them back again. The ASCii data format is being used with a resolution of 5 digits. More information about data transfer is available in the "Transferring Data" section earlier in this chapter.

In addition to the channel 1 formatted data array used in this example, there are a number of arrays that can be accessed inside the instrument. These arrays and their corresponding mnemonics are listed below.

CH1FDATA	Formatted data – channel 1
CH2FDATA	Formatted data – channel 2
CH1SDATA	Error corrected but unformatted data – channel 1
CH2SDATA	Error corrected but unformatted data – channel 2
CH1SMEM	Trace memory, error corrected but unformatted – channel 1
CH2SMEM	Trace memory, error corrected but unformatted – channel 1
CH1AFWD	Raw uncorrected measurement data from the A (reflection) channel – channel 1
CH1BFWD	Raw uncorrected measurement data from the B (transmission) channel – channel 1
CH1RFWD	Raw uncorrected measurement data from the R (reference) channel – channel 1
CH2AFWD	Raw uncorrected measurement data from the A (reflection) channel – channel 2
CH2BFWD	Raw uncorrected measurement data from the B (transmission) channel – channel 2
CH2RFWD	Raw uncorrected measurement data from the R (reflection) channel – channel 2
CH1SCORR1	First correction array – channel 1
CH1SCORR2	Second correction array – channel 1
CH1SCORR3	Third correction array – channel 1
CH2SCORR1	First correction array – channel 2
CH2SCORR2	Second correction array – channel 2
CH2SCORR3	Third correction array – channel 2

Lines 20-80 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

```
10   REAL Data1(1:51)
20   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
30     ASSIGN @Hp8711 TO 800
40   ELSE
50     ASSIGN @Hp8711 TO 716
60     ABORT 7
70     CLEAR 716
80   END IF
```

ManualsPlus.com

```
90   OUTPUT @Hp8711;"SENS1:SWE:POIN 51;*WAI"
```
Set the analyzer to measure 51 data points.

```
100  OUTPUT @Hp8711;"ABOR;:INIT1:CONT OFF;
     :INIT1;*WAI"
```
Take a single sweep, leaving the analyzer in trigger hold mode.

```
110  OUTPUT @Hp8711;"FORM:DATA ASC,5;:TRAC?
      CH1FDATA"
```
Set up the ASCii data format with 5 significant digits and request the channel 1 formatted data array from the instrument.

```
120  ENTER @Hp8711;Data1(*)
```
Enter the data array.

```
130  DISP Data1(1),Data1(2),Data1(3),".... "
```
Display the first three numbers in the data array.

```
140  WAIT 10
```
Use this time to visually compare the numbers to the visible data trace.

```
150  DISP "Disconnect the test device --
     Press Continue"
160  PAUSE
```
Wait for the operator to disconnect the test device and continue the program.

```
165  DISP
```
Clear the display line.

```
170  OUTPUT @Hp8711;":INIT1;*WAI"
```
Take a sweep so the display shows new data.

```
180  WAIT 5
190  OUTPUT @Hp8711;"TRAC CH1FDATA";
```
Prepare the analyzer to receive the data, suppress the "end" character by using the semicolon.

```
200  FOR I=1 TO 51
210    OUTPUT @Hp8711;", ";Data1(I);
```
Send the data array one point at a time, using the semicolon to suppress the "end" character.

```
220  NEXT I
230  OUTPUT @Hp8711;""
```
Send an "end" character to terminate the input.

```
240  DISP "DATA TRANSFER COMPLETE!"
250  END
```

ManualsPlus.com

REALDATA — Transferring data using the IEEE 64-bit floating point REAL format

This program demonstrates how to read data arrays from the analyzer and write them back again. The REAL,64 data format is being used. Note that the analyzer outputs the data using the definite length block syntax. This example uses the indefinite length block syntax when data is being written back to the analyzer. More information about data transfer is available in the "Transferring Data" section earlier in this chapter. All of the arrays listed in the ASCDATA example section can also be accessed using this data format.

Lines 30-70 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

Lines 120-140 check the operating system to determine if it is IBASIC for Windows (a PC based system). When running on a PC based system, swapped byte order must be used for binary data transfers.

```
10   DIM A$[10],Data1(1:51)
20   INTEGER Digits,Bytes
30   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
40     ASSIGN @Hp8711 TO 800
50   ELSE
60     ASSIGN @Hp8711 TO 716
70     ABORT 7
80     CLEAR 716
90   END IF
```

100	`OUTPUT @Hp8711;"SENS1:SWE:POIN 51;*WAI"`	Set up the analyzer to measure 51 data points.
110	`OUTPUT @Hp8711;"ABOR;:INIT1:CONT OFF; :INIT1;*WAI"`	Take a single sweep, leaving the analyzer in trigger hold mode.
120	`OUTPUT @Hp8711;"FORM:DATA REAL,64;BORD NORM"`	Set up normal byte order and the REAL,64 data format.
130	`IF SYSTEM$("SYSTEM ID")="IBASIC/WINDOWS" THEN`	If operating system is a PC:
140	`  OUTPUT @Hp8711;"FORM:BORD SWAP"`	Set up byte swapping.
150	`END IF`	
160	`OUTPUT @Hp8711;"TRAC? CH1FDATA"`	Request the channel 1 formatted data array from the instrument.
170	`ASSIGN @Hp8711;FORMAT ON`	Turn on ASCII formatting on the I/O path – it is needed for reading the header information.
180	`ENTER @Hp8711 USING "%,A,D";A$,Digits`	Enter the data header. A$ is the # sign indicating a block data transfer, Digits is the number of digits in the number of bytes.
190	`ENTER @Hp8711 USING "%,"&VAL$(Digits)&"D"; Bytes`	Enter the number of bytes in the data array – note the use of Digits in the command.

ManualsPlus.com

```
200  ASSIGN @Hp8711;FORMAT OFF
```
Turn off ASCII formatting on the I/O path – it is not needed for transferring binary formatted data.

```
210  ENTER @Hp8711;Data1(*)
```
Enter the data array.

```
220  ASSIGN @Hp8711;FORMAT ON
```
Turn on ASCII formatting.

```
230  ENTER @Hp8711;A$
```
Enter the "end of data" character.

```
240  DISP Data1(1),Data1(2),".... "
```
Display the first two numbers in the data array.

```
250  WAIT 10
```
Use this time to visually compare the numbers to the visible data trace.

```
260  DISP "Disconnect the test device --
     Press Continue"
270  PAUSE
```
Wait for the operator to disconnect the test device and continue the program.

```
275  DISP
```
Clear the display line.

```
280  OUTPUT @Hp8711;":INIT1;*WAI"
```
Take a sweep so the display shows new data.

```
290  WAIT 5
300  OUTPUT @Hp8711;"TRAC CH1FDATA, #0";
```
Send the header for an indefinite block length data transfer.

```
310  ASSIGN @Hp8711;FORMAT OFF
```
Turn off ASCII formatting.

```
320  OUTPUT @Hp8711;Data1(*),END
```
Send the data back to the analyzer.

```
330  ASSIGN @Hp8711;FORMAT ON
```
Turn on ASCII formatting.

```
340  DISP "DATA TRANSFER COMPLETE!"
350  END
```

ManualsPlus.com

INTDATA — Transferring data using the 16-bit INTEGER format

This program demonstrates how to read data arrays from the analyzer and write them back again. The `INTeger,16` data format is being used. This data format is the instrument's internal format. It should only be used to read data that will later be returned to the instrument.

The data array dimensioned in line 20 is different from the arrays in either `REAL,64` or `ASCii` examples. This is because each data point is represented by a set of three 16-bit integers. Another difference in using this data format is that all arrays cannot be accessed with it. The formatted data arrays `CH1FDATA` and `CH2FDATA` cannot be read using the `INTEGER` format.

Note that the analyzer outputs the data using the definite length block syntax. This example uses the indefinite length block syntax when data is being written back to the analyzer. More information about data transfer is available in the "Transferring Data" section earlier in this chapter.

Lines 30-70 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

Lines 120-140 check the operating system to determine if it IBASIC for Windows (a PC based system). When running on a PC based system, swapped byte order must be used for binary data transfers.

```
10   DIM A$[10]
20   INTEGER Digits,Bytes,Data1(1:51,1:3)
30   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
40     ASSIGN @Hp8711 TO 800
50   ELSE
60     ASSIGN @Hp8711 TO 716
70     ABORT 7
80     CLEAR 716
90   END IF
100  OUTPUT @Hp8711;"SENS1:SWE:POIN 51;*WAI"          Set up the analyzer to measure 51 data
                                                      points.
110  OUTPUT @Hp8711;"ABOR;:INIT1:CONT OFF;            Take a single sweep, leaving the
     :INIT1;*WAI"                                     analyzer in trigger hold mode.
120  OUTPUT @Hp8711;"FORM:DATA INT,16;BORD NORM"      Set up normal byte order and the
                                                      INTeger,16 data format.
130  IF SYSTEM$("SYSTEM ID")="IBASIC/WINDOWS" THEN    If operating system is a PC:
140    OUTPUT @Hp8711;"FORM:BORD SWAP"                Set up byte swapping.
150  END IF
160  OUTPUT @Hp8711;"TRAC? CH1SDATA"                  Request the channel 1 unformatted
                                                      data array from the instrument.
170  ASSIGN @Hp8711;FORMAT ON                         Turn on ASCII formatting on the I/O
                                                      path – it is needed for reading the
                                                      header information.
```

ManualsPlus.com

```
180  ENTER @Hp8711 USING "%,A,D";A$,Digits
```
Enter the data header. **A$** is the # sign indicating a block data transfer, **Digits** is the number of digits in the number of bytes.

```
190  ENTER @Hp8711 USING "%,"&VAL$(Digits)&"D";
     Bytes
```
Enter the number of bytes in the data array – note the use of **Digits** in the command.

```
200  ASSIGN @Hp8711;FORMAT OFF
```
Turn off ASCII formatting on the I/O path – it is not needed for transferring binary formatted data.

```
210  ENTER @Hp8711;Data1(*)
```
Enter the data array.

```
220  ASSIGN @Hp8711;FORMAT ON
```
Turn on ASCII formatting.

```
230  ENTER @Hp8711;A$
```
Enter the "end of data" character.

```
240  DISP Data1(1,1),Data1(1,2),Data1(1,3),".... "
```
Display the first three numbers in the data array – there is no visible similarity between these numbers and the data displayed on the analyzer.

```
250  WAIT 10
```
Use this time to visually compare the numbers to the visible data trace.

```
260  DISP "Disconnect the test device --
     Press Continue"
270  PAUSE
```
Wait for the operator to disconnect the test device and continue the program.

```
275  DISP
```
Clear the display line.

```
280  OUTPUT @Hp8711;":INIT1;*WAI"
```
Take a sweep so the display shows new data.

```
290  WAIT 5
300  OUTPUT @Hp8711;"TRAC CH1SDATA, #0";
```
Send the header for an indefinite block length data transfer.

```
310  ASSIGN @Hp8711;FORMAT OFF
```
Turn off ASCII formatting.

```
320  OUTPUT @Hp8711;Data1(*),END
```
Send the data back to the analyzer.

```
330  ASSIGN @Hp8711;FORMAT ON
```
Turn on ASCII formatting.

```
340  DISP "DATA TRANSFER COMPLETE!"
350  END
```

ManualsPlus.com

TRANCAL — Performing a transmission calibration

This program demonstrates a transmission calibration performed over user-defined source settings (frequency range, power and number of points). The operation complete query is used at each step in the process to make sure the steps are taken in the correct order. More information on calibration is available in the *HP 8711 Operating Manual*.

Lines 10-70 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

```
10    IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
20      ASSIGN @Hp8711 TO 800
30    ELSE
40      ASSIGN @Hp8711 TO 716
50      ABORT 7
60      CLEAR 716
70    END IF
80    OUTPUT @Hp8711;"SENS1:FUNC 'XFR:POW: RAT 2,0';
       DET NBAN;*WAI"
90    OUTPUT @Hp8711;"SENS:CORR:COLL:CKIT 'COAX,7MM,
       TYPE-N,50,FEMALE'"
100   OUTPUT @Hp8711;"SENS1:CORR:COLL:IST OFF;
       METH TRAN1"
110   DISP "Connect THRU - Press Continue"
120   PAUSE
130   DISP "Measuring THRU"
140   OUTPUT @Hp8711;"SENS1:CORR:COLL STAN1;*OPC?"
150   ENTER @Hp8711;Opc
160   DISP "Calculating Error Coefficients"
170   OUTPUT @Hp8711;"SENS1:CORR:COLL:SAVE;*OPC?"
180   ENTER @Hp8711;Opc
190   DISP "User Defined TRANSMISSION CAL COMPLETED!"
200   END
```

Configure the analyzer to measure transmission on channel 1.

Select a calibration kit type.

Select a transmission calibration for the current source settings – the IST OFF *part of the mnemonic indicates that the current source settings will be used.*

Prompt the operator to make a through connection.

Pause until the thru connection is made and continue is pressed.

Measure the thru.

Wait until the measurement is complete.

Calculate the error coefficients and save the calibration to internal memory – this is not the same as saving the calibration using the SAVE RECALL *key.*

Wait for the calculations to complete and the calibration to be saved.

ManualsPlus.com

REFLCAL — Performing a reflection calibration

This program demonstrates a reflection calibration performed over the preset source settings (frequency range, power and number of points). The operation complete query is used at each step in the process to make sure the steps are taken in the correct order. More information on calibration is available in the *HP 8711 Operating Manual*.

Lines 20-100 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

```
10    DIM Msg$[50]
20    IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
30      ASSIGN @Hp8711 TO 800
40      Internal=1
50    ELSE
60      ASSIGN @Hp8711 TO 716
70      Internal=0
80      ABORT 7
90      CLEAR 716
100   END IF
110   OUTPUT @Hp8711;"SENS1:FUNC 'XFR:POW:RAT 1,0';
       DET NBAN;*WAI"                          ! Configure the analyzer to
                                               ! measure reflection on chan-
                                               ! nel 1.
120   OUTPUT @Hp8711;"SENS:CORR:COLL:CKIT 'COAX,7MM,
       TYPE-N,50,FEMALE'"                      ! Select a calibration kit type.
130   OUTPUT @Hp8711;"SENS1:CORR:COLL:IST ON;
       METH REFL3"                             ! Select a reflection calibra-
                                               ! tion for the preset source
                                               ! settings - the IST ON part
                                               ! of the mnemonic indicates
                                               ! that the preset source con-
                                               ! figuration will be used.
140   Msg$="Connect OPEN"
150   GOSUB Get_continue                      ! Prompt the operator to con-
                                               ! nect an open and press a
                                               ! key.
160   DISP "Measuring OPEN"
170   OUTPUT @Hp8711;"SENS1:CORR:COLL STAN1;*OPC?"  ! Measure the open.
180   ENTER @Hp8711;Opc                       ! Wait until the measurement
                                               ! is complete.
190   Msg$="Connect SHORT"
200   GOSUB Get_continue                      ! Prompt the operator to con-
                                               ! nect an short and press a
                                               ! key.
210   DISP "Measuring SHORT"
220   OUTPUT @Hp8711;"SENS1:CORR:COLL STAN2;*OPC?"  ! Measure the short.
230   ENTER @Hp8711;Opc                       ! Wait until the measurement
                                               ! is complete.
```

ManualsPlus.com

```
240   Msg$="Connect LOAD"
250   GOSUB Get_continue

260   DISP "Measuring LOAD"
270   OUTPUT @Hp8711;"SENS1:CORR:COLL STAN3;*OPC?"
280   ENTER @Hp8711;Opc

290   DISP "Calculating Error Coefficients"
300   OUTPUT @Hp8711;"SENS1:CORR:COLL:SAVE;*OPC?"

310   ENTER @Hp8711;Opc

320   DISP "Full Band REFLECTION CAL COMPLETED!"
330   STOP
340 Get_continue: !
350   IF Internal=1 THEN
360     DISP Msg$&" - Press Continue"

370     ON KEY 2 LABEL "Continue" RECOVER Go_on
380     LOOP

390     END LOOP
400   ELSE
410     OUTPUT @Hp8711;"SYST:KEY:QUE:CLE"

420     DISP Msg$&" - Press BEGIN to continue"

430     OUTPUT @Hp8711;"SYST:KEY:QUE ON"

440     LOOP

450       OUTPUT @Hp8711;"STAT:DEV:COND?"

460       ENTER @Hp8711;Dev_cond
470       IF BIT(Dev_cond,0)=1 THEN

480         OUTPUT @Hp8711;"SYST:KEY?"

490         ENTER @Hp8711;Key_code
500       END IF
510     EXIT IF Key_code=40
```

Prompt the operator to connect a load and press a key.

Measure the load.
Wait until the measurement is complete.

Calculate the error coefficients and save the calibration – this is not the same as saving the calibration using the SAVE RECALL *key.*
Wait for the calculations to complete and the calibration to be saved.

If internal control:
Use the display line for the prompt.
Use the softkeys for the response.
Loop while waiting for softkey 2 to be pressed.

If external control:
Clear the key queue so previous key presses will not interfere.
Use the BEGIN *key for the response.*
Turn on the key queue to trap all key presses.
Loop while waiting for a key to be pressed.
Read the device status condition register.

Check the bit that indicates a key press.
Read the code for the key that was pressed.

Stop looping if the BEGIN *key was pressed.*

ManualsPlus.com

```
520      END LOOP
530      Key_code=0
```

Clear the key_code to start fresh the next time the routine is called.

```
540    END IF
550 Go_on: !
560    OFF KEY
570    RETURN
580    END
```

Turn off the softkeys on the instrument and the computer. Return to the main body of the program.

ManualsPlus.com

LOADCALS — Uploading and downloading correction arrays

This program demonstrates how to read the correction arrays from the analyzer and write them back again. The INTeger,16 data format is being used because the data does not need to be interpreted, only stored and retrieved. More information about calibration is available in the *HP 8711 Operating Manual*.

The size of the arrays into which the data is read is critical. If they are not dimensioned correctly the program will not work. Most correction arrays, including the factory default (DEF) and the full band (FULL, preset source settings) arrays have 801 points. For user defined calibrations (USER) the number of points must be determined. If the number of points is other than 801 line 30 will need to be changed to allocate arrays for the correct number of points. The number of points can be found by reading the correction array's header and determining the size as shown in the example below.

Lines 40-110 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

```
10   DIM Func$[20],A$[10]
20   INTEGER Swap,Arrays,Digits,Bytes,Points
30   INTEGER Corr1(1:801,1:3),Corr2(1:801,1:3),
     Corr3(1:801,1:3)
40   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
50     ASSIGN @Hp8711 TO 800
60   ELSE
70     ASSIGN @Hp8711 TO 716
80     IF SYSTEM$("SYSTEM ID")="IBASIC/WINDOWS"
       THEN Swap=1                          If operating system is a PC set a flag
                                            for byte swapping.
90     ABORT 7
100    CLEAR 716
110  END IF
120  OUTPUT @Hp8711;"SENS1:FUNC?"          Query the measurement parameter.
130  ENTER @Hp8711;Func$                   Read the analyzer's response.
140  SELECT Func$
150  CASE """XFR:POW:RAT 2, 0"""           This is the transmission case – a ratio
                                            of the powers being measured by
                                            detector 2 (B) and detector 0 (R).
160    Arrays=1                            The transmission calibration has only
                                            one correction array.
170  CASE """XFR:POW:RAT 1, 0"""           This is the reflection case – a ratio of
                                            the powers being measured by detector
                                            1 (A) and detector 0 (R).
180    Arrays=3                            The reflection calibration has three
                                            correction arrays.
```

ManualsPlus.com

```
190  END SELECT
200  OUTPUT @Hp8711;"FORM:DATA INT,16"
210  OUTPUT @Hp8711;"FORM:BORD NORM"
220  IF Swap=1 THEN OUTPUT @Hp8711;
     "FORM:BORD SWAP"
230  OUTPUT @Hp8711;"TRAC? CH1SCORR1"
240  ASSIGN @Hp8711;FORMAT ON
250  ENTER @Hp8711 USING "%,A,D";A$,Digits
260  ENTER @Hp8711 USING "%,"&VAL$(Digits)&"D";
     Bytes
270  Points=Bytes/6
280  IF Points<>801 THEN
290    DISP "Arrays are not dimensioned for this
       calibration"
300    STOP
310  END IF
320  DISP "Uploading calibration arrays ... "
330  ASSIGN @Hp8711;FORMAT OFF
340  ENTER @Hp8711;Corr1(*)
350  ASSIGN @Hp8711;FORMAT ON
360  ENTER @Hp8711;A$
370  IF Arrays=3 THEN
380    OUTPUT @Hp8711;"TRAC? CH1SCORR2"
390    Read_array(@Hp8711,Corr2(*))
400    OUTPUT @Hp8711;"TRAC? CH1SCORR3"
410    Read_array(@Hp8711,Corr3(*))
420  END IF
430  DISP "Calibration arrays have been uploaded."
440  WAIT 5
450  DISP "Downloading calibration arrays ... "
460  OUTPUT @Hp8711;"SENS1:CORR:STAT OFF"
```

Select up the 16-bit integer binary data format.

Select the normal byte order.

If operating on a PC based system then change to the swapped byte order.

Request the first correction array from the analyzer.

Turn on ASCII formatting on the I/O path to read the file header.

Read the file header, including the number of digits needed to read the size of the file.

Read the size of the file.

Determine the number of points in the array from the size of the file.

This example was only designed for 801 points.

If another number of points is needed the arrays in line 30 above need to be dimensioned appropriately.

Turn off ASCII formatting on the I/O path.

Read the first correction array.

Turn on ASCII formatting.

Read the "end of file" character.

If this is a reflection calibration there are two more arrays.

Request the second correction array from the analyzer.

Read the second correction array.

Request the third correction array.

Read the third correction array.

Turn off correction before writing a calibration back into the analyzer.

ManualsPlus.com

```
470  OUTPUT @Hp8711;"SENS1:SWE:POIN";Points
```
Set the number of points for the correction arrays – even if the calibration was originally the default from the factory or a full band calibration, it is considered to be a user-defined calibration when it is downloaded.

```
480  OUTPUT @Hp8711;"TRAC CH1SCORR1, #0";
```
Prepare the analyzer to receive the first correction array in the indefinite block length format.

```
490  ASSIGN @Hp8711;FORMAT OFF
```
Turn off ASCII formatting.

```
500  OUTPUT @Hp8711;Corr1(*),END
```
Send the first correction array to the analyzer. The array is followed immediately by an "end of transfer" signal.

```
510  ASSIGN @Hp8711;FORMAT ON
```
Turn on ASCII formatting.

```
520  IF Arrays=3 THEN
```
If this is a reflection calibration there are two more arrays.

```
530    OUTPUT @Hp8711;"TRAC CH1SCORR2, ";
```
Prepare the analyzer to receive the second array.

```
540    Write_array(@Hp8711,Corr2(*))
```
Send the second array.

```
550    OUTPUT @Hp8711;"TRAC CH1SCORR3, ";
```
Prepare the analyzer to receive the third array.

```
560    Write_array(@Hp8711,Corr3(*))
```
Send the third array.

```
570  END IF
580  OUTPUT @Hp8711;"SENS1:CORR:STAT ON;*WAI"
```
Turn on the calibration.

```
590  DISP "Calibration arrays have been
     downloaded."
600  END
610  SUB Read_array(@Hp8711,INTEGER Array(*))
```
This subprogram contains the standard form used for transferring <block> data in the binary formats. It assumes that the command requesting the data has already been sent.

```
620    DIM A$[10]
```
Dimension a string character to receive the header and end of file symbols.

```
630    INTEGER Digits,Bytes
```
Declare integer variables for the number of digits and the number of bytes.

```
640    ASSIGN @Hp8711;FORMAT ON
```
Turn on ASCII formatting to read the header (some of the header is non-numeric).

```
650    ENTER @Hp8711 USING "%,A,D";A$,Digits
```
Enter the header character and the number of digits needed to read the size of the file.

```
660    ENTER @Hp8711 USING "%,"&VAL$(Digits)&"D";
       Bytes
```
Enter the number of bytes in the file (not including the header).

ManualsPlus.com

```
670    ASSIGN @Hp8711;FORMAT OFF
```
Turn off ASCII formatting for the transfer of binary data.

```
680    ENTER @Hp8711;Array(*)
```
Enter the data array.

```
690    ASSIGN @Hp8711;FORMAT ON
```
Turn on ASCII formatting to read the non-numeric character(s) at the end of the file.

```
700    ENTER @Hp8711;A$
```
Enter the end of file characters.

```
710  SUBEND
720  SUB Write_array(@Hp8711,INTEGER Array(*))
```
This subprogram contains the standard form used for writing `<block>` data to the analyzer in the binary formats. It assumes that the command preparing the analyzer for the data has already been sent and the I/O path has been left with ASCII formatting on.

```
730    OUTPUT @Hp8711;"#0";
```
Send the header, `#0` signifies a block data transfer of undefined length.

```
740    ASSIGN @Hp8711;FORMAT OFF
```
Turn off ASCII formatting to write the binary formatted data.

```
750    OUTPUT @Hp8711;Array(*),END
```
Write the binary formatted data array. Follow it immediately with a BASIC `END` statement that signifies the end of the data transfer.

```
760    ASSIGN @Hp8711;FORMAT ON
```
Turn on ASCII formatting.

```
770  SUBEND
```

ManualsPlus.com

LEARNSTR — Using the learn string to upload and download instrument states

This program demonstrates how to upload and download instrument states using the learn string. The learn string is a fast and easy way to read an instrument state. It is read out using the *LRN? query (an IEEE 488.2 common commands). To restore the learn string simply output the string to the analyzer.

The learn string contains a mnemonic at the beginning that tells the analyzer to restore the instrument state.

The learn string is transferred as a block. The header is ASCII formatted and the data is in the instrument's internal binary format. The number of bytes in the block of data is determined by the instrument state (no more than 20000 bytes).

```
"SYST:SET #<digits><bytes><learn string data>"
```

Lines 20-80 are explained in the introduction to the example programs section. They determine which system controller is being used and prepare the instrument for remote operation.

```
10   DIM Learnstr$[20000]
20   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
30     ASSIGN @Hp8711 TO 800
40   ELSE
50     ASSIGN @Hp8711 TO 716
60     ABORT 7
70     CLEAR 716
80   END IF
90   OUTPUT @Hp8711;"*LRN?"
```

Request the learnstring.

```
100  ENTER @Hp8711 USING "-K";Learnstr$
```

Read the learn string from the analyzer. The `USING "-K"` format allows the data being transmitted to include characters (such as the line feed character) that would otherwise terminate the input.

```
110  DISP "Learn string has been read"
120  WAIT 5
130  OUTPUT @Hp8711;"SYST:PRES;*OPC?"
```

Preset the analyzer.

```
140  ENTER @Hp8711;Opc
```

Wait for the preset operation to complete.

```
150  DISP "Instrument has been PRESET"
160  WAIT 5
170  OUTPUT @Hp8711;Learnstr$
```

Output the learnstring to the analyzer. The mnemonic is included in the string so no command is necessary.

```
180  DISP "Instrument state has been restored"
190  END
```

ManualsPlus.com

SAVERCL — Saving and recalling instrument states, calibrations and data

This program demonstrates how to save instrument states, calibrations and data to a mass storage device. The device used in this example is the HP 8711's internal floppy disk drive. The only change needed to use this program with the internal non-volatile memory is to change the mass storage unit specifier. The three choices are the internal floppy disk drive ('INT:'), the internal non-volatile memory ('MEM:') and an external HP-IB floppy disk drive ('EXT:'). To perform a save/recall to an external disk drive requires passing control of the HP-IB from the controller to the analyzer. For more information on passing control of the bus refer to the previous "Passing Control" section and the PASSCTRL example program.

Lines 10-70 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

Lines 80-130 are an example of saving an instrument state and calibration on the internal floppy disk drive.

Lines 190-200 are an example of recalling that instrument state and calibration.

Lines 210-230 are an example of saving a data trace (magnitude and frequency values) to an ASCII formatted file on the internal floppy disk drive. This file cannot be recalled into the instrument. It can, however, be imported directly into spreadsheets and word processors.

```
10   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
20     ASSIGN @Hp8711 TO 800
30   ELSE
40     ASSIGN @Hp8711 TO 716
50     ABORT 7
60     CLEAR 716
70   END IF
80   OUTPUT @Hp8711;"MMEM:MSIS 'INT:'"
90   OUTPUT @Hp8711;"MMEM:STOR:STAT:IST ON"
100  OUTPUT @Hp8711;"MMEM:STOR:STAT:CORR ON"
110  OUTPUT @Hp8711;"MMEM:STOR:STAT:TRAC OFF"
120  OUTPUT @Hp8711;"MMEM:STOR:STAT 1,'FILTER';
     *OPC?"
130  ENTER @Hp8711;Opc
140  DISP "Instrument state and calibration
     have been saved"
```

Select the internal floppy disk drive as the mass storage device.

Turn on the instrument state as part of the definition of "Save/Recall".

Turn on the calibration so it will be saved with the instrument state.

Turn off the trace data.

Save the current instrument state (STAT 1) into a file named 'FILTER'. Use *OPC? to make sure the operation is completed before any other operation begins.

Wait for the reply.

ManualsPlus.com

```
150  OUTPUT @Hp8711;"SYST:PRES;*OPC?"
```

Preset the instrument so the change in instrument state is easy to see.

```
160  ENTER @Hp8711;Opc
```

Wait for the preset to complete.

```
170  DISP "Instrument has been PRESET"
180  WAIT 5
190  OUTPUT @Hp8711;"MMEM:LOAD:STAT 1,
     'INT:FILTER';*OPC?"
```

Load the file 'FILTER' off the internal floppy disk drive. This becomes the new instrument state. Use the *OPC? query to hold off further commands until the analyzer is reconfigured.

```
200  ENTER @Hp8711;Opc
```

Wait for the reply.

```
210  OUTPUT @Hp8711;"ABOR;:INIT:CONT OFF;
     :INIT;*WAI"
```

Take a single sweep to make sure there is valid measurement data present.

```
220  DISP "Instrument state and calibration
     have been recalled"
230  OUTPUT @Hp8711;"MMEM:STOR:TRAC CH1FDATA,
     'INT:DATA0001'"
```

Save that measurement data into an ASCII file (named DATA0001) on the internal drive.

```
240  DISP "Data has been saved (ASCII format)"
250  END
```

ManualsPlus.com

PRINTPLT — Using the serial and parallel ports for hardcopy output

This program demonstrates how to send a hardcopy to a printer on the serial interface. This is done by selecting the appropriate device, setting up the baud rate and hardware handshaking, and sending the command to print or plot. The *OPC? query is used in this example to indicate when the printout is complete. Another method of obtaining the same results is to monitor the Hardcopy in Progress bit (bit 9 in the Operational Status Register). More information on printing or plotting is available in the *HP 8711 Operating Manual*.

Lines 10-70 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

Lines 80-150 demonstrate sending a hardcopy output to a printer connected to the serial port. The same program could be used to send hardcopy output to a device on the parallel port. The only changes would be deleting lines 100-110 and changing line 90 to read HCOP:DEV:PORT PAR.

Lines 160-260 demonstrate how to create an HPGL file (plotter language) and send it to the disk in the internal floppy disk drive. HPGL files are supported by many applications including the leading word processors and desktop publishing products.

```
10    IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
20      ASSIGN @Hp8711 TO 800
30    ELSE
40      ASSIGN @Hp8711 TO 716
50      ABORT 7
60      CLEAR 716
70    END IF
80    OUTPUT @Hp8711;"HCOP:DEV:LANG PCL;PORT SER"      ! Select the output language (PCL - printer control language). Select the output port (Serial).
90    OUTPUT @Hp8711;"SYST:COMM:SER:TRAN:BAUD 19200"   ! Select the baud rate.
100   OUTPUT @Hp8711;"SYST:COMM:SER:TRAN:HAND XON"     ! Select the handshaking protocol.
110   OUTPUT @Hp8711;"HCOP:DEV:MODE TABL"              ! Select the type of output - listing of values table.
120   OUTPUT @Hp8711;"HCOP;*OPC?"                      ! Send the command to start a hardcopy to the selected device. Use the *OPC? query to make sure the hardcopy is complete before continuing.
130   ENTER @Hp8711;Opc                                ! Wait for a reply.
140   DISP "Hardcopy to serial printer - COMPLETE!"
150   OUTPUT @Hp8711;"HCOP:DEV:LANG HPGL;PORT MMEM"    ! Select the HPGL output language (for graphics) and the output device (mass memory).
160   OUTPUT @Hp8711;"HCOP:ITEM:TRAC:STAT ON"          ! Include the data trace in the plot (default).
```

ManualsPlus.com

```
170   OUTPUT @Hp8711;"HCOP:ITEM:GRAT:STAT OFF"

180   OUTPUT @Hp8711;"HCOP:ITEM:ANN:STAT ON"

190   OUTPUT @Hp8711;"HCOP:ITEM:MARK:STAT ON"

200   OUTPUT @Hp8711;"HCOP:ITEM:TITL:STAT ON"

210   OUTPUT @Hp8711;"HCOP:DEV:MODE GMAR"

220   OUTPUT @Hp8711;"HCOP;*OPC?"

230   ENTER @Hp8711;Opc
240   DISP "Plot to floppy disk - COMPLETE!"

250   END
```

Do not show the graticule in the plot (default is on).
Include the frequency and measurement annotation in the plot (default).
Include the marker symbols in the plot (default).
Include the title and/or time/date stamp in the plot (default).
Define the plot to be both the graph and a marker table if markers are in use.
Begin the plot to a file, use `*OPC?`.
Wait for the reply.

ManualsPlus.com

PASSCTRL — Using pass control and the HP-IB for hardcopy output

This program demonstrates how to send a hardcopy to an HP-IB printer. This is done by passing active control of the bus to the analyzer so it can control the printer. More information about passing control to the analyzer is available in the "Passing Control" section earlier in this chapter.

Lines 10-90 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

```
10   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
20     ASSIGN @Hp8711 TO 800
30     Internal=1
40   ELSE
50     ASSIGN @Hp8711 TO 716
60     Internal=0
70     ABORT 7
80     CLEAR 716
90   END IF
100  OUTPUT @Hp8711;"HCOP:DEV:LANG PCL;PORT GPIB"     Select the output port (HP-IB) and
                                                      language (PCL – printer control
                                                      language).
110  OUTPUT @Hp8711;"SYST:COMM:GPIB:HCOP:ADDR 1"      Set the HP-IB address of the output
                                                      device.
120  OUTPUT @Hp8711;"HCOP:DEV:MODE GRAP"              Select the type of output – graph only.
130  IF Internal=1 THEN                               If the internal controller is being used:
140    OUTPUT @Hp8711;"SYST:COMM:GPIB:CONT ON"        make the analyzer the system controller.
150  END IF
160  OUTPUT @Hp8711;"*CLS"                            Clear all the event registers.
170  OUTPUT @Hp8711;"*ESE 2"                          Enable the Request Control bit in the
                                                      event status register.
180  OUTPUT @Hp8711;"*SRE 0"                          Clear the Service Request enable
                                                      register – SRQ is not being used.
190  OUTPUT @Hp8711;"HCOP"                            Start the hardcopy output.
200  LOOP
210    Stat=SPOLL(@Hp8711)                            Read the status byte using a serial poll.
220  EXIT IF BIT(Stat,5)=1                            Exit when the analyzer requests control
                                                      of the bus by setting bit 5 of the status
                                                      byte.
230  END LOOP
```

ManualsPlus.com

```
240  PASS CONTROL @Hp8711
250  DISP "Hardcopy in Progress"
260  IF Internal=1 THEN
270    OUTPUT @Hp8711;"*OPC?"
280    ENTER @Hp8711;Opc
290  ELSE
300    LOOP
310      STATUS 7,6;Hpib
320    EXIT IF BIT(Hpib,6)=1
330    END LOOP
340  END IF
350  DISP "HARDCOPY COMPLETE!"
360  END
```

Pass control of the bus to the analyzer.

If using the internal controller:

Send the operation complete query.

Wait for the reply when the hardcopy is complete.

If using an external controller:

Monitor the computer's status on the HP-IB. The **STATUS** command loads the interface 7 (HP-IB) register 6 – the computer's status with respect to HP-IB, into the variable **Hpib**.

Exit if control has returned – bit 6 is set when the computer is the active controller.

ManualsPlus.com

AVERAGE — Generating a service request interrupt

This program demonstrates generating a service request interrupt. The SRQ is used to indicate when averaging is complete. More information on service requests and the status registers is available in the "Using the Status Registers" section earlier in this chapter.

In this program, the `STATus:PRESet` executed in line 130 has the effect of setting all bits in the averaging status transition registers (positive transitions to 0, negative transitions to 1). It also sets up the operational status transition registers (positive transitions to 1, negative transitions to 0). These are the states needed to generate an interrupt when averaging is complete.

Lines 10-90 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

```
10   IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
20     ASSIGN @Hp8711 TO 800
30     Isc=8
40   ELSE
50     ASSIGN @Hp8711 TO 716
60     Isc=7
70     ABORT 7
80     CLEAR 716
90   END IF
100  OUTPUT @Hp8711;"*CLS"                          Clear all the event registers.
110  OUTPUT @Hp8711;"*SRE 0"                        Clear the Service Request Enable
                                                    register.
120  OUTPUT @Hp8711;"*ESE 0"                        Clear the Standard Event Status
                                                    Enable register.
130  OUTPUT @Hp8711;"STAT:PRES"                     Preset the remaining status registers.
140  OUTPUT @Hp8711;"STAT:OPER:ENAB 256"            Set operational status register to report
                                                    to the status byte on positive transition
                                                    of the averaging bit.
150  OUTPUT @Hp8711;"STAT:OPER:AVER:ENAB 1"         Set averaging status register to report
                                                    to the operational status register on
                                                    negative transition of the averaging
                                                    done bits.
160  OUTPUT @Hp8711;"*SRE 128"                      Enable the operational status bit in the
                                                    status byte to generate an SRQ.
```

ManualsPlus.com

```
170  ON INTR Isc,2 GOSUB Srq_handler
```
Set up a program branch to handle the interrupt.

```
180  ENABLE INTR Isc;2
```
Enable an interrupt from the interface (HP-IB) when the SRQ bit (bit 1 – decimal value 2) is set.

```
190  OUTPUT @Hp8711;"SENS1:AVER:COUN 8;*WAI"
```
Set an averaging factor of 8.

```
200  OUTPUT @Hp8711;"SENS1:AVER ON;AVER:CLE;*WAI"
```
Turn on and restart averaging.

```
210  OUTPUT @Hp8711;"ABOR;:INIT1:CONT ON;*WAI"
```
Turn on the continuously sweeping trigger mode.

```
220  Avg_done=0
```
Set a flag indicating that averaging is not complete.

```
230  LOOP
```
Loop while waiting for an interrupt.

```
240    DISP "Waiting for average to complete"
250  EXIT IF Avg_done=1
```
Check the flag and exit if averaging is done.

```
260  END LOOP
270  DISP "AVERAGING COMPLETE!"
280  STOP
290  !
300 Srq_handler:!
310  Stb=SPOLL(@Hp8711)
```
Determine if the HP 8711 generated the interrupt.

```
320  IF BINAND(Stb,128)<>0 THEN
```
Determine if the operational status register caused the interrupt.

```
330    OUTPUT @Hp8711;"STAT:OPER:EVEN?"
```
Read the operational status event register.

```
340    ENTER @Hp8711;Op_event
350    IF BINAND(Op_event,256)<>0 THEN
```
Determine if the averaging status register caused the interrupt.

```
360      Avg_done=1
```
Set the flag indicating that averaging is complete.

```
370    END IF
380  END IF
390  RETURN
```
Return to the main program.

```
400  END
```

ManualsPlus.com

GRAPHICS — Using graphics and softkeys to create a customized procedure

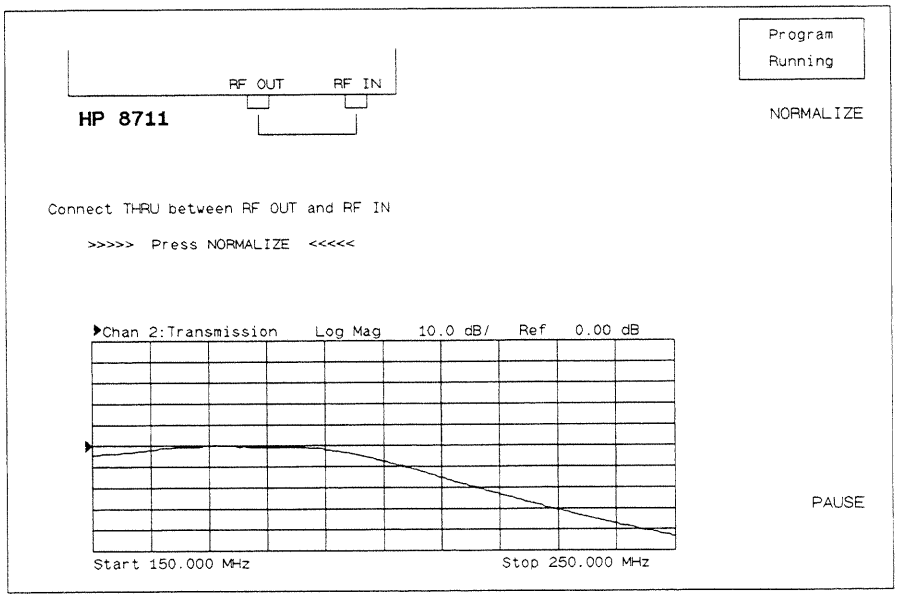
This program demonstrates how to use the HP 8711's user graphics commands to draw setup diagrams. It also demonstrates generating a service request in response to a keyboard interrupt. More information on user graphics commands is available in the previous section, "User Graphics" and in the *HP-IB Command Reference*. Information on generating a service request and using the status reporting structure is in the "Using the Status Registers" section.

Note that this program uses the HP 8711's user graphics commands. If the IBASIC option is installed, graphics may sometimes be more easily implemented using BASIC commands such as POLYGON and RECTANGLE. For further information, see the "BAR" program description in the *HP Instrument BASIC Users Handbook*.

Lines 10-110 are explained in the introduction to the example programs section. They determine which system controller is being used, set flags, and prepare the instrument for remote operation.

Lines 170-240 draw and label a representation of the HP 8711 for a connection diagram. This example is a simple front view from the top.

Lines 250-450 draw the connection needed for a normalization. The operator is prompted to make this connection and to press a softkey on the instrument. A flashing message is used to attract attention.



GRAPHICS example connection diagram

Lines 460-580 perform the normalization, erase the prompts (without erasing the whole screen) and prepare for the test.

Lines 590-730 are a branching routine that handles the service request generated interrupts used by the external controller.

ManualsPlus.com

```
10    IF SYSTEM$("SYSTEM ID")="HP 8711" THEN
20      ASSIGN @Hp8711 TO 800
30      Internal=1
40      Isc=8
50    ELSE
60      ASSIGN @Hp8711 TO 716
70      Internal=0
80      Isc=7
90      ABORT 7
100     CLEAR 716
110   END IF
```

```
120   OUTPUT @Hp8711;"DISP:PROG UPP"[1]
```
Allocate an IBASIC display partition to show the graphics.

```
130   OUTPUT @Hp8711;"DISP:WIND10:GRAP:CLE"[2]
```
Clear the IBASIC display partition.

```
140   OUTPUT @Hp8711;"SENS2:STAT ON;*WAI"
```
Turn on channel 2 for measurements – the upper part of the display is devoted to graphics so the lower part (channel 2) is used for measurements.

```
150   OUTPUT @Hp8711;"ABOR;:INIT2:CONT OFF;
      :INIT2;*OPC?"
```
Take a single controlled sweep to ensure good data.

```
160   ENTER @Hp8711;Opc
```
Wait for the `*OPC?` reply.

```
170   OUTPUT @Hp8711;"DISP:WIND10:GRAP:COL 1;
      LAB:FONT BOLD"[2]
```
Select the bright pen and a bold font.

```
180   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 45,120;LAB 'HP 8711'"[2]
```
Draw a label reading 'HP 8711' at 45 pixels to the right and 120 pixels above the origin – the origin is the lower left corner of the current graphics window (the top half of the display).

```
190   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 30,175;DRAW 30,140;DRAW 480,140;
      DRAW 480,175"[2]
```
Draw a box to represent the HP 8711.

```
200   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 275,140;DRAW 275,130;DRAW 305,130;
      DRAW 305,140"[2]
```
Draw a box to represent the RF OUT port.

```
210   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 410,140;DRAW 410,130;DRAW 440,130;
      DRAW 440,140"[2]
```
Draw a box to represent the RF IN port.

```
220   OUTPUT @Hp8711;"DISP:WIND10:GRAP:LAB:
      FONT SMAL"[2]
```
Change the text font to small – this is the font the instrument uses to `DISP` or `PRINT` things.

```
230   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 250,145;LAB 'RF OUT'"[2]
```
Label the RF OUT port.

```
240   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 395,145;LAB 'RF IN'"[2]
```
Label the RF IN port.

ManualsPlus.com

```
250 Normalize: !
260   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 290,125;DRAW 290,110;DRAW 425,110;
      DRAW 425,125"[2]
```

Draw a THRU connection between the RF OUT and RF IN ports.

```
270   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 1,50;LAB 'Connect THRU between
      RF OUT and RF IN'"[2]
```

Prompt the operator to connect the THRU.

```
280   IF Internal=1 THEN
```

If using the internal IBASIC controller:

```
290     ON KEY 1 LABEL "NORMALIZE" RECOVER Norm
```

Use the simple `ON KEY` statement to handle the user interface.

```
300   ELSE
```

If using an external controller:

```
310     Keycode=-1
```

Set a flag for checking on keyboard interrupts.

```
320     OUTPUT @Hp8711;"DISP:MENU:KEY1
         'NORMALIZE'"
```

Label the softkey.

```
330     OUTPUT @Hp8711;"*CLS;*ESE 0"
```

Clear the event status registers.

```
340     OUTPUT @Hp8711;"STAT:PRES;DEV:ENAB 1;
        *SRE 4"
```

Preset the status registers, this effects the enable, positive and negative transition registers. Enable the Device Status register to report to the Status Byte on positive transition of bit 0 (Key Pressed). Enable the Status Byte to generate an interrupt when the Device Status register's summary bit reports in.

```
350     OUTPUT @Hp8711;"SYST:KEY:QUE:CLE"
```

Clear the key queue to make sure previous key presses don't generate an interrupt.

```
360     ON INTR Isc,5 RECOVER Srq
```

This branching must be done before the interrupt is enabled. It provides the path for the program to take when an interrupt occurs.

```
370     ENABLE INTR Isc;2
```

Enable an interrupt to occur when the service request is received.

```
380   END IF
390   OUTPUT @Hp8711;"DISP:WIND10:GRAP:BUFF OFF"[2]
```

Turn off the graphics buffer (this is not necessary for the program to run – the purpose is to avoid the error message when the buffer fills up during the following loop).

```
400   LOOP
```

Wait for a response – flash a message "Press Normalize" on the screen as a prompt. Note there is no exit from this loop other than a keyboard interrupt.

```
410     OUTPUT @Hp8711;"DISP:WIND10:GRAP:
        MOVE 55,18;LAB '>>>>>  Press NORMALIZE
          <<<<<'"[2]
```

ManualsPlus.com

```
420     WAIT .2
430     OUTPUT @Hp8711;"DISP:WIND10:GRAP:
        MOVE 55,18;LAB '       Press NORMALIZE
              '"[2]
440     WAIT .2
450   END LOOP
460 Norm: !
470   IF Keycode<>0 THEN GOTO Normalize
480   OFF KEY
490   OUTPUT @Hp8711;"DISP:WIND10:GRAP:BUFF ON"[2]
500   OUTPUT @Hp8711;"DISP:WIND10:GRAP:COL 0;
      MOVE 55,18;LAB '>>>>>  Press NORMALIZE
        <<<<<'"[2]
510   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 1,50;LAB 'Connect THRU between
      RF OUT and RF IN'"[2]
520   OUTPUT @Hp8711;"DISP:MENU:KEY1
        '          '"
530   OUTPUT @Hp8711;"CALC2:MATH (IMPL)"
540   OUTPUT @Hp8711;"INIT2;*WAI"
550   OUTPUT @Hp8711;"TRAC CH2SMEM,CH2SDATA;*WAI"
560   OUTPUT @Hp8711;"CALC2:MATH (IMPL/CH2SMEM)"
570   OUTPUT @Hp8711;"DISP:WIND2:TRAC1 ON;
      TRAC2 OFF"
580   OUTPUT @Hp8711;"DISP:WIND10:GRAP:
      MOVE 290,110;DRAW 425,110;DRAW 425,125;
      COL 1"[2]
590   STOP
600   !
610 Srq: !
620   Stb=SPOLL(@Hp8711)
630   IF BINAND(Stb,4)<>0 THEN
640     OUTPUT @Hp8711;"STAT:DEV:EVEN?"
650     ENTER @Hp8711;Dev_event
```

If the wrong key was pressed return to the previous message and loop.

The THRU should now be connected and ready to measure.

Turn the graphics buffer back on now that the loop is past.

Select the eraser (pen color 0) and erase the prompts.

Display the active data trace only – turn off any previous normalization.

Take a single sweep on channel 2.

Copy the new data trace into the memory array.

Normalize – display the active data relative to the memory trace.

Display only one of the traces (the normalized trace).

Erase the thru connection and select the bright pen (color 1).

This is the end.

This is the branching routine that handles service request generated interrupts.

Do a serial poll to find out if the analyzer generated the interrupt.

Determine if the Device Status register's summary bit (bit 2 of the Status Byte) has reported in.

If it has, read the Device Status event register.

ManualsPlus.com

```
660     IF BINAND(Dev_event,1)<>0 THEN
670       OUTPUT @Hp8711;"SYST:KEY?"
680       ENTER @Hp8711;Keycode
690     END IF
700   END IF
710   ENABLE INTR Isc
720   GOTO Norm
730   END
```

Determine if the event was a key press (bit 0 of the Device Status register).

If yes, query the analyzer what key has been pressed.

Re-enable the interrupt in cast the wrong key was pressed.

The thru is now connected – return to that part of the routine and do it.

This is really the end.

1 This command is only available on instruments that have the IBASIC (1C2) option.

2 Window 10 is the IBASIC display window. Instruments without the IBASIC option (1C2) cannot draw to this window. The other graphics windows (1 – channel 1, and 2 – channel 2) are available on all instruments. Turning off the data trace and graticule on a measurement channel that is not being used results in a clearly visible graphics viewing area.

ManualsPlus.com

CALKIT — Instrument state file for downloading user-defined calibration kit definitions.

This instrument state file demonstrates the type of file required to download user-defined calibration kits. To see an example of using this feature, refer to "To Calibrate with Other Cal Kits" in chapter 6 of the *HP 8711 Operating and Programming Manual*.

```
10  !$  Standard Definitions for HP 85054B Precision Type-N Cal Kit.
20  !
30  !$  Definitions for 50 Ohm jack (FEMALE center contact) test
40  !$  ports, plug (MALE center contact) standards.
50  !
60  ! OPEN:  $ HP 85054-60027 Open Circuit Plug
70  !    Z0  50.0 $ Ohms
80  !    DELAY  57.993E-12 $ Sec
90  !    LOSS  0.8E+9 $ Ohms/Sec
100 !    C0  88.308E-15 $ Farads
110 !    C1  1667.2E-27 $ Farads/Hz
120 !    C2 -146.61E-36 $ Farads/Hz^2
130 !    C3  9.7531E-45 $ Farads/Hz^3
140 !
150 ! SHORT:  $ HP 85054-60025 Short Circuit Plug
160 !    Z0  50.0 $ Ohms
170 !    DELAY  63.078E-12 $ Sec
180 !    LOSS  8.E+8 $ Ohms/Sec
190 !
200 ! LOAD:  $ HP 00909-60011 Broadband Load Plug
210 !    Z0  50.0 $ Ohms
220 !    DELAY  0.0 $ Sec
230 !    LOSS   0.0 $ Ohms/Sec
240 !
250 ! THRU:  $ HP 85054-60038 Plug to Plug Adapter
260 !    Z0  50.0 $ Ohms
270 !    DELAY  196.0E-12 $ Sec
280 !    LOSS   2.2E+9 $ Ohms/Sec
290 !
300 END
```