

# Notice

---

## Hewlett-Packard to Agilent Technologies Transition

This manual may contain references to HP or Hewlett-Packard. Please note that Hewlett-Packard's former test and measurement, semiconductor products and chemical analysis businesses are now part of Agilent Technologies. To reduce potential confusion, the only change to product numbers and names has been in the company name prefix: where a product name/number was HP XXXX the current name/number is now Agilent XXXX. For example, model number HP 8648 is now model number Agilent 8648.

## Contacting Agilent Sales and Service Offices

The sales and service contact information in this manual may be out of date. The latest service and contact information for your location can be found on the Web at:

<http://www.agilent.com/find/assist>

If you do not have access to the Internet, contact your field engineer. In any correspondence or telephone conversation, refer to your instrument by its model number and full serial number.



# Programming Guide

## HP ESG Series Signal Generators

### Serial Number Prefixes:

HP ESG 1000A, US3704

HP ESG 2000A, US3704

HP ESG 3000A, US3704

HP ESG 4000A, US3704



**HP Part No. E4400-90077**

**Printed in USA**

**Print Date: March 1997**

©Copyright Hewlett-Packard Company 1997. All Rights Reserved. Reproduction, adaptation, or translation without prior written permission is prohibited, except as allowed under the copyright laws.

1400 Fountaingrove Parkway, Santa Rosa, CA 95403-1799, USA



---

# Contents

## 1 Preparing for Use

Setting up the Equipment for Remote Operation .....	1-2
Verifying Remote Programming Functionality .....	1-5

## 2 Programming Fundamentals

Programming the Signal Generator .....	2-2
Data Transfers Between Computer and Peripheral.....	2-3
Overview of Serial Interface (RS-232) Programming.....	2-4
Transferring Data .....	2-8
HP-IB Instrument Nomenclature .....	2-11
HP-IB Command Statements.....	2-12
Getting Started with SCPI.....	2-19
Programming the Status Register System.....	2-37
Advanced Programming Information .....	2-65

## 3 Programming Examples

Using the Example Programs .....	3-2
HP-IB Check, Example Program 1 .....	3-3
Local Lockout Demonstration, Example Program 2 .....	3-5
Using Queries, Example Program 3 .....	3-8
Generating a CW Signal, Example Program 4 .....	3-11
Generating an AC-Coupled External FM Signal, Example Program 5.....	3-13
Generating an AC-Coupled Internal FM Signal, Example Program 6.....	3-15

---

## Contents

Generating a Step-Swept Signal, Example Program 7 .....	3-17
Generating an External DC-Coupled Pulse Modulated Signal, Example Program 8 .....	3-20
Saving and Recalling States, Example Program 9 .....	3-22
Reading the Status Byte, Example Program 10.....	3-26
End of Sweep Service Request, Example Program 11 .....	3-30
<b>4 Programming Command Cross Reference</b>	
Front Panel Key Versus Command.....	4-2
HP 8656/57-Compatible Language .....	4-25
<b>5 Language Reference</b>	
Command Syntax .....	5-2
IEEE 488.2 Common Commands .....	5-3
Subsystem Commands.....	5-6
:AM Subsystem .....	5-7
:CALibration Subsystem .....	5-16
:COMMunicate Subsystem .....	5-17
:DIAGnostic Subsystem .....	5-21
:DISPlay Subsystem .....	5-23
:FM Subsystem.....	5-25
:FREQuency Subsystem.....	5-33
:LFOutput Subsystem.....	5-41
:LIST Subsystem .....	5-47

---

## Contents

:MEMory and :MMEMory Subsystems .....	5-53
:OUTPut Subsystem .....	5-61
:PM Subsystem .....	5-64
:POWer Subsystem .....	5-71
:PULM Subsystem.....	5-77
:STATus Subsystem .....	5-81
:SWEep Subsystem.....	5-91
:SYSTem Subsystem.....	5-93
:TRIGger Subsystem .....	5-98
<b>6 Error Messages</b>	
Error Messages .....	6-2
Querying the Error Queue .....	6-4
Error Numbers .....	6-5
No Error .....	6-6
SCPI Standard Error Messages.....	6-7
Command Error .....	6-8
Execution Error.....	6-13
Device-Specific Error .....	6-20
Query Error.....	6-22
ESG Series Signal Generator Instrument-Specific Error Messages .....	6-24

---

## Contents

---

# 1 Preparing for Use

---

This chapter explains how to set up the equipment for remote programming of the signal generator, what the signal generator's HP-IB capabilities are, and provides a program for an operational check of remote programming functionality.



## Setting up the Equipment for Remote Operation

### HP-IB Overview

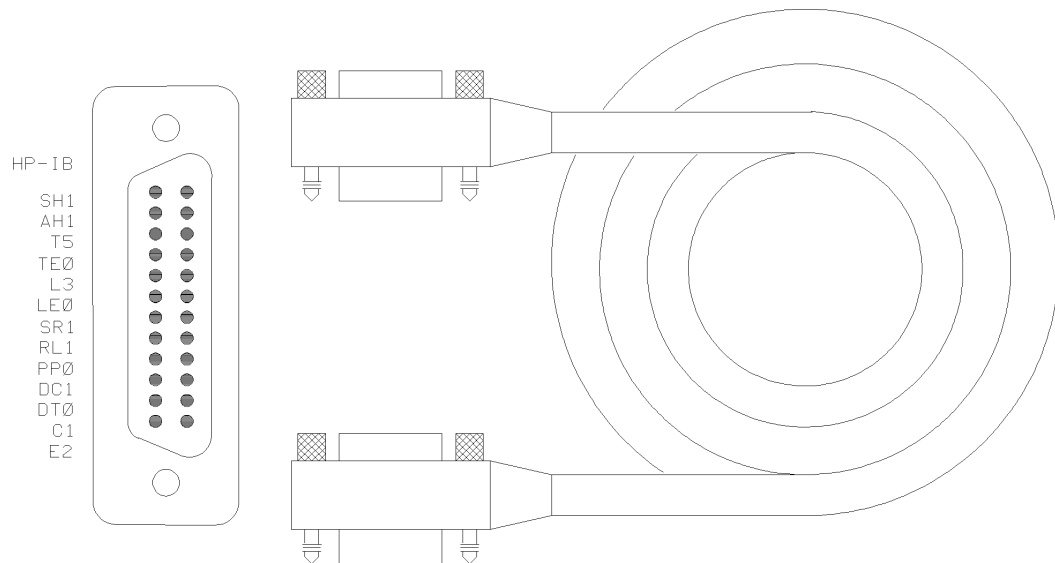
HP-IB, the Hewlett-Packard Interface Bus, is a high-performance bus that allows individual instruments and computers to be combined into integrated test systems. The bus and its associated interface operations are defined by the IEEE 488.1 standard. The IEEE 488.2 standard defines the interface capabilities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

Commands are sent over the HP-IB via a controller's language system. HP BASIC is the language used in the programming examples in this book. BASIC was selected because the majority of HP-IB computers have BASIC language capability. However, other languages can also be used. The use of BASIC is explained in more detail in Chapter 2.

### Interconnecting Cables

The HP-IB connector allows the signal generator to be connected to any other instrument or device on the interface bus. All HP-IB instruments can be connected with HP-IB cables. An example of an HP-IB cable is shown in Figure 1-1. As many as 14 instruments can be connected to the signal generator via HP-IB (15 total instruments in the system). The cables can be interconnected in a star pattern (one central instrument, with the HP-IB cables emanating from that instrument like spokes on a wheel), or in a linear pattern (like boxcars on a train), or any combination pattern. The following restrictions apply:

- Each instrument must have a unique HP-IB address ranging from 0 through 30 (decimal). See "Instrument Addresses," in this section, for more information.
- In a two-instrument system that uses just one HP-IB cable, the cable length must not exceed 4 meters (9.13 ft.) between the two instruments.
- When more than two instruments are connected on the bus, the cable length between each instrument must not exceed 2 meters (6.5 ft.) per unit.
- The total cable length between all instruments must not exceed 20 meters (65 ft.).



**Figure 1-1. HP-IB Connector and Cable**

The codes next to the HP-IB connector, shown in Figure 1-1, describe the HP-IB electrical capabilities of the signal generator, using IEEE Std. 488-1978 mnemonics (HP-IB, GP-IB, IEEE-488, and IEC-625 are all electrically equivalent, though IEC-625 uses a unique connector). Briefly, the mnemonics translate as follows:

<b>SH1</b>	Source Handshake, complete capability.
<b>AH1</b>	Acceptor Handshake, complete capability.
<b>T6</b>	Talker; capable of basic talker, serial poll, and unaddress if MLA.
<b>TE0</b>	Talker, Extended address; no capability.
<b>L3</b>	Listener, capable of basic listener, and unaddress if MTA.
<b>LE0</b>	Listener, Extended address; no capability.
<b>SR1</b>	Service Request, complete capability.
<b>RL1</b>	Remote Local, complete capability.
<b>PP0</b>	Parallel Poll, no capability.
<b>DC1</b>	Device Clear, complete capability.
<b>DT1</b>	Device Trigger, complete capability.
<b>C0, 1, 2, 3, 28</b>	Controller capability options; C0, no capabilities; C1, system controller; C2, send IFC and take charge; C3, send REN; C28, send I.F. messages.
<b>E1</b>	Electrical specification indicating open collector outputs.

## Instrument Addresses

Each instrument in an HP-IB network must have a unique address, ranging in value from 00 through 30 (decimal). The default address for the signal generator is 19, but this can be changed by pressing **Utility**, **HP-IB/RS-232**, **HP-IB Address**, entering the new address with the numeric keypad and pressing **Enter**.

---

**NOTE:** Programming examples in this book assume that the signal generator's HP-IB address is 19. Modify the examples, if necessary, to correspond to your signal generator's address.

---

## Set Up For Remote Operation of the Signal Generator

The following procedure describes how to connect your equipment for remote operation of the signal generator.

1. Connect a computer and any other peripherals to the signal generator with HP-IB cables.
2. Reset all instruments connected to the bus. If you are not sure how to reset a device, switch off the line power to the device and then switch the power back on.
3. Check the HP-IB address of the signal generator by pressing **Utility**, **HP-IB/RS-232**.

---

## Verifying Remote Programming Functionality

This program verifies that the HP-IB connections and interface are functional. With the equipment set up as described in the previous section, clear and reset the controller. Type in the following program:

```

10   Sig_gen=719
20   ABORT 7
30   LOCAL Sig_gen
40   CLEAR Sig_gen
50   OUTPUT Sig_gen;"*RST"
60   REMOTE Sig_gen
70   CLEAR SCREEN
80   PRINT "The source should now be in REMOTE."
90   PRINT "Verify that the 'R' annunciator is displayed"
100  END

```

Run the program and verify that the **R** (remote) annunciator is activated on the signal generator's display. If it is not, verify that the signal generator address is set to 19 (see the section titled "Instrument Addresses") and that the interface cable is properly connected.

If the controller display indicates an error message, it is possible that the program was entered in incorrectly. If the controller accepts the remote statement but the signal generator's remote annunciator does not appear on the signal generator's display, refer to the service guide for troubleshooting information.

### Program Comments

10:	Set up a variable to contain the HP-IB address of the signal generator.
20:	Abort any bus activity and return the HP-IB interfaces to their reset states.
30:	Place the signal generator into LOCAL mode to cancel any local lockouts that may have been set up.
40 to 50:	Reset the signal generator's parser and clear any pending output from the signal generator. Prepare the signal generator to receive new commands.
60:	Place the signal generator into remote mode.
70:	Clear the controller's display.
80 to 90:	Print a message to the controller's display.
100:	End the program.

Preparing for Use  
Verifying Remote Programming Functionality

---

## 2 Programming Fundamentals

---

This chapter explains how to program the signal generator using HP-IB command statements and the SCPI language.

## **Programming the Signal Generator**

The signal generator can be controlled entirely by a computer (although the line power switch must be operated manually). Several functions are possible only by remote control. Computer programming procedures for the signal generator involve selecting an HP-IB command statement and then adding the specific programming codes (SCPI or HP 8656/67-compatible) to that statement to achieve the desired operating conditions. The programming codes can be categorized into two groups: those that mimic front panel keystrokes, and those that are unique and have no front panel equivalent.

In the programming explanations that follow, specific examples are included that are written in a generic dialect of the HP BASIC language. HP BASIC was selected because the majority of HP-IB computers have HP BASIC language capability. However, other languages can be used as well.

---

## Data Transfers Between Computer and Peripheral

Five statements are used to transfer information between your desktop computer and the interface card:

- The OUTPUT statement sends data to the interface which, in turn, sends the information to the peripheral device.
- The ENTER statement inputs data from the interface card after the interface has received it from the peripheral device.
- The STATUS statement is used to monitor the interface and obtain information about interface operation such as buffer status, detected errors, and interrupt enable status.
- The CONTROL statement is used to control interface operation and defines such parameters as baud rate, character format, or parity.
- The TRANSFER statement is used to input or output data from/to the interface and, in turn, from/to the peripheral device.

Since the interface has no on-board processor, ENTER and OUTPUT statements cause the computer to wait until the ENTER or OUTPUT operation is complete before continuing to the next line. For OUTPUT statements, this means that the computer waits until the last bit of the last character has been sent over the serial line before continuing with the next program statement.



## Overview of Serial Interface (RS-232) Programming

Serial interface programming techniques are similar to most general I/O applications. The interface card is initialized by use of CONTROL statements; STATUS statements evaluate its readiness for use. Data is transferred between the desktop computer and a peripheral device by OUTPUT and ENTER statements.

Due to the asynchronous nature of serial I/O operations, special care must be exercised to ensure that data is not lost by sending to another device before the device is ready to receive. Modem line handshaking can be used to help solve this problem. These and other topics are discussed in greater detail elsewhere in this chapter.

### Determining Operating Parameters

Before you can successfully transfer information to a device, you must match the operating characteristics of the interface to the corresponding characteristics of the peripheral device. This includes matching signal lines and their functions as well as matching the character format for both devices.

#### Handshake and Baud Rate

To determine hardware operating parameters, you need to know the answer for each of the following questions about the peripheral device:

- Which of the following signal and control lines are actively used during communication with the peripheral?
  - Data Set Ready (DSR)
  - Clear to Send (CTS)
- What baud rate (line speed) is expected by the peripheral?

#### Character Format Parameters

To define the character format, you must know the requirements of the peripheral device for the following parameters:

- Character Length: Eight data bits are used for each character, excluding start, stop, and parity bits.
- Parity Enable: Parity is disabled (absent) for each character.
- Stop Bits: One stop bit is included with each character.

## Serial Configuration for BASIC/UX

There is no capability in BASIC/UX for reading the hardware bit settings on either the HP 98626 or HP 98644 Serial Interface cards. Therefore, BASIC/UX provides two methods for configuring modem control options:

- The **stty** command from the HP-UX environment.
- The keyword CONTROL and registers directly related to the modem control options.

Of the two methods mentioned above, the best one to use while in the HP-UX environment is the **stty** command. The reason for this is any modem control options set by using the keyword CONTROL are lost when you leave BASIC/UX. However, if you prefer to change these options while in the BASIC/UX environment, then read the subsequent section “Using Program Control to Override Defaults.”

This section deals with the first method mentioned above which is the use of the **stty** command from the HP-UX environment.

### Defaults for the Serial Interface

When HP-UX is being booted up, the defaults for all serial interfaces are:

<b>Baud Rate</b>	300
<b>Bits per character</b>	8
<b>Parity</b>	Off
<b>Stop bits</b>	1

The above values are used by BASIC/UX as defaults, unless configured as explained in the next section.

Some common serial interface configuration settings are:

<b>Baud Rate to</b>	9600
<b>Bits per character to</b>	8
<b>Parity to</b>	Odd and disabled
<b>Stop bits to</b>	1

### Configuring a Serial Interface for BASIC/UX

To configure your serial interface with the values mentioned in the previous section, you can execute the following HP-UX command before entering BASIC/UX:

```
/bin/stty 9600 cs8 -cstopb < /dev/rmb/serialnn
```

where:

**9600** is the baud rate. The following are baud rates you can use with the **stty** command:

300          1200          2400          4800          9600          19 200

**cs8** is the number of bits per character. For this signal generator, the number of bits per character is 8.

**-cstopb** causes one stop bit per character to be used.

**< /dev/rmb/serialnn** assigns the stty options to the serial interface located at select code number nn.

For more information on stty options, see the HP-UX Language Reference.

## Using Program Control to Override Defaults

You can override some of the interface default configuration options by use of CONTROL statements. This not only enables you to guarantee certain parameters, but also provides a means for changing selected parameters in the course of a running program.

### Interface Reset

Whenever an interface is connected to a modem that may still be connected to a telecommunications link from a previous session, it is good programming practice to reset the interface to force the modem to disconnect, unless the status of the link and remote connection are known. When the interface is connected to a line printer or similar peripheral, resetting the interface is usually unnecessary unless an error condition requires it.

```
100 CONTROL Sc,0;1          ! Resets Interface.
```

When the interface is reset by use of a CONTROL statement to CONTROL Register 0 with a non-zero value, the interface is restored to the BASIC/UX power-up condition whether or not it is the same as the current default switch configuration. If you are not sure of the present settings, or if your application requires changing the configuration during program operation, you can use CONTROL statements to configure the interface. An example of where this may be necessary is when several peripherals share a single interface through a manually operated RS-232 switch such as those used to connect multiple terminals to a single computer port, or a single terminal to multiple computers.

### Selecting the Baud Rate

In order to successfully transfer information between the interface card and a peripheral, the interface and peripheral must be set to the same baud rate. A CONTROL statement to register 3 (or 13 with 98644 interfaces) can be used to set the interface baud rate to any one of the following values:

300      1200      2400      4800      9600      19 200

For example, to select a baud rate of 9600, the following program statement is used:

```
1190 CONTROL Sc,3;9600
```

Use of values other than those shown may result in incorrect operation.

To verify the current baud rate setting, use a STATUS statement addressed to register 3. All rates are in baud (bits/second).

**Setting Character Format and Parity**

CONTROL Register 4 overrides the Line Control switches<sup>1</sup> that control parity and character format. To determine the value sent to the register, add the appropriate values selected from the following table:

Table 2-1.                      Character Format and Parity Settings Handshake

Handshake (Bits <sup>1</sup> 7 & 6)	Parity Enable (Bit 3)	Stop Bits (Bit 2)	Character Length (Bits 1 & 0)
01 Xon/Xoff Bidirectional  11 Handshake Disabled	0 Disabled	0 One stop bit	11 Eight bits/char

1. All bits in this table correspond to equivalent switch settings on the HP 98626 and HP 98644 serial interface cards. A 1 is the same as set.

For example, to configure a character format of 8 bits per character, one stop bit, and disabled parity, with XON/XOFF; use the following CONTROL statement:

```
1200 CONTROL Sc,4;IVAL("10011",2)
-or-
1200 CONTROL Sc,4;19
```

1. With HP 98644 interfaces, there are no Line Control switches. You can simulate their effect by writing to CONTROL register 14. Note that individual bits of this register are the same as for register 4.

## Transferring Data

The serial interface card is designed for relatively simple serial I/O operations. It is not intended for sophisticated applications that use ON INTR statements to service the interface.

### Entering and Outputting Data

When the interface is properly configured, either by use of default switches or CONTROL statements, you are ready to begin data transfers. OUTPUT statements are used to send information to the peripheral; ENTER statements to input information from the external device.

```
OUTPUT 20;"String data",Numeric_var,Etc
```

```
ENTER 20;String_var$,Numeric_var,Etc
```

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

Various other I/O statements can be used in addition to OUTPUT and ENTER, depending on the situation. For example, the LIST statement can be used to list programs to an RS-232 line printer -- provided the interface is properly configured before the operation begins.

### Outputting Data

To send data to a peripheral, use OUTPUT, OUTPUT USING, or any other similar or equivalent construct. Suppression of end-of-line delimiters and other formatting capabilities are identical to normal operation in general I/O applications. The OUTPUT statement hangs the computer until the last bit of the last character in the statement variable list is transmitted by the interface. When the output operation is complete, the computer then continues to the next line in the program.

### Entering Data

To input data from a peripheral, use ENTER, ENTER USING, or an equivalent statement. Inclusion or elimination of end-of-line delimiters and other information is determined by the formatting specified in the ENTER statement. The ENTER statement hangs the computer until the input variables list is satisfied. To minimize the risk of waiting for another variable that isn't coming, you may prefer to specify only one variable for each ENTER statement, and analyze the result before starting the next input operation.

Be sure that the peripheral is not transmitting data to the interface while no ENTER is in progress. Otherwise, data may be lost because the card provides buffering for only one character. Also, interrupts from other I/O devices, or operator inputs to the computer keyboard can cause delay in computer service to the interface that result in buffer overrun at higher baud rates.

## Modem Line Handshaking

Modem line handshaking, when used, is performed automatically by the computer as part of the OUTPUT or ENTER operation. If the modem line states have not been latched in a fixed state by Control Register, the following sequence of events is executed automatically during each OUTPUT or ENTER operation:

For OUTPUT operations:

1. Set Data Terminal Ready and Request-to-Send modem lines to active state.
2. Check Data Set Ready and Clear-to-Send modem lines to be sure they are active.
3. Send information to the interface and thence to the peripheral.
4. After data transfer is complete, clear Data Terminal Ready and Request-to-Send signals.

For ENTER operations:

1. Set Data Terminal Ready line to active state. Leave Request-to-Send inactive.
2. Check Data Set Ready and Data Carrier Detect modem lines to be sure they are active.
3. Input information from the interface as it is received from the peripheral.
4. After the input operation is complete, clear the Data Terminal Ready signal.

After a given OUTPUT or ENTER operation is completed, the program continues execution on the next line.

Control Register 5 can be used to force selected modem control lines to their active states. The Data Rate Select line is set or cleared by bit 2. Request-to-send and Data Terminal Ready are held in their active states when bits 1 and 0 are true, respectively. If bits 1 or 0 are false, the corresponding modem line is toggled during OUTPUT or ENTER as explained previously.

## Incoming Data Error Detection and Handling (BASIC/WS only)

The serial interface card can generate several errors that are caused when certain conditions are encountered while receiving data from the peripheral device. The UART detects a given error condition. The card then generates a pending error to BASIC. Errors can be generated by any of the following conditions:

- Parity error. The parity bit on an incoming character does not match the parity expected by the receiver. This condition is most commonly caused by line noise.
- Framing error. Start and stop bits do not match the timing expectations of the receiver. This can occur when line noise causes the receiver to miss the start bit or obscures the stop bits.
- Overrun error. Incoming data buffer overrun caused a loss of one or more data characters. This is usually caused when data is received by the interface, but no ENTER statement has been activated to input the information.
- Break received. A BREAK was sent to the interface by the peripheral device. The desktop computer program must be able to properly interpret the meaning of a break and take appropriate action.

## Programming Fundamentals

### Transferring Data

All UART status errors are generated by incoming data, never by outbound data. When a UART error occurs, the corresponding bit of Status Register 10 is set, and a pending error (ERROR 167: Interface status error) is sent to BASIC. BASIC processes the error according to the following rules:

- If an ENTER is in progress, the error is handled immediately as part of the ENTER process. An active ON ERROR causes the error trap to be executed. If no ON ERROR is active, the error is fatal and causes the program to terminate.
- If an OUTPUT is in progress, or if there is no current activity between the computer and interface, the error is flagged, but nothing is done by BASIC until an ENTER statement is encountered. When the computer begins execution of the ENTER statement, if an ON ERROR is active, the error trap is executed. If there is no active ON ERROR for that select code, the fatal ERROR 167 causes the BASIC program to terminate.
- If a STATUS statement is executed to Status Register 10 before an ENTER statement is encountered for that select code, the pending BASIC error is cleared, and the program continues as if no error had been generated.

Note that the above UART status errors cannot be detected using BASIC/UX.

---

## HP-IB Instrument Nomenclature

An instrument that is part of an HP-IB network is categorized as a listener, talker, or controller, depending on its current function in the network.

- Listener**                      A listener is a device capable of receiving data or commands from other instruments. Any number of instruments in the HP-IB network can be listeners simultaneously.
- Talker**                         A talker is a device capable of transmitting data or commands to other instruments. To avoid confusion, an HP-IB system allows only one device at a time to be an active talker.
- Controller**                    A controller is an instrument, typically a computer, capable of managing the various HP-IB activities. Only one device at a time can be an active controller.



---

## HP-IB Command Statements

Command statements form the nucleus of HP-IB programming; they are understood by all instruments in the network. When combined with the programming language codes, they provide all management and data communication instructions for the system.

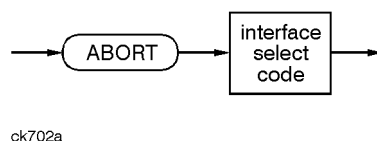
An explanation of the fundamental command statements follows. Some computers may use a slightly different terminology, or support an extended or enhanced version of these commands. Use the following explanations as a starting point and refer to the HP BASIC language reference manual, the I/O programming guide, and the HP-IB manual for the computer you are using.

Syntax drawings accompany each statement. All items enclosed by a circle or oval are computer-specific terms that must be entered exactly as described; items enclosed in a rectangular box are names of parameters used in the statement; and the arrows indicate a path that generates a valid combination of statement elements.

The seven fundamental command statements are as follows:

### Abort

ABORT abruptly terminates all listener/talker activity on the interface bus, and prepares all instruments to receive a new command from the controller. Typically, this is an initialization command used to place the bus in a known starting condition. The syntax is:



**Figure 2-1. Abort Command Syntax**

where the interface select code is the computer's HP-IB I/O port, which is typically port 7.

### A BASIC Example

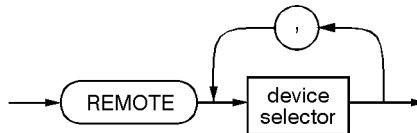
```
10 ABORT 7
100 IF V>20 THEN ABORT 7
```

### Related Statements Used by Some Computers

- ABORTIO (used by HP-80 series computers)
- HALT
- RESET

## Remote

REMOTE causes an instrument to change from local control to remote control. In remote control, the front panel keys are disabled except for the **Local** key and the line power switch. The syntax is:



ck703a

**Figure 2-2. Remote Command Syntax**

where the device selector is the address of the instrument appended to the HP-IB port number. Typically, the HP-IB port number is 7 and the default address for the signal generator is 19, so the device selector is 719.

### Some BASIC Examples

```
10 REMOTE 7
```

which prepares all HP-IB instruments for remote operation (although nothing appears to happen to the instruments until they are addressed to talk), or

```
10 REMOTE 719
```

which affects the HP-IB instrument located at address 19, or

```
10 REMOTE 719, 721, 726, 715
```

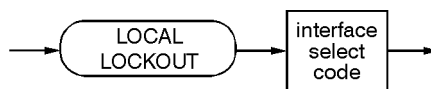
which affects four instruments that have addresses 19, 21, 26, and 15.

### Related Statements Used by Some Computers

- RESUME

## Local Lockout

LOCAL LOCKOUT can be used with REMOTE to disable the front panel **Local** key. With the **Local** key disabled, only the controller (or a hard reset by the line power switch) can restore local control. The syntax is:



ck704a

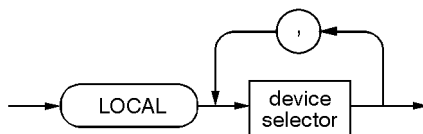
Figure 2-3. Local Lockout Command Syntax

## A BASIC Example

```
10 REMOTE 719
20 LOCAL LOCKOUT 7
```

## Local

LOCAL is the complement to REMOTE, causing an instrument to return to local control with a fully enabled front panel. The syntax is:



ck705a

Figure 2-4. Local Command Syntax

## Some BASIC Examples

```
10 LOCAL 7
```

which affects all instruments in the network, or

```
10 LOCAL 719
```

for an addressed instrument (address 19).

## Related Statements Used by Some Computers

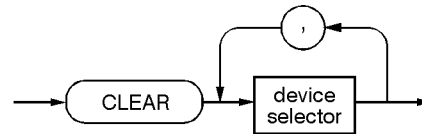
- RESUME

## Clear

CLEAR causes all HP-IB instruments, or addressed instruments, to assume a cleared condition. The definition of clear is unique for each instrument. For the signal generator:

1. All pending output-parameter operations are halted.
2. The parser (the software that interprets the programming codes) is reset and now expects to receive the first character of a programming code.
3. Any sweep in progress is aborted and continuous sweep is turned off.
4. Any I/Q calibration in progress will be aborted.

The syntax is:



ck706a

**Figure 2-5. Clear Command Syntax**

### Some BASIC Examples

```
10 CLEAR 7
```

to clear all HP-IB instruments, or

```
10 CLEAR 719
```

to clear an addressed instrument (address 19)

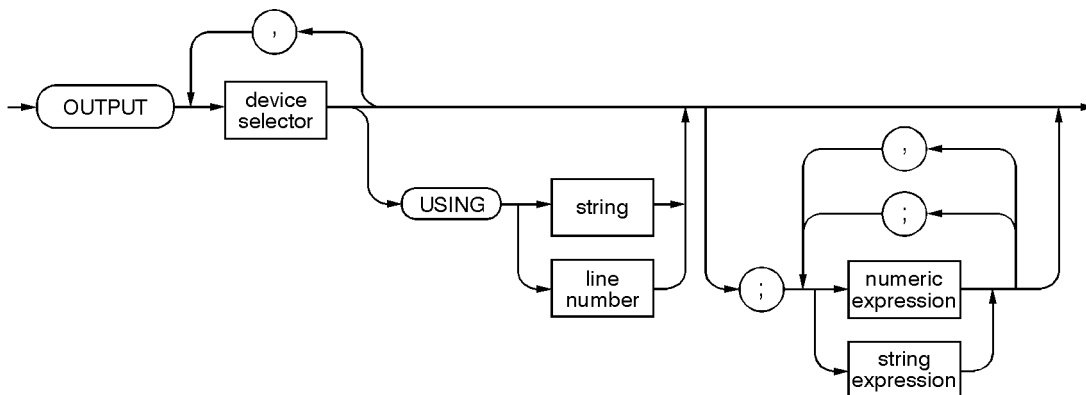
### Related Statements Used by Some Computers

- RESET
- CONTROL
- SEND

The preceding statements are primarily management commands that do not incorporate programming codes. The following two statements do incorporate programming codes and are used for data communication.

## Output

OUTPUT is used to send function commands and data commands from the controller to the addressed instrument. The syntax is:



ck707a

Figure 2-6. Output Command Syntax

where USING is a secondary command that formats the output in a particular way, such as a binary or ASCII representation of numbers. The USING command is followed by image items that precisely define the format of the output; these image items can be a string of code characters or a reference to a statement line in the program. Image items are explained in the programming codes where they are needed. Notice that this syntax is virtually identical to the syntax for the ENTER statement that follows.

### A BASIC Example

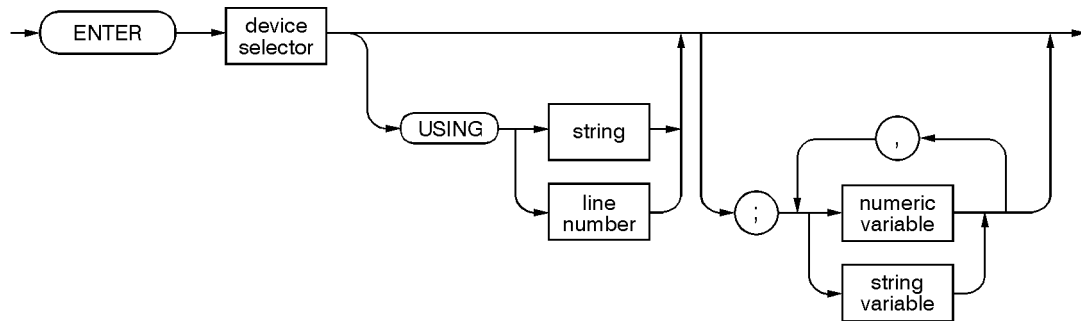
```
100 OUTPUT 719; "programming codes"
```

### Related Statements Used by Some Computers

- CONTROL
- CONVERT
- IMAGE
- IOBUFFER
- TRANSFER

## Enter

ENTER is the complement of OUTPUT and is used to transfer data from the addressed instrument to the controller. The syntax is:



ck708a

Figure 2-7. Enter Command Syntax

ENTER is nearly always used in conjunction with OUTPUT.

### Some BASIC Examples

```

100 OUTPUT 719, "...programming codes..."
110 ENTER 719; "...response data..."
  
```

ENTER statements are commonly formatted, which requires the secondary command USING and the appropriate image items. The most-used image items involve end-of-line (EOI) suppression, binary inputs, and literal inputs. For example:

```
100 ENTER 719 USING "#, B"; A, B, C
```

suppresses the EOI sequence (#), and indicates that variables A, B, and C are to be filled with binary (B) data. In another example:

```
100 ENTER 719 USING "#, 123A"; A$
```

suppresses EOI and indicates that string variable A\$ is to be filled with 123 bytes of literal data (123A).

---

#### NOTE:

Be careful when using byte-counting image specifiers. If the requested number of bytes does not match the actual number available, data might be lost or the program might enter an endless wait state.

---

The suppression of the EOI sequence is frequently necessary to prevent a premature termination of the data input. When not specified, the typical EOI termination occurs when an ASCII LF (line feed) is received. However the LF bit pattern could coincidentally occur randomly in a long string of binary data, where it might cause a false termination. Also the bit patterns for the ASCII CR (carriage return), comma, or semicolon might cause a false termination. Suppression of the EOI causes the computer to accept all bit patterns as data, not commands, and relies on the HP-IB EOI (end or identify) line for correct end-of-data termination.

**Related Statements Used by Some Computers**

- CONVERT
- IMAGE
- IOBUFFER
- ON TIMEOUT
- SET TIMEOUT
- TRANSFER

---

## Getting Started with SCPI

This section describes the use of the Standard Commands for Programmable Instruments language (SCPI). This section explains how to use SCPI commands in general. For a list of the specific SCPI commands available in the signal generator, refer to Chapters 4 and 5.

### Understanding Common Terms

The following terms are used throughout the remainder of this chapter.

<b>Controller</b>	A controller is any computer used to communicate with a SCPI instrument. A controller can be a personal computer, a minicomputer, or a plug-in card in a card cage. Some intelligent instruments can also function as controllers.
<b>Instrument</b>	An instrument is any device that implements SCPI. Most instruments are electronic measurement or stimulus devices, but this is not a requirement. Similarly, most instruments use an HP-IB or RS-232 interface for communication. The same concepts apply regardless of the instrument function or the type of interface used.
<b>Program Message</b>	A program message is a combination of one or more properly formatted SCPI commands. Program messages always go from a controller to an instrument. Program messages tell the instrument how to make measurements and output signals.
<b>Response Message</b>	A response message is a collection of data in specific SCPI formats. Response messages always go from an instrument to a controller or listening instrument. Response messages tell the controller about the internal state of the instrument and about measured values.
<b>Command</b>	A command is an instruction in SCPI. You combine commands to form messages that control instruments. In general, a command consists of mnemonics (keywords), parameters, and punctuation.
<b>Query</b>	A query is a special type of command. Queries instruct the instrument to make response data available to the controller. Query mnemonics always end with a question mark.



## Standard Notation

This section uses several forms of notation that have specific meaning:

**Command Mnemonics** Many commands have both a long and a short form and you must use either one or the other (SCPI does not accept a combination of the two.) Consider the **FREQuency** command, for example. The short form is **FREQ** and the long form is **FREQUENCY**. This notation type is a shorthand to document both the long and short form of commands. SCPI is not case sensitive, so **fREquEnCy** is just as valid as **FREQUENCY**, but **FREQ** and **FREQUENCY** are the only valid forms of the **FREQuency** command.

**Angle Brackets** Angle brackets indicate that the word or words enclosed represent something other than themselves. For example, **<new line>** represents the ASCII character with the decimal value 10. Similarly, **<^END>** means that EOI is asserted on the HP-IB interface. Words in angle brackets have much more rigidly defined meaning than words shown in ordinary text. For example, this section uses the word “message” to talk about messages generally. But the bracketed words **<program message>** indicate a precisely defined element of SCPI. If you need them, you can find the exact definitions of words such as **<program message>** in a syntax diagram.

## How to Use Examples

Programming with SCPI requires knowledge of two languages. You must know the programming language of your controller (BASIC, C, Pascal) as well as the language of your instrument (SCPI). The semantic requirements of your controller’s language determine how the SCPI commands and responses are handled in your application.

### Command Examples

Command examples look like this:

```
:FREQuency: CW?
```

This example tells you to put the string **:FREQuency: CW?** in the output statement appropriate to your application programming language. If you encounter problems, study the details of how the output statement handles message terminators such as **<new line>**. If you are using simple OUTPUT statements in HP BASIC, this is taken care of for you. In HP BASIC, you type:

```
OUTPUT 719 " :FREQuency: CW? "
```

Command examples do not show message terminators because they are used at the end of every program message. “Details of Commands and Responses” discusses message terminators in more detail.

## Response Examples

Response examples look like this:

```
3.000000000000E+009
```

These are the characters you would read from an instrument after sending a query command. To actually pull them from the instrument into the controller, use the input statement appropriate to your application programming language. If you have problems, study the details of how the input statement operates. In particular, investigate how the input statement handles punctuation characters such as the comma and the semicolon and how it handles **<new line>** and EOI. To enter the previous response in HP BASIC you type:

```
ENTER 719;CW_frequency
```

Response examples do not show response message terminators because they are always **<new line>** **<^END>**. These terminators are typically automatically handled by the input statement. See “Details of Commands and Responses,” later in this chapter, for more information about terminators.

## Program and Response Messages

To understand how your instrument and controller communicate using SCPI, you must understand the concepts of program and response messages. Program messages are the formatted data sent from the controller to the instrument. Conversely, response messages are the formatted data sent from the instrument to the controller. Program messages contain one or more commands, and response messages contain one or more responses.

The controller may send commands at any time, but the instrument sends responses only when specifically instructed to do so. The special type of command used to instruct the instrument to send a response message is the query. All query mnemonics end with a question mark. Queries return either measured values or internal instrument settings. Any internal setting that can be programmed with SCPI can also be queried.

### Forgiving Listening and Precise Talking

SCPI uses the concept of forgiving listening and precise talking outlined in IEEE 488.2.

Forgiving listening means that instruments are very flexible in accepting various command and parameter formats. For example, the signal generator accepts either **:POWER:ALC[:STATe] ON** or **:POWER:ALC[STATe] 1** to turn on the source’s RF output.

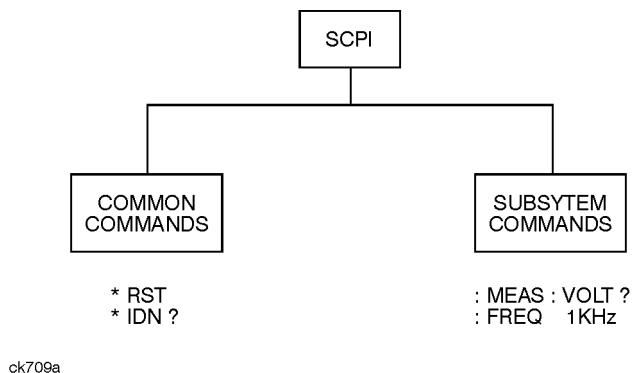
Precise talking means that the response format for a particular query is always the same. For example, if you query the power state when it is on (using **:POWER:ALC[:STATe]?**), the response is always 1, regardless of whether you previously sent **:POWER:ALC[:STATe] 1** or **:POWER:ALC[:STATe] ON**.

## Types of Commands

Commands can be separated into two groups, common commands and subsystem commands.

Common commands are generally not measurement related. They are used to manage macros, status registers, synchronization, and data storage. Common commands are easy to recognize because they all begin with an asterisk, such as **\*IDN?**, **\*OPC**, and **\*RST**. Common commands are defined by IEEE 488.2.

Subsystem commands include all measurement functions and some general purpose functions. Subsystem commands are distinguished by the colon used between keywords, as in **:FREQuency: CW?**. Each command subsystem is a set of commands that roughly corresponds to a functional block inside the instrument. For example, the **:POWeR** subsystem contains commands for power generation, while the **:STATus** subsystem contains commands for accessing status registers.



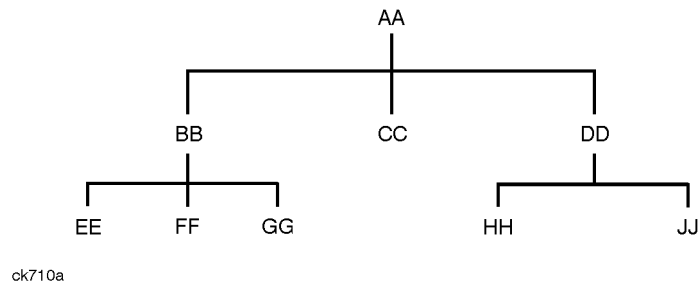
**Figure 2-8. SCPI Command Types**

The remaining paragraphs in this subsection discuss subsystem commands in more detail. Remember, some commands are implemented in one instrument and not in another, depending on its measurement function.

## Subsystem Command Trees

### Command Tree Structure

Most programming tasks involve subsystem commands. SCPI uses a hierarchical structure for subsystem commands similar to the file systems on most computers. In SCPI, this command structure is called a command tree.



**Figure 2-9. A Simplified Command Tree**

In the command tree shown in Figure 2-9, the command closest to the top is the root command, or simply “the root.” Notice that you must follow a particular path to reach lower level subcommands. For example, if you wish to access the GG command, you must follow the path AA to BB to GG.

### Paths Through the Command Tree

To access commands in different paths in the command tree, you must understand how an instrument interprets commands. The parser, a part of the instrument firmware, decodes each message sent to the instrument. The parser breaks up the message into component commands using a set of rules to determine the command tree path used. The parser keeps track of the current path: the level in the command tree where it expects to find the next command you send. This is important because the same keyword may appear in different paths. The particular path you use determines how the keyword is interpreted. The following rules are used by the parser:

#### Power On and Reset Message Terminators

After power is cycled or after **\*RST**, the current path is set to the root.

A message terminator, such as a **<new line>** character, sets the current path to the root. Many programming languages have output statements that send message terminators automatically. See “Details of Commands and Responses,” later in this chapter for more information about message terminators.

#### Colon

When a colon is placed between two command mnemonics, it moves the current path down one level in the command tree. For example, the colon in **MEAS : VOLT** specifies that **VOLT** is one level below **MEAS**. When the colon is the first character of a command, it specifies that the next command mnemonic is a root level command. For example, the colon in **: INIT** specifies that **INIT** is a root level command.

**Semicolon**

A semicolon separates two commands in the same message without changing the current path.

**White Space**

White space characters, such as <tab> and <space>, are generally ignored. There are two important exceptions. White space inside a keyword, such as:

**:FREQ uency**

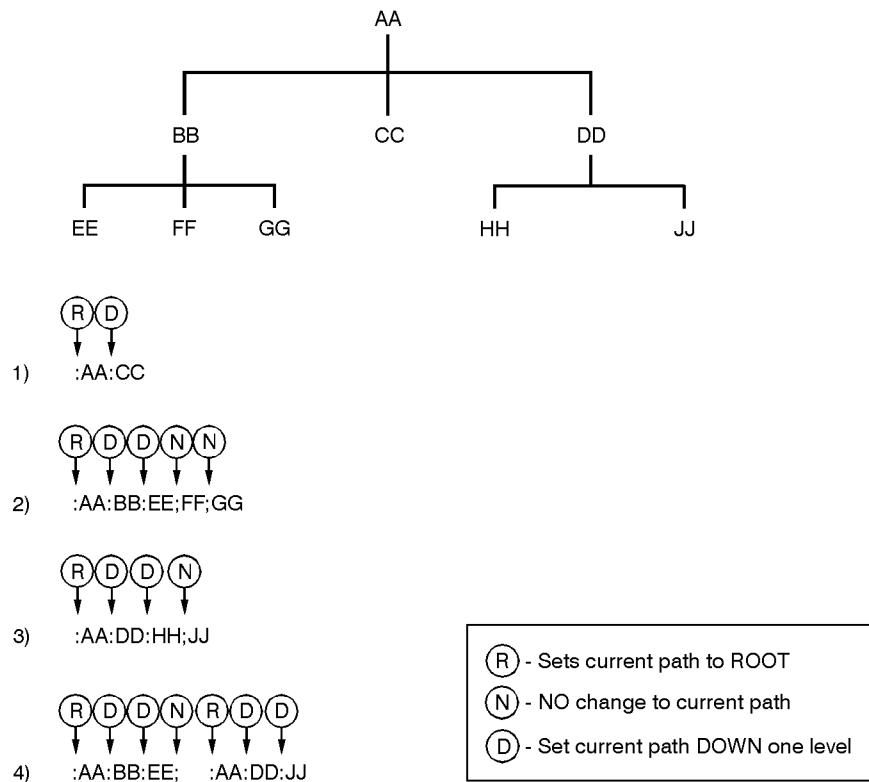
is not allowed. You must use white space to separate parameters from commands. For example, the <space> between **LEVEL** and **6.2** in the command **:POWER:LEVEL 6.2** is mandatory. White space does not affect the current path.

**Commas**

If a command requires more than one parameter, you must separate adjacent parameters using a comma. Commas do not affect the current path.

**Common Commands**

Common commands, such as **\*RST**, are not part of any subsystem. An instrument interprets them in the same way, regardless of the current path setting.



ck711a

**Figure 2-10. Proper Use of the Colon and Semicolon**

Figure 2-10 shows examples of how to use the colon and semicolon to navigate efficiently through the command tree. Notice how proper use of the semicolon can reduce the amount of information that must be sent over the interface.

Sending this message:

```
:AA:BB:EE; FF; GG
```

is the same as sending these three messages:

```
:AA:BB:EE
```

```
:AA:BB:FF
```

```
:AA:BB:GG
```

## More About Commands

### Query and Event Commands

You can query any value that you can set. For example, the presence of the signal generator **FREQUENCY:OFFSET** command implies that a **FREQUENCY:OFFSET?** also exists. If you see a command ending with a question mark, it is a query-only command. Some commands are events and cannot be queried. An event has no corresponding setting if it causes something to happen inside the instrument at a particular instant.

### Implied Commands

Implied commands appear in square brackets. If you send a subcommand immediately preceding an implied command, but do not send the implied command, the instrument assumes you intend to use the implied command and behaves just as if you had sent it. Notice that this means that the instrument expects you to include any parameters required by the implied command. The following example illustrates equivalent ways to program the signal generator using explicit and implied commands.

Example signal generator commands with and without an implied command:

<b>FREQUENCY[:CW] 500 MHz</b>	using explicit commands
<b>FREQUENCY 500 MHz</b>	using implied commands

### Optional Parameters

Optional parameter names are enclosed in square brackets. If you do not send a value for an optional parameter, the instrument chooses a default value. The instrument's command dictionary documents the values used for optional parameters.

## Program Message Examples

The following parts of the signal generator SCPI command set will be used to demonstrate how to create complete SCPI program messages:

```
:FREQuency  
:POWER
```

### Example 1

```
"FREQuency:START 500 MHz; STOP 1000 MHz"
```

The command is correct and will not cause errors. It is equivalent to sending the following:

```
"FREQuency:START 500 MHz; FREQuency:STOP 1000 MHz"
```

### Example 2

```
"POWER 10 DBM; :OFFSet 5 DB"
```

This command results in a command error. The command makes use of the default `POWER[:LEVEL][:IMMEDIATE]` node. When using a default node, there is no change to the current path position. Since there is no command `"OFFSet"` at the root, an error results. A correct way to send this is:

```
"POWER 10 DBM; :POWER:OFFSet 5 DB"
```

### Example 3

```
"POWER:OFFSet 5 DB; POWER 10 DBM"
```

This command results in a command error. The `POWER 10 DBM` portion of the command is missing a leading colon. The path level is dropped at each colon until it is in the `POWER:OFFSet` subsystem.

When the `POWER 10 DBM` command is sent, it then causes confusion because no such node occurs in the `POWER:OFFSet` subsystem. By adding a leading colon, the current path is reset to the root. The correct command is:

```
"POWER:OFFSet 5 DB; :POWER 10 DBM"
```

### Example 4

```
"FREQ 500 MHz; POWER 4 DBM"
```

In this example the keyword short form is used. The command is correct. It utilizes the default nodes of `[:CW]` and `[:LEVEL]`. Since default nodes do not affect the current path, it is not necessary to use a leading colon before `POWER`.

## Reading Instrument Errors

When debugging a program, you may want to know if an instrument error has occurred. The signal generator can display error messages on their front panel displays. If your system includes an instrument that does not have this capability, you can put the following code segment in your program to read error messages and print them on the controller's display.

```
10 !
20 ! The rest of your
30 ! variable declarations
40 ! Assign @box to 719
50 DIM Err_msg$(75)
60 INTEGER Err_num
70 !
80 ! Part of your program
90 ! that generates errors
100 !
110 !
200 REPEAT
210 OUTPUT @Box;":SYST:ERR?"
220 ! Query instrument error
230 ENTER @Box;Err_num,Err_msg$
240 ! Read error #, message
250 PRINT Err_num,Err_msg$
260 ! Print error message
270 UNTIL Err_num = 0
280 ! Repeat until no errors
290 !
300 ! The rest of your program
310 !
```

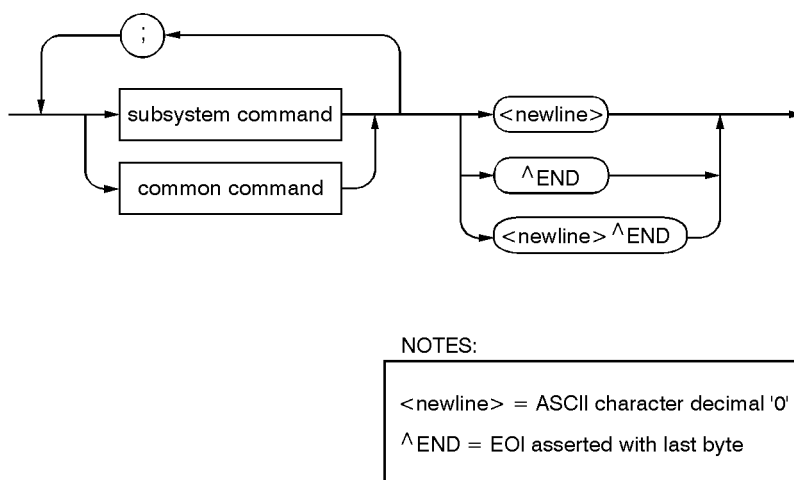


## Details of Commands and Responses

This section describes the syntax of SCPI commands and responses. It provides many examples of the data types used for command parameters and response data.

### Program Message Syntax

These program messages contain commands combined with appropriate punctuation and program message terminators.

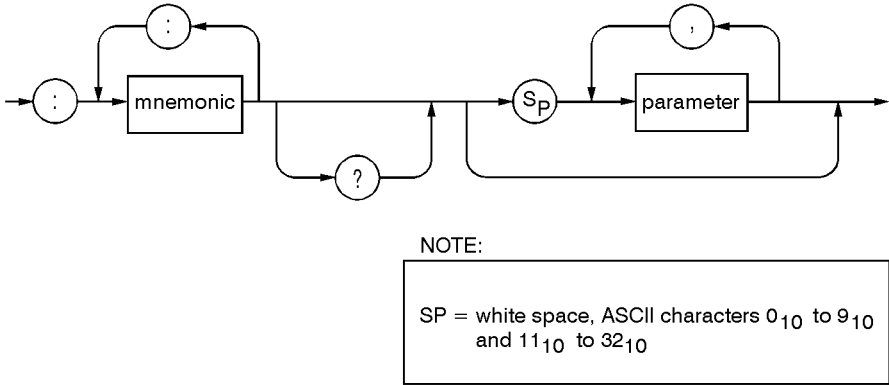


ck712a

**Figure 2-11. Simplified Program Message Syntax**

As Figure 2-11 shows, you can send common commands and subsystem commands in the same message. If you send more than one command in the same message, you must separate them with a semicolon. You must always end a program message with one of the three program message terminators shown in Figure 2-11. Use **<new line>**, **&^END**, or **<new line> &^END** as the program message terminator. The word **&^END** means that EOI is asserted on the HP-IB interface at the same time the preceding data byte is sent. Most programming languages send these terminators automatically. For example, if you use the HP BASIC **OUTPUT** statement, **<new line>** is automatically sent after your last data byte. If you are using a PC, you can usually configure the system to send whatever terminator you specify.

**SCPI Subsystem Command Syntax**

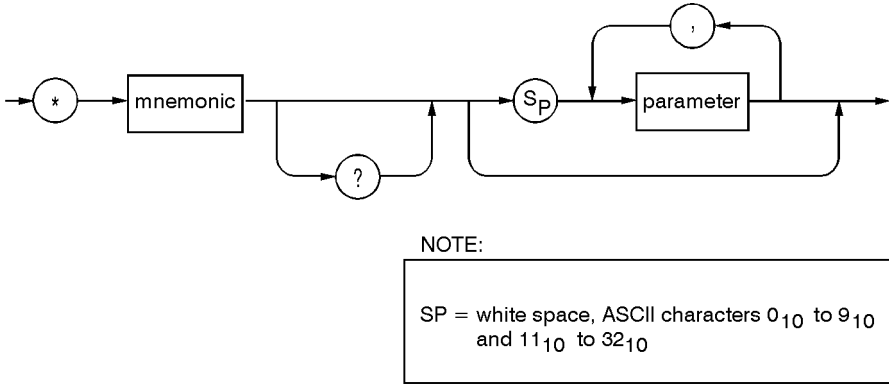


ck713a

**Figure 2-12. SCPI Simplified Subsystem Command Syntax**

As Figure 2-12 shows, there must be a <space> between the last command mnemonic and the first parameter in a subsystem command. This is one of the few places in SCPI where <space> is required. Note that if you send more than one parameter with a single command, you must separate adjacent parameters with a comma. Parameter types are explained later in this subsection.

**Common Command Syntax**

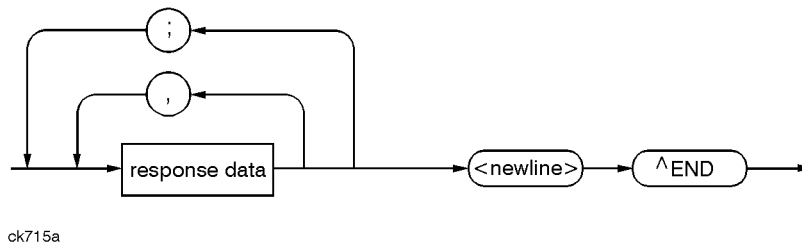


ck714a

**Figure 2-13. Simplified Common Command Syntax**

As with subsystem commands, use a <space> to separate a command mnemonic from subsequent parameters. Separate adjacent parameters with a comma. Parameter types are explained later in this section.

## Response Message Syntax



**Figure 2-14. Simplified Response Message Syntax**

Response messages can contain both commas and semicolons as separators. When a single query command returns multiple values, a comma separates each data item. When multiple queries are sent in the same message, the groups of data items corresponding to each query are separated by a semicolon. For example, the fictitious query `:QUERY1?:QUERY2?` might return a response message of:

```
<data1> , <data1> ; <data2> , <data2>
```

Response data types are explained later in this subsection. Note that `<new line><^END>` is always sent as a response message terminator.

## SCPI Data Types

SCPI defines different data formats for use in program messages and response messages. It does this to accommodate the principle of forgiving listening and precise talking. Forgiving listening means that instruments are flexible, accepting commands and parameters in various formats. Precise talking means an instrument always responds to a particular query in a predefined, rigid format. Parameter data types are designed to be flexible in the spirit of forgiving listening. Conversely, response data types are defined to meet the requirements of precise talking.

Parameter Types	Response Data Types
Numeric	Real or Integer
Extended Numeric	Integer
Discrete	Discrete
Boolean	Numeric Boolean
String	String
Block	Definite Length Block (all interfaces) Indefinite Length Block (not supported with RS-232)
Non-decimal Numeric	Hexadecimal Octal Binary

Each parameter type has one or more corresponding response data types. For example, a setting that you program using a numeric parameter returns either real or integer response data when queried. Whether real or integer response data is returned depends on the instrument used. However, precise talking requires that the response data type be clearly defined for a particular instrument and query. Chapter 4 contains information about data types for individual commands.

## Parameter Types

### Numeric Parameters

Numeric parameters are used in both subsystem commands and common commands. Numeric parameters accept all commonly used decimal representations of numbers including optional signs, decimal points, and scientific notation.

If an instrument setting programmed with a numeric parameter can only assume a finite number of values, the instrument automatically rounds the parameter. For example, if an instrument has a programmable output impedance of 50 or 75 ohms, and you specified **76.1** for output impedance, the value is rounded to **75**. If the instrument setting can only assume integer values, it automatically rounds the value to an integer. For example sending **\*ESE 10.123** is the same as sending **\*ESE 10**.

Examples of numeric parameters:

100	no decimal point required
100.	fractional digits optional
-1.23	leading signs allowed
4.56e<space>3	space allowed after e in exponentials
-7.89E-01	use either E or e in exponentials
+256	leading + allowed
.5	digits left of decimal point optional

### Extended Numeric Parameters

Most subsystems use extended numeric parameters to specify physical quantities. Extended numeric parameters accept all numeric parameter values and other special values as well. All extended numeric parameters accept **MAXimum** and **MINimum** as values. Other special values, such as **UP** and **DOWN** may be available as documented in Chapter 4. Notice that **MINimum** and **MAXimum** can be used to set or query values. The query forms are useful for determining the range of values allowed for a given parameter.

In some instruments, extended numeric parameters accept engineering unit suffixes as part of the parameter value.

Notice that extended numeric parameters are not used for common commands or **STATus** subsystem commands.

Examples of extended numeric parameters:

100.	any simple numeric values
-1.23	
4.56e<space>3	
-7.89E-01	
+256	
.5	
MAX	largest valid setting
MIN	valid setting nearest negative infinity
-100 mV	negative 100 millivolts

### Discrete Parameters

Use discrete parameters to program settings that have a finite number of values. Discrete parameters use mnemonics to represent each valid setting. They have a long and a short form, just like command mnemonics. You can use mixed upper and lower case letters for discrete parameters.

Examples of discrete parameters used with the **TRIG:SOURce** subsystem:

BUS	HP-IB triggering
IMMediate	immediate trigger
EXTernal	external triggering

Although discrete parameter values look like command keywords, do not confuse the two. In particular, be sure to use colons and spaces properly. Use a colon to separate command mnemonics from each other. Use a space to separate parameters from command mnemonics.

Examples of discrete parameters in commands:

```
100 OUTPUT @Source;"TRIGger:SOURce BUS"  
100 OUTPUT @Source;"TRIGger:SOURce IMMediate"  
100 OUTPUT @Source;"TRIGger:SOURce EXTernal"
```

### Boolean Parameters

Boolean parameters represent a single binary condition that is either true or false. There are only four possible representations for a Boolean parameter:

ON	Boolean true, upper/lower case allowed
OFF	Boolean false, upper/lower case allowed
1	Boolean true
0	Boolean false

### Block Parameters

A data block contains the data of primary interest. It may contain dimensioned data such as DATA(CURVe), or specific sets of data (WAVEform, etc.). At least one data block is required and multiple data blocks are allowed. The following table details the data block for a List Pattern data block, used to write pattern lists directly to the instrument's baseband generator board:

Bit 0 (1)	data value: 0 or 1 as required for a data bit.
Bit 1 (2)	Always 0
Bit 2 (4)	Burst control: 0 for burst off, 1 for burst on. All data values that require power out must have this bit on.
Bit 3 (8)	Always 0
Bit 4 (16)	Always 16
Bit 5 (32)	Always 0
Bit 6 (64)	Event 1 control: 0 or 1, as desired on the EVENT1 output.
Bit 7 (128)	Pattern reset: Reset the pattern to start after this entry is processed.

## Response Data Types

### Real Response Data

A large portion of all measurement data are formatted as real response data. Real response data are decimal numbers in either fixed decimal notation or scientific notation. Most high-level programming languages that support instrument I/O handle either decimal or scientific notation transparently.

Examples of real response data:

```
1.23E+0
-1.0E+2
+1.0E+2
0.5E+0
1.23
-100.0
+100.0
0.5
```

### Integer Response Data

Integer response data are decimal representations of integer values including optional signs. Most status register related queries return integer response data.

Examples of integer response data:

```
0      signs are optional
+100   leading + sign allowed
-100   leading sign allowed
256    never any decimal point
```

### Discrete Response Data

Discrete response data are similar to discrete parameters. The main difference is that discrete response data return only the short form of a particular mnemonic, in all upper case letters.

Examples of discrete response data:

```
IMM     Immediate
EXT     External
```



### String Response Data

String response data are similar to string parameters. The main difference is that string response data use only double quotes as delimiters, rather than single quotes. Embedded double quotes may be present in string response data. Embedded quotes appear as two adjacent double quotes with no characters between them.

Examples of string response data:

“This IS valid”

“SO IS THIS”” “

“I said, ““Hello!””””

---

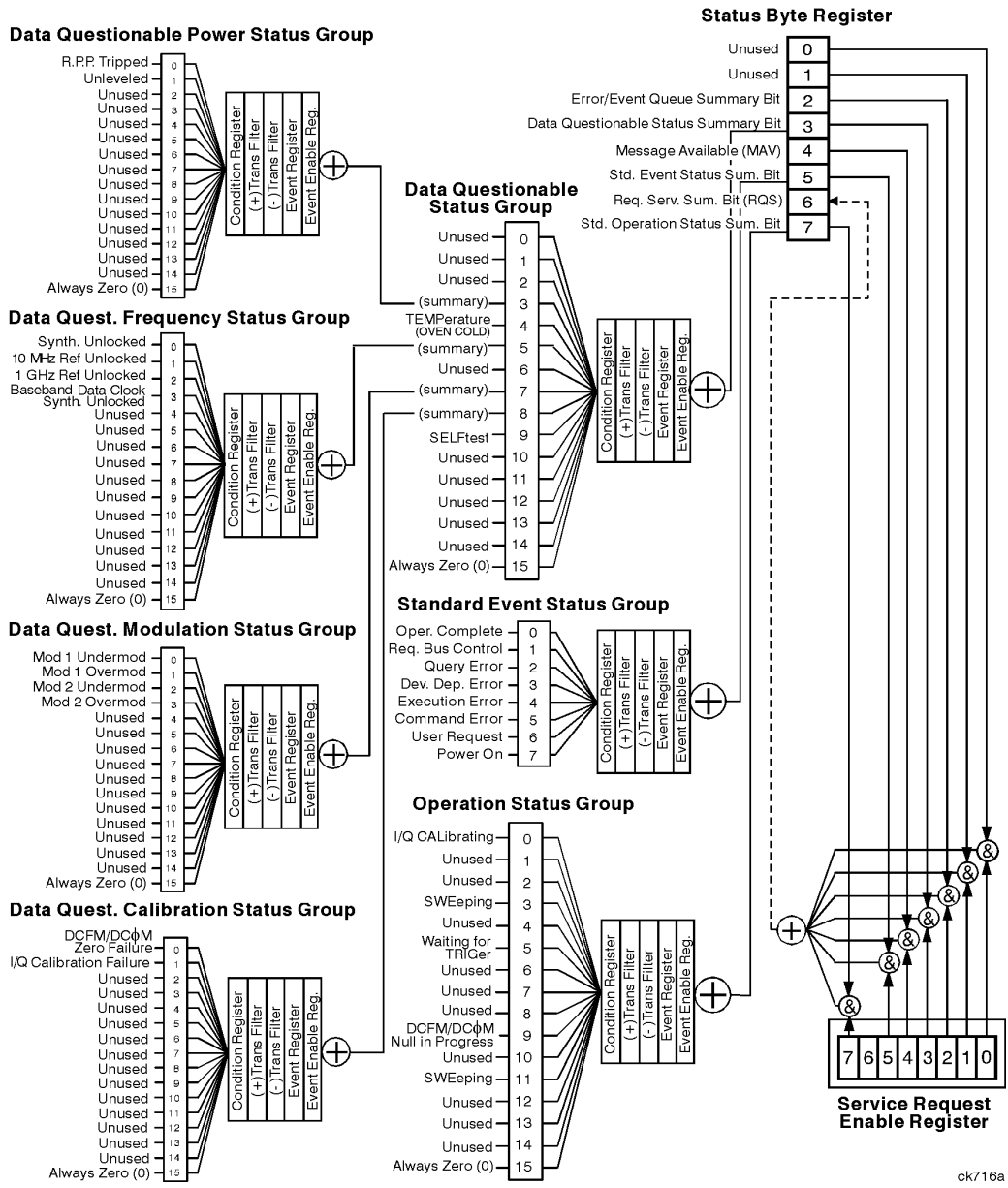
## Programming the Status Register System

The signal generator's instrument status system provides complete IEEE 488.2 Device Standard data structures for reporting instrument status using the register model.

The IEEE 488.2 register model of the status system is comprised of multiple registers which are arranged in a hierarchical order. The lower-priority status registers propagate their data to the higher-priority registers in the data structures by means of summary bits. The Status Byte Register is at the top of the hierarchy and contains the general status information for the instrument's events and conditions. All other individual registers are used to determine the specific events or conditions.

You can determine the state of certain instrument hardware and firmware events and conditions by programming the status register system.

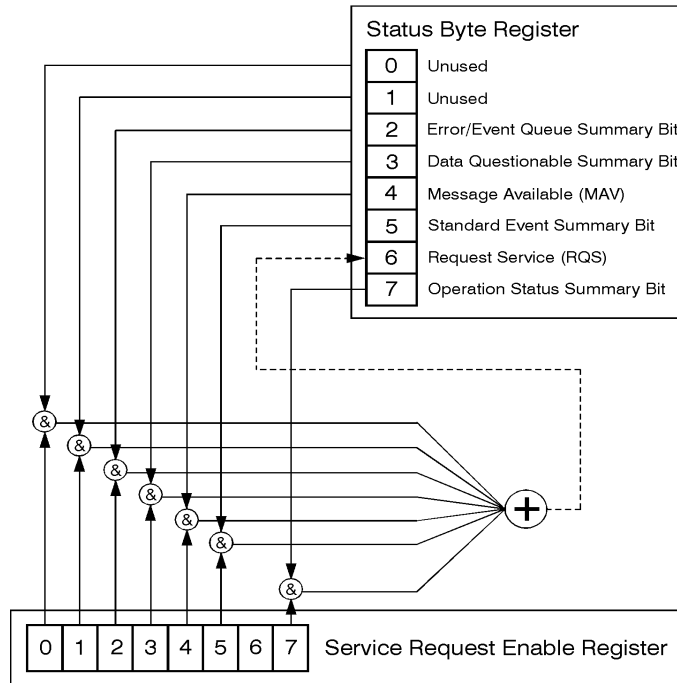
Programming Fundamentals  
 Programming the Status Register System



ck716a

Figure 2-15. The Overall Status Byte Register System

## Status Byte Group

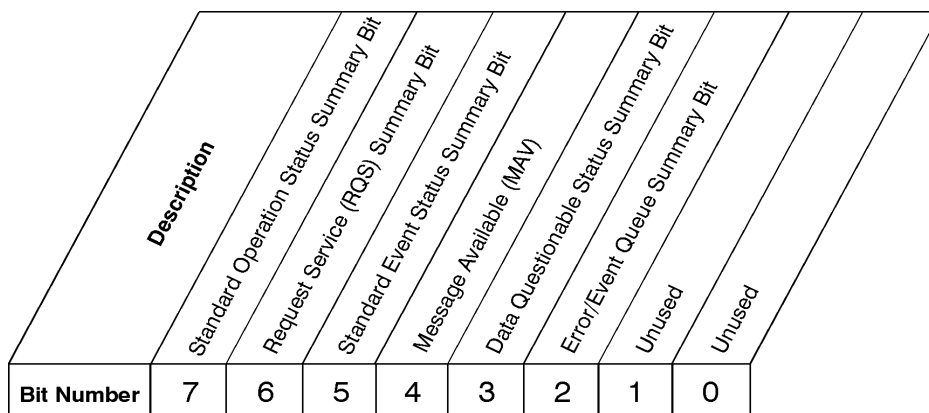


ck721a

Figure 2-16. The Status Byte Group

Programming Fundamentals  
 Programming the Status Register System

The Status Byte Group consists of the Status Byte Register and the Service Request Enable Register. The Status Byte Register contains the following bits:



\*STB?

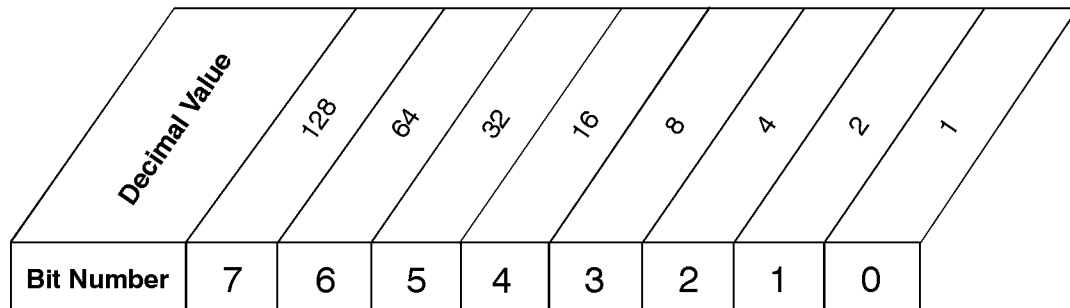
**Status Byte Register**

ck725a

Bit	Description
0, 1	These bits are always set to 0.
2	A 1 in this bit position indicates that the SCPI error queue is not empty. The SCPI error queue contains at least one error message.
3	A 1 in this bit position indicates that the Data Questionable summary bit has been set. The Data Questionable Event Register can then be read to determine the specific condition that caused this bit to be set.
4	A 1 in this bit position indicates that the signal generator has data ready in the output queue. There are no lower status groups that provide input to this bit.
5	A 1 in this bit position indicates that the Standard Event summary bit has been set. The Standard Event Status Register can then be read to determine the specific event that caused this bit to be set.
6	A 1 in this bit position indicates that the instrument has at least one reason to require service. This bit is also called the Master Summary Status bit (MSS). The individual bits in the Status Byte are individually ANDed with their corresponding service request enable register, then each individual bit value is ORed and input to this bit.
7	A 1 in this bit position indicates that the Standard Operation summary bit has been set. The Standard Operation Event Register can then be read to determine the specific condition that caused this bit to be set.

To query the Status Byte Register, send the command **\*STB?** The response will be the *decimal* sum of the bits which are set to 1. For example, if bit number 7 and bit number 3 are set to 1, the decimal sum of the 2 bits is 128 plus 8. So the decimal value 136 is returned.

In addition to the Status Byte Register, the Status Byte Group also contains a Service Request Enable Register. This register lets you choose which bits in the Status Byte Register will trigger a service request. Send the **\*SRE <num>** command where **<num>** is the sum of the decimal values of the bits you want to enable plus the decimal value of bit 6. For example, to enable bit 7 so that whenever the Standard Operation Status Register summary bit is set to 1 it will trigger a service request, send the command **\*SRE 192** (128 + 64). You must always enable bit 6 when you enable any other bits for a service request. The command **\*SRE?** returns the decimal value of the sum of the bits previously enabled with the **\*SRE <num>** command.



\*SRE <num>  
 \*SRE?

**Service Request Enable Register**

ck726a

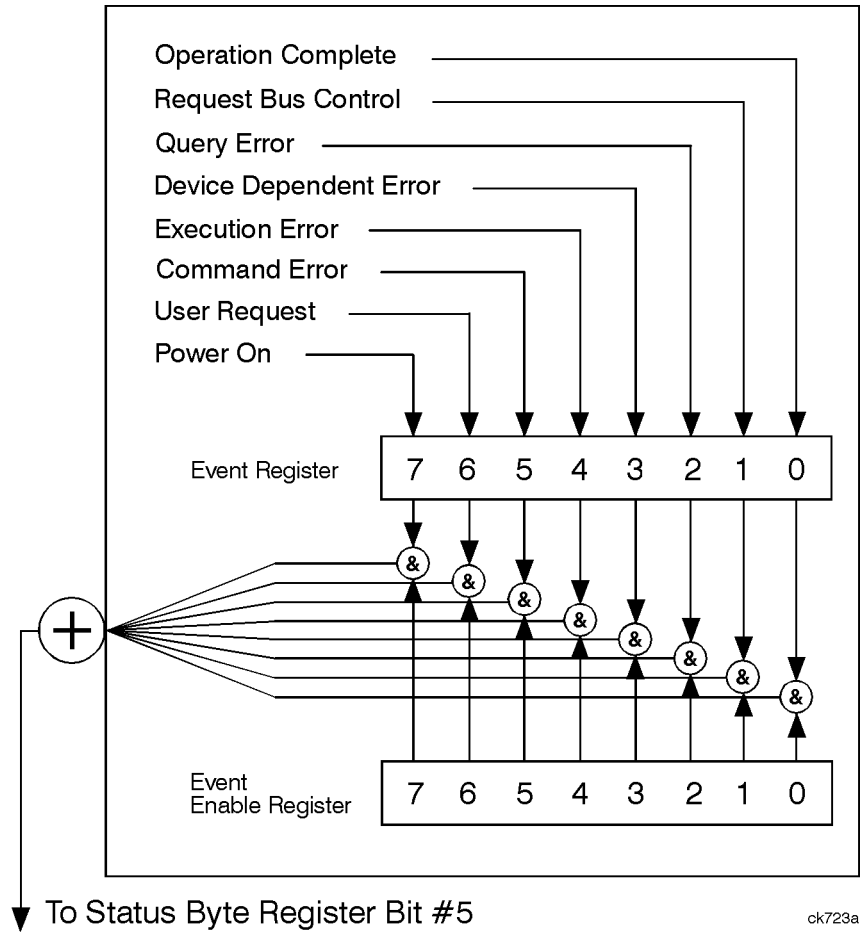
## Status Groups

The Standard Operation Status Group and the Data Questionable Status Group each consist of the following registers; the Standard Event Status Group is similar but does not have negative or positive transition filters.

<b>Condition Register</b>	A condition register continuously monitors the hardware and firmware status of the signal generator. There is no latching or buffering for a condition register; it is updated in real time.
<b>Negative Transition Filter</b>	A negative transition filter specifies the bits in the condition register that will set corresponding bits in the event register when the condition bit changes from 1 to 0.
<b>Positive Transition Filter</b>	A positive transition filter specifies the bits in the condition register that will set corresponding bits in the event register when the condition bit changes from 0 to 1.
<b>Event Register</b>	An event register latches transition events from the condition register as specified by the positive and negative transition filters. Bits in the event register are latched, and once set, they remain set until cleared by either querying the register contents or sending the <b>*CLS</b> command.
<b>Event Enable Register</b>	An enable register specifies the bits in the event register that can generate a summary bit. The signal generator logically ANDs corresponding bits in the event and enable registers and ORs all the resulting bits to produce a summary bit. Summary bits are, in turn, used by the Status Byte Register.

In general, a status group is a set of related registers whose contents are programmed in order to produce status summary bits. In each status group, corresponding bits in the condition register are filtered by the negative and positive transition filters and stored in the event register. The contents of the event register are logically ANDed with the contents of the enable register and the result is logically ORed to produce a status summary bit in the Status Byte Register.

**Standard Event Status Group**

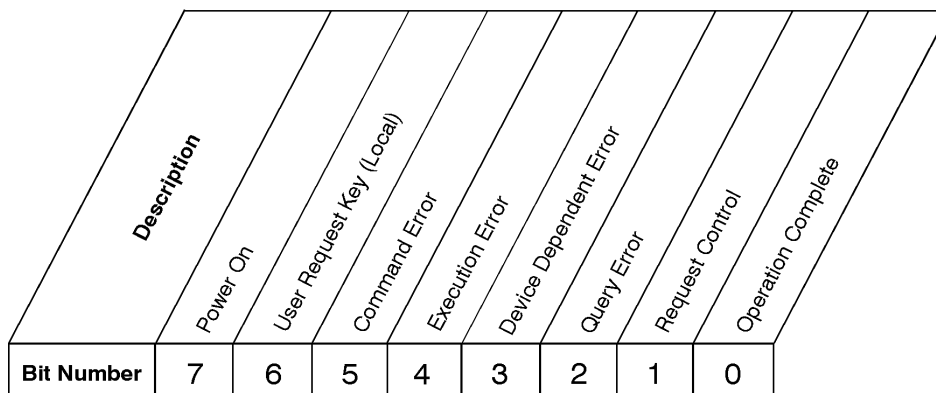


**Figure 2-17. The Standard Event Status Group**

The Standard Event Status Group is used to determine the specific event that set bit 5 in the Status Byte Register. The Standard Event Status Group consists of the Standard Event Status Register (an event register) and the Standard Event Status Enable Register. The Standard Event Status Register contains the following bits:



Programming Fundamentals  
 Programming the Status Register System



\*ESR?

**Standard Event Status Register**

ck727a

Bit	Description
0	A 1 in this bit position indicates that all pending signal generator operations were completed following execution of the *OPC command.
1	This bit is always set to 0. (The signal generator does not request control.)
2	A 1 in this bit position indicates that a query error has occurred. Query errors have SCPI error numbers from -499 to -400.
3	A 1 in this bit position indicates that a device dependent error has occurred. Device dependent errors have SCPI error numbers from -399 to -300 and 1 to 32767.
4	A 1 in this bit position indicates that an execution error has occurred. Execution errors have SCPI error numbers from -299 to -200.
5	A 1 in this bit position indicates that a command error has occurred. Command errors have SCPI error numbers from -199 to -100.
6	A 1 in this bit position indicates that the <b>Local</b> key has been pressed. This is true even if the signal generator is in local lockout mode.
7	A 1 in this bit position indicates that the signal generator has been turned off and then on.

To query the Standard Event Status Register, send the command \*ESR?. The response will be the *decimal* sum of the bits which are set to 1. For example, if bit number 7 and bit number 3 are set to 1, the decimal sum of the 2 bits is 128 plus 8. So the decimal value 136 is returned.

<b>Decimal Value</b>	128	64	32	16	8	4	2	1
<b>Bit Number</b>	7	6	5	4	3	2	1	0

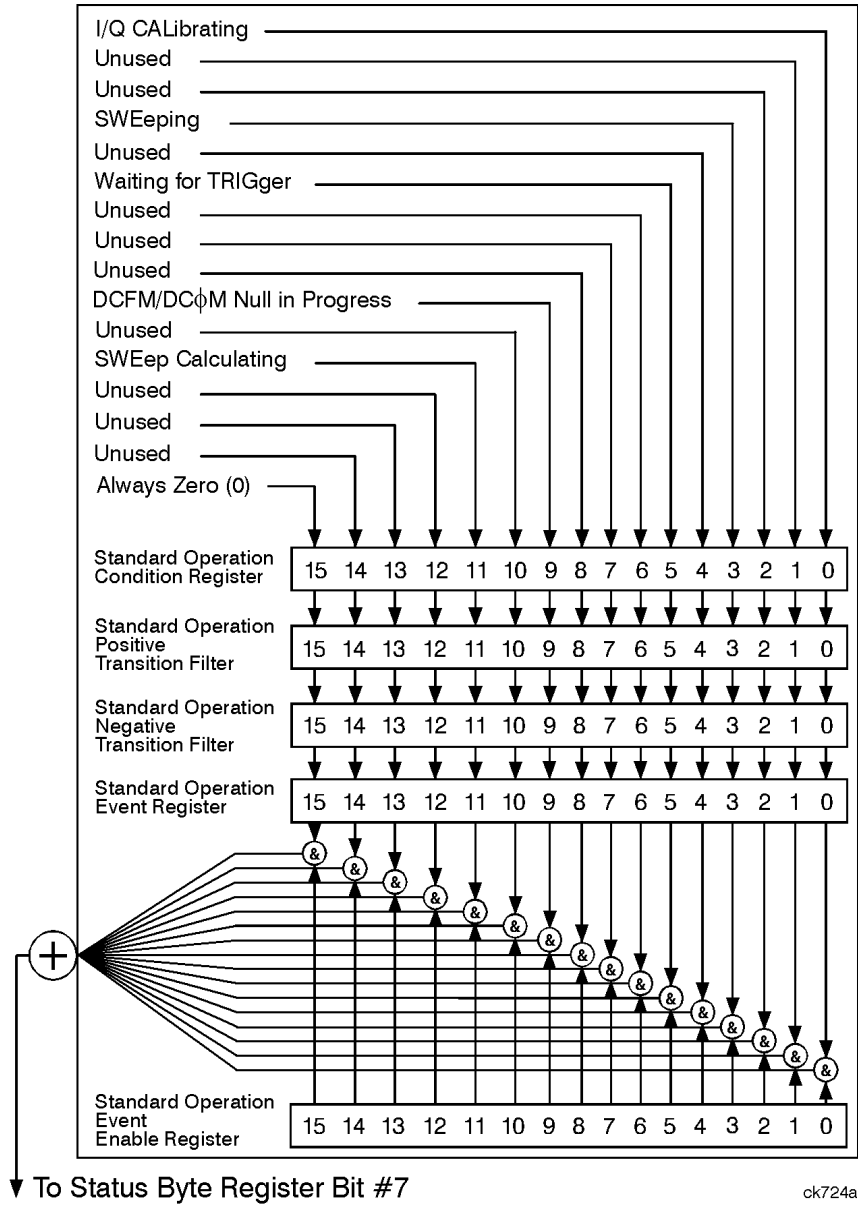
\*ESE <num>  
 \*ESE?

### Standard Event Status Enable Register

ck728a

In addition to the Standard Event Status Register, the Standard Event Status Group also contains a Standard Event Status Enable Register. This register lets you choose which bits in the Standard Event Status Register will set the summary bit (bit 5 of the Status Byte Register) to 1. Send the **\*ESE <num>** command where **<num>** is the sum of the decimal values of the bits you want to enable. For example, to enable bit 7 and bit 6 so that whenever either of those bits is set to 1, the Standard Event Status summary bit of the Status Byte Register will be set to 1, send the command **\*ESE 192** (128 + 64). The command **\*ESE?** returns the decimal value of the sum of the bits previously enabled with the **\*ESE <num>** command.

**Standard Operation Status Group**



ck724a

**Figure 2-18. The Standard Operation Status Group**

The Standard Operation Status Group is used to determine the specific event that set bit 7 in the Status Byte Register. The Standard Operation Status Group consists of the Standard Operation Condition Register, the Standard Operation Negative Transition Filter, the Standard Operation Positive Transition Filter, the Standard Operation Event Register, and the Standard Operation Event Enable Register. The Standard Operation Condition Register contains the following bits:

	<i>Description</i>																			
	Always Zero (0)	Unused	Unused	Unused	SWEEP Calculating	Unused	DCFM/DCΦM Null In Progress	Unused	Unused	Unused	Waiting for TRIGGER	Unused	SWEEPing	Unused	Unused	I/Q CALibrating				
<b>Bit Number</b>	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				

STATus:OPERation:CONDition?

### Standard Operation Condition Register

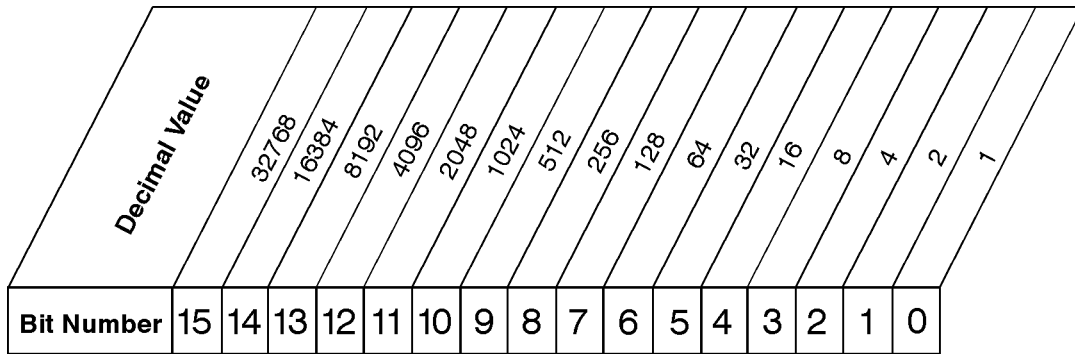
ck729a

Bit	Description
0	A 1 in this bit position indicates that an I/Q calibration is being performed.
1, 2	Unused. These bits are always set to 0.
3	A 1 in this bit position indicates that a sweep is in progress.
4	Unused. This bit is always set to 0.
5	A 1 in this bit position indicates that the source is in a “wait for trigger” state of the trigger model.
6,7,8	Unused. These bits are always set to 0.
9	A 1 in this bit position indicates that the signal generator is currently performing a DCFM/DCΦM zero calibration.
10	Unused. This bit is always set to 0.
11	A 1 in this bit position indicates that the signal generator is currently doing the necessary pre-sweep calculations.
12, 13, 14, 15	Unused. These bits are always set to zero.
15	Always Zero (0).

The Standard Operation Condition Register continuously monitors the hardware and firmware status of the instrument. Condition registers are read-only. To query the condition register, send the command **STATUS:OPERation:CONDition?** The response will be the *decimal* sum of the bits which are set to 1. For example, if bit number 9 and bit number 3 are set to 1, the decimal sum of the 2 bits is 512 plus 8. So the decimal value 520 is returned.

The transition filter specifies which types of bit state changes in the condition register will set corresponding bits in the event register. The changes may be positive (from 0 to 1) or negative (from 1 to 0). Send the command **STATUS:OPERation:NTRansition <num>** (negative) or **STATUS:OPERation:PTRansition <num>** (positive) where **<num>** is the sum of the decimal values of the bits you want to enable.

The Standard Operation Event Register latches transition events from the condition register as specified by the transition filters. Event registers are destructive read-only. Reading data from an event register will clear the content of that register. To query the event register, send the command **STATUS:OPERation[:EVENT]?**



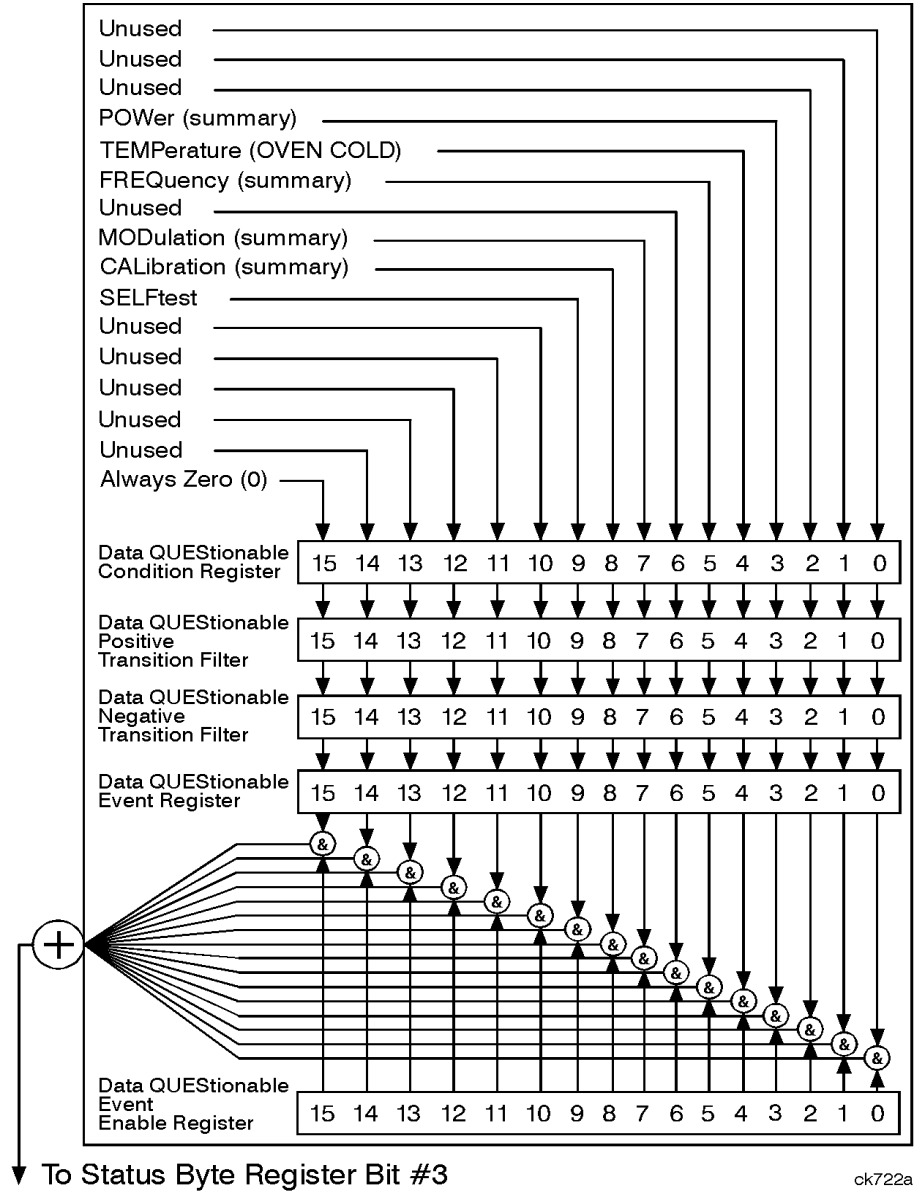
STATUS:OPERation:ENABle <num>  
 STATUS:OPERation:ENABle?

**Standard Operation Event Enable Register**

ck730a

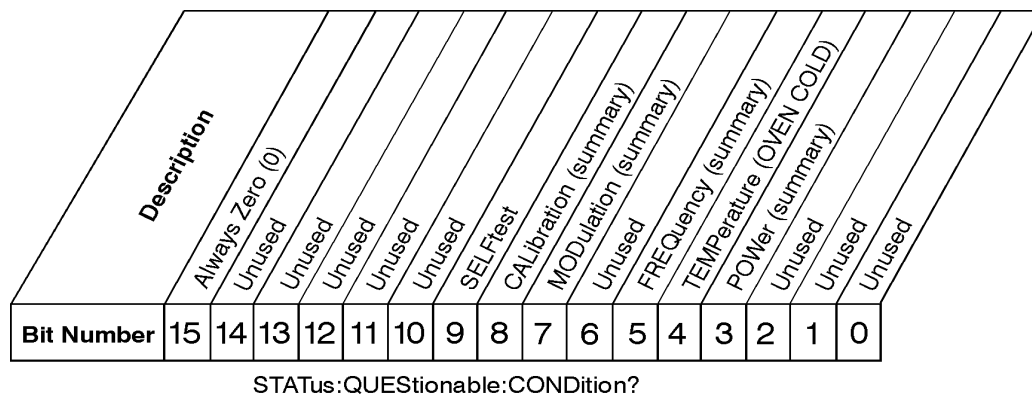
The Standard Operation Status Group also contains a Standard Operation Event Enable Register. This register lets you choose which bits in the Standard Operation Event Register will set the summary bit (bit 7 of the Status Byte Register) to 1. Send the **STATUS:OPERation:ENABle <num>** command where **<num>** is the sum of the decimal values of the bits you want to enable. For example, to enable bit 9 and bit 3 so that whenever either of those bits is set to 1, the Standard Operation Status summary bit of the Status Byte Register will be set to 1, send the command **STAT:OPER:ENAB 520** (512 + 8). The command **STATUS:OPERation:ENABle?** returns the decimal value of the sum of the bits previously enabled with the **STATUS:OPERation:ENABle <num>** command.

**Data Questionable Status Group**



**Figure 2-19. The Data Questionable Status Group**

The Data Questionable Status Group is used to determine the specific event that set bit 3 in the Status Byte Register. The Data Questionable Status Group consists of the Data Questionable Condition Register, the Data Questionable Negative Transition Filter, the Data Questionable Positive Transition Filter, the Data Questionable Event Register, and the Data Questionable Event Enable Register. The Data Questionable Condition Register contains the following bits:



**Data Questionable Condition Register**

ck731a

Bit	Description
0, 1, 2	Unused. These bits are always set to 0.
3	This is a summary bit taken from the QUESTionable:POWER register. A 1 in this bit position indicates that one of the following may have happened: The ALC (Automatic Leveling Control) is unable to maintain a leveled RF output power (i.e., ALC is UNLEVELED), or the reverse power protection circuit has been tripped.
4	A 1 in this bit position indicates that the internal reference oscillator (reference oven) is cold. (Option 1EH only.)
5	This is a summary bit taken from the QUESTionable:FREQuency register. A 1 in this bit position indicates that one of the following may have happened: synthesizer PPL unlocked, 10 MHz reference VCO PPL unlocked, heterodyned VCO PPL unlocked, or baseband PPL unlocked. See the Data Questionable Frequency Status Group for more information.
6	Unused. This bit is always set to 0.
7	This is a summary bit taken from the QUESTionable:MODulation register. A 1 in this bit position indicates that one of the following may have happened: modulation source 1 underrange, modulation source 1 overrange, modulation source 2 underrange, or modulation source 2 overrange. See the Data Questionable Modulation Status Group for more information.

Bit	Description
8	This is a summary bit taken from the QUESTionable:CALibration register. A 1 in this bit position indicates that one of the following may have happened: an error has occurred in the DCFM/DCΦM zero calibration or an error has occurred in the I/Q calibration. See the Data Questionable Calibration Status Group for more information.
9	A 1 in this bit position indicates that a self-test has failed during power-up. This bit can only be cleared by cycling the instrument's line power. *CLS will not clear this bit.
10, 11, 12, 13, 14	Unused. These bits are always set to 0.
15	Always Zero (0).

The Data Questionable Condition Register continuously monitors the hardware and firmware status of the instrument. Condition registers are read-only. To query the condition register, send the command **STATUS:QUESTIONABLE:CONDITION?** The response will be the *decimal* sum of the bits which are set to 1. For example, if bit number 9 and bit number 3 are set to 1, the decimal sum of the 2 bits is 512 plus 8. So the decimal value 520 is returned.

The transition filter specifies which types of bit state changes in the condition register will set corresponding bits in the event register. The changes may be positive (from 0 to 1) or negative (from 1 to 0). Send the command **STATUS:QUESTIONABLE:NTRANSITION <num>** (negative) or **STATUS:QUESTIONABLE:PTRANSITION <num>** (positive) where <num> is the sum of the decimal values of the bits you want to enable.

The Data Questionable Event Register latches transition events from the condition register as specified by the transition filters. Event registers are destructive read-only. Reading data from an event register will clear the content of that register. To query the event register, send the command **STATUS:QUESTIONABLE[:EVENT]?**



<b>Decimal Value</b>																	
	32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
<b>Bit Number</b>	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	

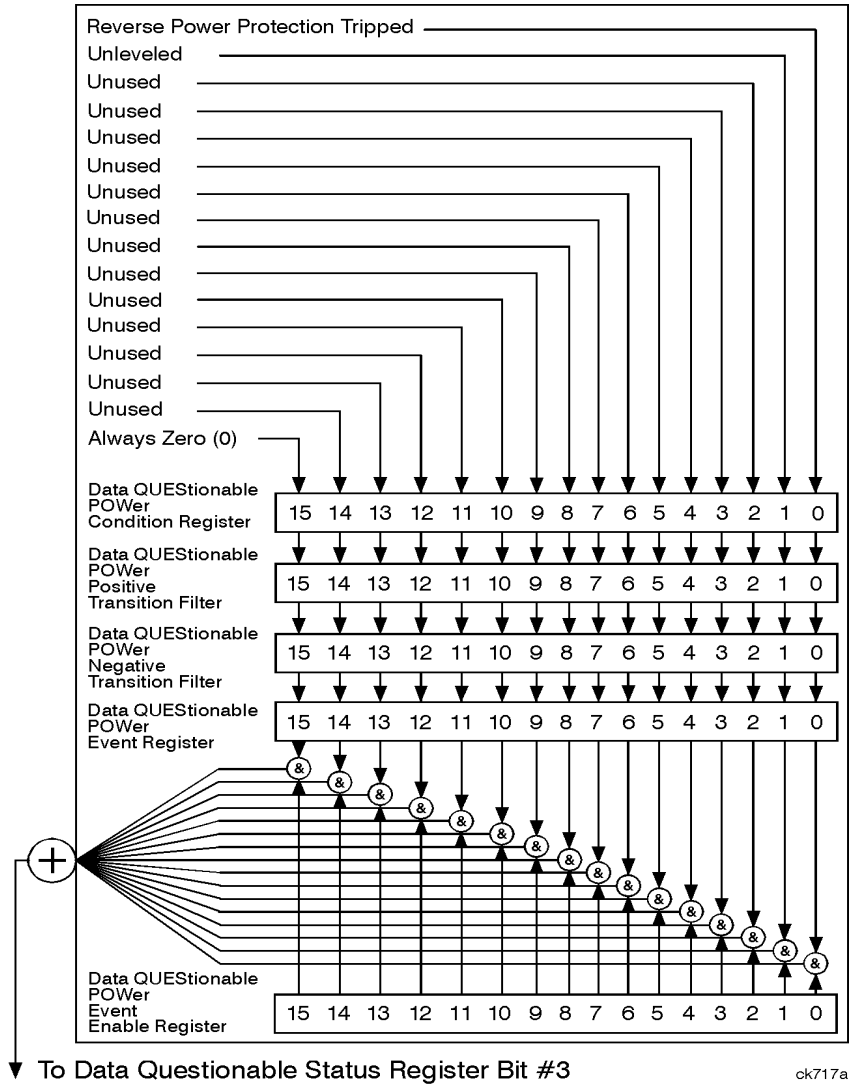
STATus:QUESTionable:ENABle <num>  
 STATus:QUESTionable:ENABle?

### Data Questionable Event Enable Register

ck732a

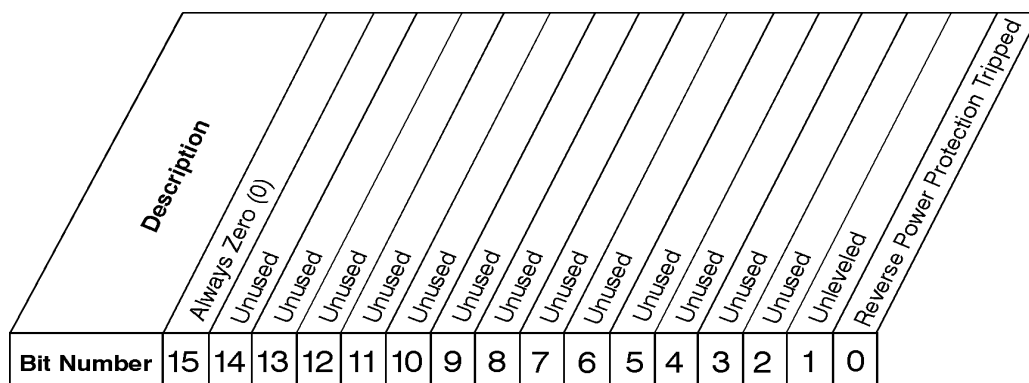
The Data Questionable Status Group also contains a Data Questionable Event Enable Register. This register lets you choose which bits in the Data Questionable Event Register will set the summary bit (bit 3 of the Status Byte Register) to 1. Send the **STATus:QUESTionable:ENABle <num>** command where **<num>** is the sum of the decimal values of the bits you want to enable. For example, to enable bit 9 and bit 3 so that whenever either of those bits is set to 1, the Data Questionable Status summary bit of the Status Byte Register will be set to 1, send the command **STAT:QUES:ENAB 520** (512 + 8). The command **STATus:QUESTionable:ENABle?** returns the decimal value of the sum of the bits previously enabled with the **STATus:QUESTionable:ENABle <num>** command.

**Data Questionable Power Status Group**



**Figure 2-20. The Data Questionable Power Status Group**

The Data Questionable Power Status Group is used to determine the specific event that set bit 3 in the Data Questionable Condition Register. The Data Questionable Power Status Group consists of the Data Questionable Power Condition Register, the Data Questionable Power Negative Transition Filter, the Data Questionable Power Positive Transition Filter, the Data Questionable Power Event Register, and the Data Questionable Power Event Enable Register. The Data Questionable Power Condition Register contains the following bits:



STATUS:QUESTIONABLE:POWER:CONDITION?

**Data Questionable Power Condition Register**

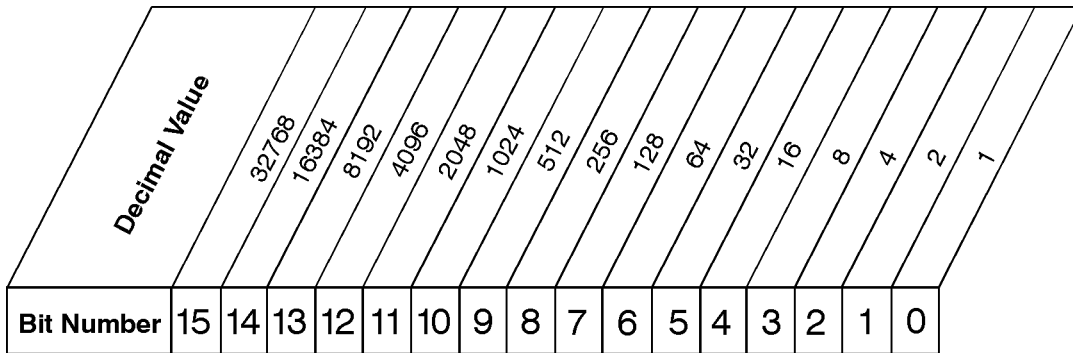
ck735a

Bit	Description
0	A 1 in this bit indicates that the reverse power protection circuit has been tripped. There is no output in this state. Any conditions which may have caused reverse power should be corrected. After correcting the problem, the RPP circuit may be reset by sending the remote SCPI command statement <b>:OUTPUT:PROTECTION:CLEAR</b> or by pressing the <b>Reset RPP</b> softkey on the front panel. In HP 8648 mode, any SCPI command will reset the reverse power protection circuit.
1	A 1 in this bit indicates that the output leveling loop is unable to set the output power.
2-14	Unused. These bits are always set to 0.
15	Always Zero (0).

The Data Questionable Power Condition Register continuously monitors output power status of the instrument. Condition registers are read-only. To query the condition register, send the command **STATUS:QUESTIONABLE:POWER:CONDITION?** The response will be the *decimal* sum of the bits which are set to 1.

The transition filter specifies which types of bit state changes in the condition register will set corresponding bits in the event register. The changes may be positive (from 0 to 1) or negative (from 1 to 0). Send the command **STATUS:QUESTIONABLE:POWER:NTRANSITION <num>** (negative) or **STATUS:QUESTIONABLE:POWER:PTRANSITION <num>** (positive) where **<num>** is the sum of the decimal values of the bits you want to enable.

The Data Questionable Power Event Register latches transition events from the condition register as specified by the transition filters. Event registers are destructive read-only. Reading data from an event register will clear the content of that register. To query the event register, send the command **STATus:QUESTionable:POWer[:EVENT]?**



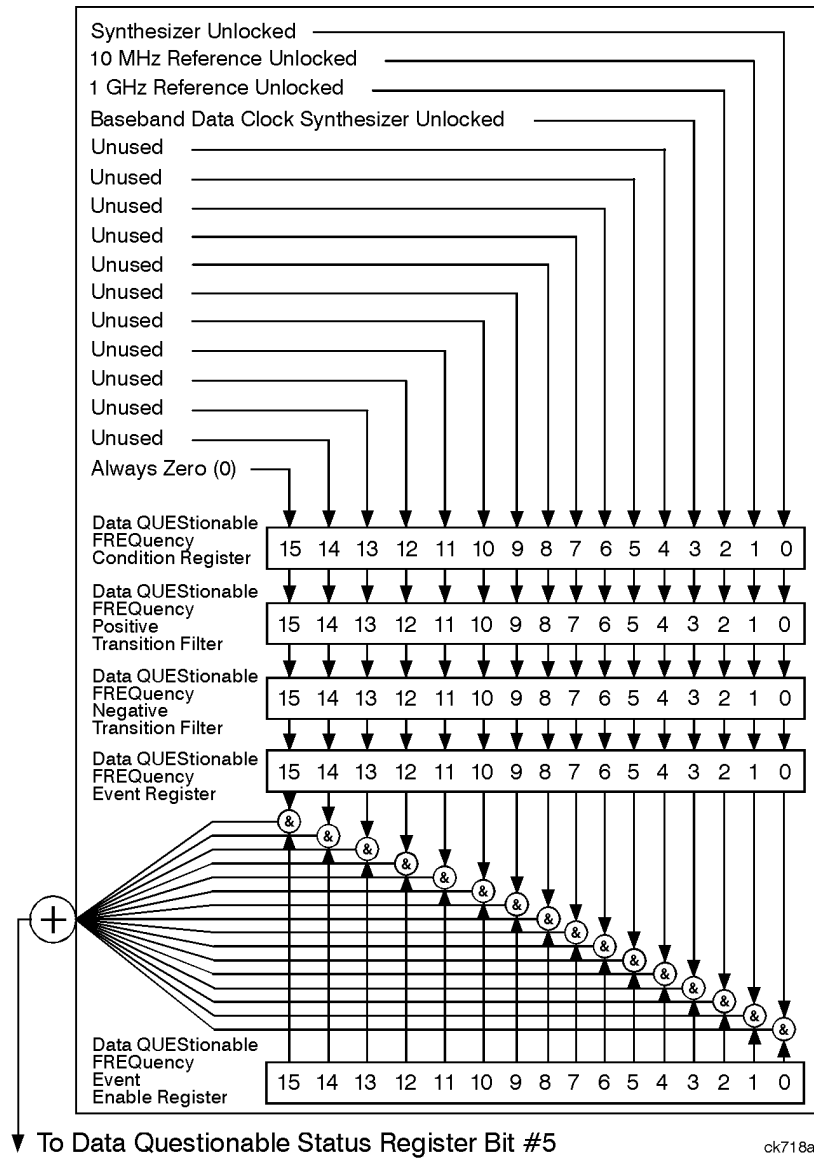
STATus:QUESTionable:POWer:ENABle <num>  
 STATus:QUESTionable:POWer:ENABle?

### Data Questionable Power Event Enable Register

ck736a

The Data Questionable Power Status Group also contains a Data Questionable Power Event Enable Register. This register lets you choose which bits in the Data Questionable Power Event Register will set the summary bit (bit 3 of the Data Questionable Condition Register) to 1. Send the **STATus:QUESTionable:POWer:ENABle <num>** command where <num> is the sum of the decimal values of the bits you want to enable. For example, to enable bit 9 and bit 3 so that whenever either of those bits is set to 1, the Data Questionable Power summary bit of the Data Questionable Condition Register will be set to 1, send the command **STAT:QUES:POW:ENAB 520** (512 + 8). The command **STATus:QUESTionable:POWer:ENABle?** returns the decimal value of the sum of the bits previously enabled with the **STATus:QUESTionable:POWer:ENABle <num>** command.

**Data Questionable Frequency Status Group**



**Figure 2-21. Data Questionable Frequency Status Group**

The Data Questionable Frequency Status Group is used to determine the specific event that set bit 5 in the Data Questionable Condition Register. The Data Questionable Frequency Status Group consists of the Data Questionable Frequency Condition Register, the Data Questionable Frequency Negative Transition

Filter, the Data Questionable Frequency Positive Transition Filter, the Data Questionable Frequency Event Register, and the Data Questionable Frequency Event Enable Register. The Data Questionable Frequency Condition Register contains the following bits:

Bit Number	Description
15	Always Zero (0)
14	Unused
13	Unused
12	Unused
11	Unused
10	Unused
9	Unused
8	Unused
7	Unused
6	Unused
5	Unused
4	Unused
3	Unused
2	Baseband Data Clock Synthesizer Unlocked
1	1 GHz Reference Unlocked
0	10 MHz Reference Unlocked Synthesizer Unlocked

STATus:QUESTionable:FREQUENCY:CONDition?

**Data Questionable Frequency Condition Register**

ck733a

Bit	Description
0	A 1 in this bit indicates that the synthesizer is unlocked.
1	A 1 in this bit indicates that the 10 MHz reference signal is unlocked.
2	A 1 in this bit indicates that the 1 GHz reference signal is unlocked.
3	A 1 in this bit indicates that the baseband data clock synthesizer is unlocked.
4-14	Unused. These bits are always set to 0.
15	Always Zero (0).

The Data Questionable Frequency Condition Register continuously monitors output frequency status of the instrument. Condition registers are read-only. To query the condition register, send the command **STATus:QUESTionable:FREQUENCY:CONDition?** The response will be the *decimal* sum of the bits which are set to 1.

The transition filter specifies which types of bit state changes in the condition register will set corresponding bits in the event register. The changes may be positive (from 0 to 1) or negative (from 1 to 0). Send the command **STATus:QUESTionable:FREQUENCY:NTRansition <num>** (negative) or **STATus:QUESTionable:FREQUENCY:PTRansition <num>** (positive) where **<num>** is the sum of the decimal values of the bits you want to enable.

The Data Questionable Frequency Event Register latches transition events from the condition register as specified by the transition filters. Event registers are destructive read-only. Reading data from an event register will clear the content of that register. To query the event register, send the command `STATUS:QUESTIONABLE:FREQUENCY[:EVENT]?`

<b>Decimal Value</b>																
		32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2
<b>Bit Number</b>	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

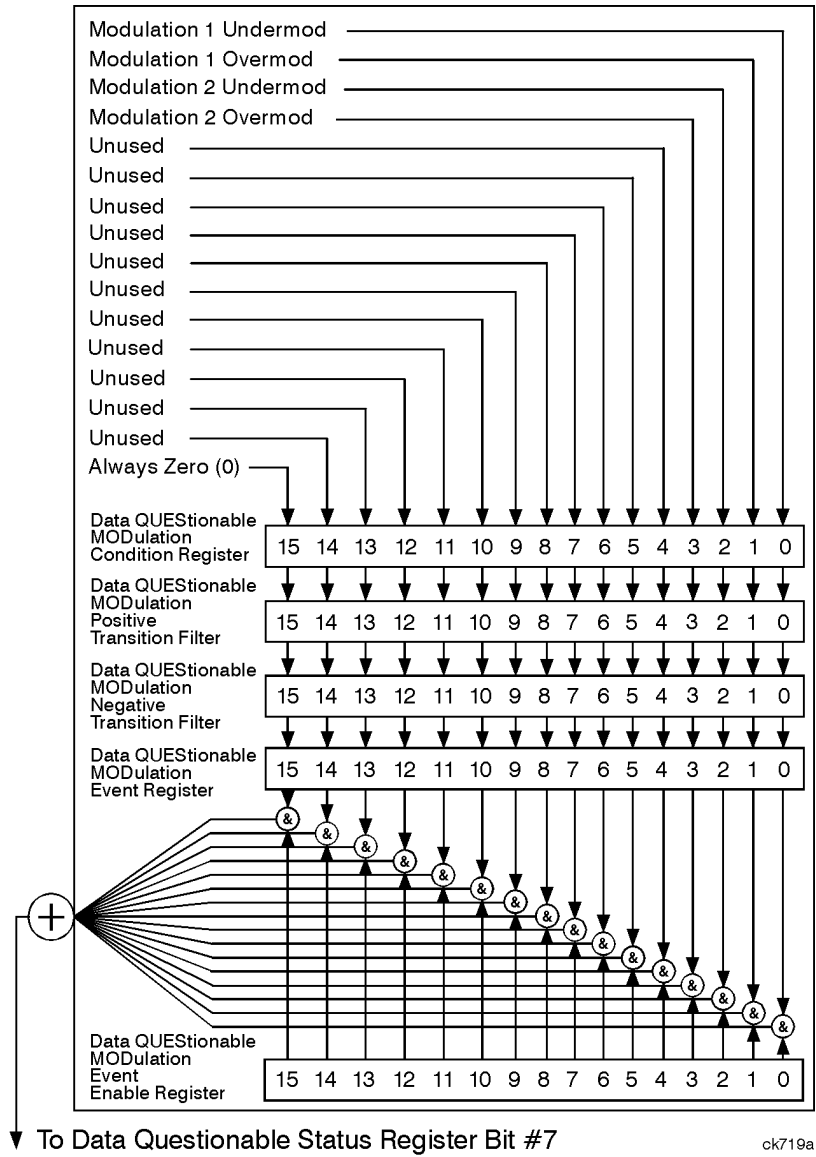
`STATUS:QUESTIONABLE:FREQUENCY:ENABLE <num>`  
`STATUS:QUESTIONABLE:FREQUENCY:ENABLE?`

### Data Questionable Frequency Event Enable Register

ck734a

The Data Questionable Frequency Status Group also contains a Data Questionable Frequency Event Enable Register. This register lets you choose which bits in the Data Questionable Frequency Event Register will set the summary bit (bit 5 of the Data Questionable Condition Register) to 1. Send the `STATUS:QUESTIONABLE:FREQUENCY:ENABLE <num>` command where `<num>` is the sum of the decimal values of the bits you want to enable. For example, to enable bit 9 and bit 3 so that whenever either of those bits is set to 1, the Data Questionable Frequency summary bit of the Data Questionable Condition Register will be set to 1, send the command `STAT:QUES:FREQ:ENAB 520` (512 + 8). The command `STATUS:QUESTIONABLE:FREQ:ENABLE?` returns the decimal value of the sum of the bits previously enabled with the `STATUS:QUESTIONABLE:FREQUENCY:ENABLE <num>` command.

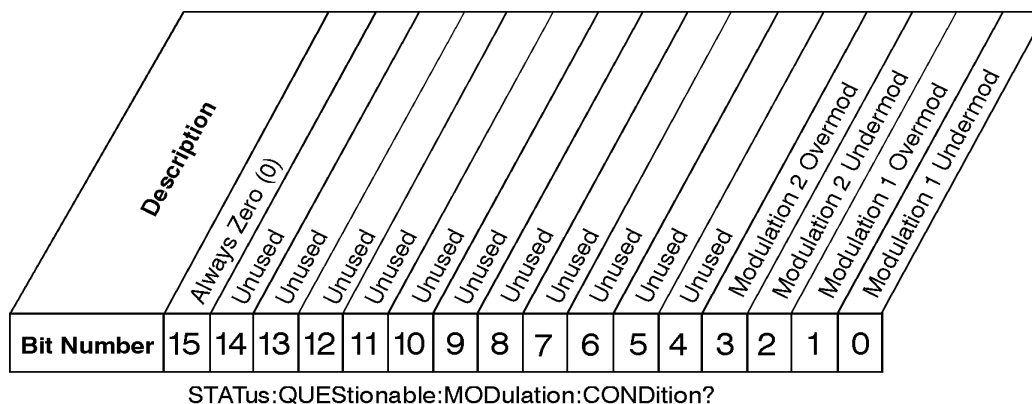
**Data Questionable Modulation Status Group**



**Figure 2-22. Data Questionable Modulation Status Group**



The Data Questionable Modulation Status Group is used to determine the specific event that set bit 7 in the Data Questionable Condition Register. The Data Questionable Modulation Status Group consists of the Data Questionable Modulation Condition Register, the Data Questionable Modulation Negative Transition Filter, the Data Questionable Modulation Positive Transition Filter, the Data Questionable Modulation Event Register, and the Data Questionable Modulation Event Enable Register. The Data Questionable Modulation Condition Register contains the following bits:



**Data Questionable Modulation Condition Register**

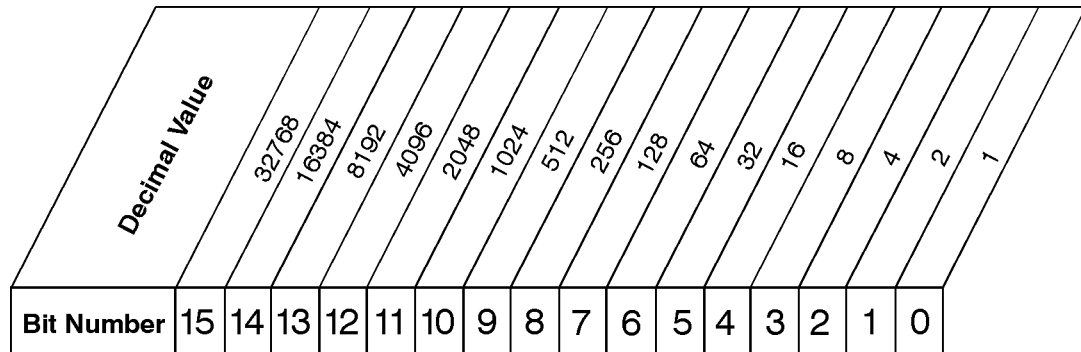
ck737a

Bit	Description
0	A 1 in this bit indicates that the External 1 input, AC coupling on, is less than 0.97 volts.
1	A 1 in this bit indicates that the External 1 input, AC coupling on, is more than 1.03 volts.
2	A 1 in this bit indicates that the External 2 input, AC coupling on, is less than 0.97 volts.
3	A 1 in this bit indicates that the External 2 input, AC coupling on, is more than 1.03 volts.
4-14	Unused. These bits are always set to 0.
15	Always Zero (0).

The Data Questionable Modulation Condition Register continuously monitors the modulation status of the instrument. Condition registers are read-only. To query the condition register, send the command **STATus:QUESTionable:MODulation:CONDition?** The response will be the *decimal* sum of the bits which are set to 1.

The transition filter specifies which types of bit state changes in the condition register will set corresponding bits in the event register. The changes may be positive (from 0 to 1) or negative (from 1 to 0). Send the command **STATus:QUESTionable:MODulation:NTRansition <num>** (negative) or **STATus:QUESTionable:MODulation:PTRansition <num>** (positive) where <num> is the sum of the decimal values of the bits you want to enable.

The Data Questionable Modulation Event Register latches transition events from the condition register as specified by the transition filters. Event registers are destructive read-only. Reading data from an event register will clear the content of that register. To query the event register, send the command **STATUS:QUESTIONABLE:MODULATION[:EVENT]?**



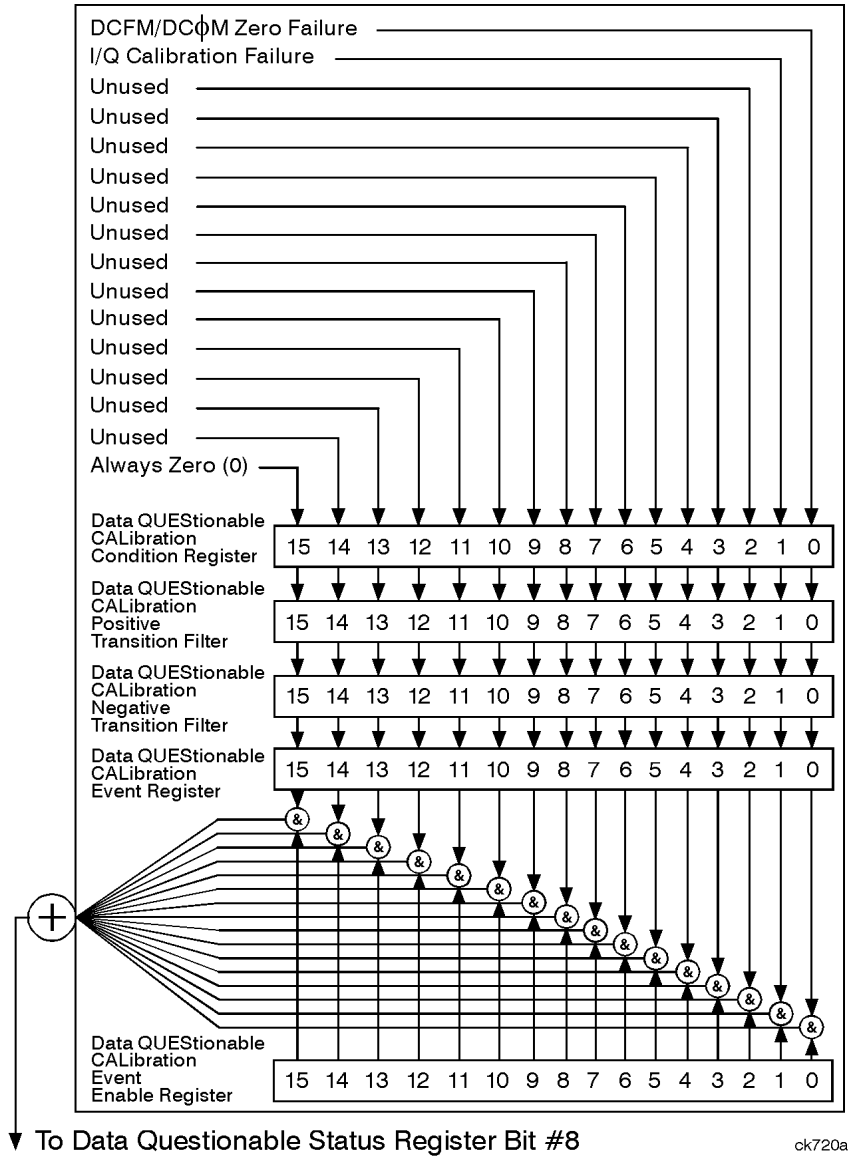
STATUS:QUESTIONABLE:MODULATION:ENABLE <num>  
 STATUS:QUESTIONABLE:MODULATION:ENABLE?

### Data Questionable Modulation Event Enable Register

ck738a

The Data Questionable Modulation Status Group also contains a Data Questionable Modulation Event Enable Register. This register lets you choose which bits in the Data Questionable Modulation Event Register will set the summary bit (bit 7 of the Data Questionable Condition Register) to 1. Send the **STATUS:QUESTIONABLE:MODULATION:ENABLE <num>** command where **<num>** is the sum of the decimal values of the bits you want to enable. For example, to enable bit 9 and bit 3 so that whenever either of those bits is set to 1, the Data Questionable Modulation summary bit of the Data Questionable Condition Register will be set to 1, send the command **STAT:QUES:MOD:ENAB 520** (512 + 8). The command **STATUS:QUESTIONABLE:MODULATION:ENABLE?** returns the decimal value of the sum of the bits previously enabled with the **STATUS:QUESTIONABLE:MODULATION:ENABLE <num>** command.

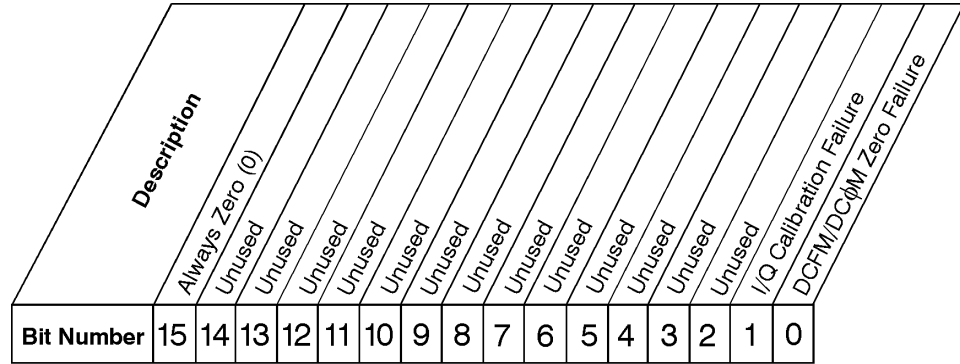
**Data Questionable Calibration Status Group**



**Figure 2-23. Data Questionable Calibration Status Group**

The Data Questionable Calibration Status Group is used to determine the specific event that set bit 8 in the Data Questionable Condition Register. The Data Questionable Calibration Status Group consists of the Data Questionable Calibration Condition Register, the Data Questionable Calibration Negative Transition

Filter, the Data Questionable Calibration Positive Transition Filter, the Data Questionable Calibration Event Register, and the Data Questionable Calibration Event Enable Register. The Data Questionable Calibration Condition Register contains the following bits:



STATus:QUESTionable:CALibration:ENABLE <num>  
STATus:QUESTionable:CALibration:ENABLE?

**Data Questionable Calibration Condition Register**

ck740a

Bit	Description
0	A 1 in this bit indicates that the DCFM/DCΦM zero calibration routine has failed. This is a critical error. The output of the source has no validity until the condition of this bit is 0.
1	A 1 in this bit indicates that the I/Q calibration routine has failed. An I/Q calibration failure does not affect the validity of the source output.
2-14	Unused. These bits are always set to 0.
15	Always Zero (0).

The Data Questionable Calibration Condition Register continuously monitors the calibration status of the instrument. Condition registers are read-only. To query the condition register, send the command **STATus:QUESTionable:CALibration:CONDition?** The response will be the *decimal* sum of the bits which are set to 1.

The transition filter specifies which types of bit state changes in the condition register will set corresponding bits in the event register. The changes may be positive (from 0 to 1) or negative (from 1 to 0). Send the command **STATus:QUESTionable:CALibration:NTRansition <num>** (negative) or **STATus:QUESTionable:CALibration:PTRansition <num>** (positive) where <num> is the sum of the decimal values of the bits you want to enable.

The Data Questionable Calibration Event Register latches transition events from the condition register as specified by the transition filters. Event registers are destructive read-only. Reading data from an event register will clear the content of that register. To query the event register, send the command **STATUS:QUESTIONABLE:CALibration[:EVENT]?**

<b>Decimal Value</b>																	
	32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1	
<b>Bit Number</b>	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	

STATUS:QUESTIONABLE:CALibration:ENABLE < num >  
 STATUS:QUESTIONABLE:CALibration:ENABLE?

### Data Questionable Calibration Event Enable Register

ck741a

The Data Questionable Calibration Status Group also contains a Data Questionable Calibration Event Enable Register. This register lets you choose which bits in the Data Questionable Calibration Event Register will set the summary bit (bit 8 of the Data Questionable Condition register) to 1. Send the **STATUS:QUESTIONABLE:CALibration:ENABLE <num>** command where <num> is the sum of the decimal values of the bits you want to enable. For example, to enable bit 9 and bit 3 so that whenever either of those bits is set to 1, the Data Questionable Calibration summary bit of the Data Questionable Condition Register will be set to 1, send the command **STAT:QUES:CAL:ENAB 520** (512 + 8). The command **STATUS:QUESTIONABLE:CALibration:ENABLE?** returns the decimal value of the sum of the bits previously enabled with the **STATUS:QUESTIONABLE:CALibration:ENABLE <num>** command.

---

## Advanced Programming Information

This section provides advanced programming information for applications requiring special techniques.

### Sending BREAK Messages

A BREAK is a special character transmission that usually indicates a change in operating conditions. Interpretation of break messages varies with the application. To send a break message, send a non-zero value to Control Register 1 as follows (Sc is the interface select code):

1640 CONTROL Sc,1;1      ! Send a Break to peripheral.

### Using the Modem Control Register

Control Register 5 controls various functions related to modem operation. Bits 0 through 3 control modem lines, and bit 4 enables a self-test loopback configuration.

#### Modem Handshake Lines (RTS and DTR)

Request-to-send and Data Terminal Ready lines are set or cleared at the beginning and end of each OUTPUT or ENTER operation. In some cases, it may be advantageous or necessary to maintain either or both in an active state. This is done by setting bit 1 or 0 respectively in Control Register 5 as follows:

1650 CONTROL Sc,5;2      ! Set RTS line only and hold active.  
1660 CONTROL Sc,5;1      ! Set DTR line only and hold active.  
1670 CONTROL Sc,5;3      ! Set DTR line only and hold active.  
1680 CONTROL Sc,5;0      ! Set both RTS and DTR lines active.

When RTS or DTR are set by Control Register 5, they are not toggled during OUTPUT or ENTER operations, but remain constantly in an active state until the CONTROL register is cleared by:

- writing a different value to CONTROL register 5.
- an interface reset to CONTROL register 0.
- an interface reset (Reset) from the keyboard (Shift Break on an ITF keyboard, or SHIFT PAUSE on a 98203 keyboard).

### Programming the DRS Modem Line

Bit 2 of Control Register 5 controls the present state of the Data Rate Select (DRS). When bit 2 is set, the modem line is activated. When bit 2 is cleared, the modem line is cleared. To set the DRS line, the following statement or its equivalent can be used:

```
1690 CONTROL Sc,5;4      ! Sets the DRS line.
```

This line is also cleared by a CONTROL statement to Control Register 5 with bit 2 cleared, or by an interface reset.

---

## 3 Programming Examples

---

This chapter provides several example programs to help you understand how the general SCPI concepts presented in Chapter 2 apply to programming real measurements.



## Using the Example Programs

The example programs are interactive. They require active participation by the operator. To gain an understanding of the principles without following all of the instructions, read the “Program Comments” sections to follow the programmed activity.

The HP-IB select code is assumed to be preset to 7. All example programs in this section expect the signal generator’s HP-IB address to be decimal 19.

To verify the signal generator’s current HP-IB address, press **Utility, HP-IB/RS-232, HP-IB Address**.

The active entry area indicates the present HP-IB address. If the number displayed is not 19, press **19** on the numeric keypad and then press **Enter** to set the address to 19.

Now check that the interface language is set to SCPI. Press **Utility, HP-IB/RS-232, Remote Language**. The selected interface language is then shown. If the instrument’s remote language is not set to SCPI, press **Utility, HP-IB/RS-232, Remote Language, SCPI**.

---

## HP-IB Check, Example Program 1

Verify that the remote annunciator (R) is activated on the signal generator's display. If it is not, verify that the signal generator's address is set to 19 and that the interface cable is properly connected.

If the controller display indicates an error message, it is possible that the program was typed incorrectly. If the controller accepts the REMOTE statement but the signal generator's remote annunciator does not appear on the display, refer to the service guide for troubleshooting information.

This program verifies that the HP-IB connections and interface are functional. Connect a controller to the signal generator using an HP-IB cable.

CLEAR and RESET the controller and type in the following commands and RUN the program:

```
10      !*****
20      !
30      !  PROGRAM NAME:      HPIB_CK                      Rev.  2A796
40      !
50      !  PROGRAM DESCRIPTION:  This program verifies that the HP-IB connections and
60      !                          interface are functional.
70      !
80      !                          Connect a controller to the signal generator using an
90      !                          HP-IB cable.
100     !
110     !  CLEAR and RESET the controller and type in the following commands and RUN
120     !  the program:
130     !
140     !*****
150     !
160     Sig_gen=719
170     LOCAL Sig_gen
180     CLEAR Sig_gen
190     CLEAR SCREEN
200     OUTPUT Sig_gen;"*RST"
210     REMOTE Sig_gen
220     PRINT "The signal generator should now be in REMOTE."
230     PRINT
240     PRINT "Verify that the remote [R] annunciator is on.  Press the 'Local' key, "
```

Programming Examples  
HP-IB Check, Example Program 1





```
250 PRINT "on the front panel, to return the signal generator to LOCAL control."  
260 PRINT  
270 PRINT "Press RUN to start again."  
280 END
```

### Program Comments

10 to 150:	Title and program description
160:	Set up a variable to contain the HP-IB address of the source.
170:	Place the signal generator into LOCAL mode.
180:	Reset the signal generator's parser and clear any pending output from the source.
190:	Clear the controller's display.
200:	Set the signal generator to a defined state.
210:	Set the signal generator to REMOTE mode.
220 to 270:	Print messages to the controller's display.
280:	End the program.

---

## Local Lockout Demonstration, Example Program 2

When the signal generator is in REMOTE mode, all the front panel keys are disabled except for the **Local**, , and  keys. But, when the LOCAL LOCKOUT command is sent, the **Local** key also is disabled and only  and  are allowed. The LOCAL command, executed from the controller, is then the only way to return all (or selected) instruments to front-panel control.

CLEAR and RESET the controller, type the following commands, and RUN the program.

```
10  !*****
20  !
30  ! PROGRAM NAME:                LCLLCK_OT                Rev.  2A796
40  !
50  ! PROGRAM DESCRIPTION:  When the signal generator is in REMOTE mode, all
60  !                        functional front panel keys are disabled except for
70  !                        the Local and Contrast keys.  But, when the LOCAL LOCKOUT
80  !                        command is set on the bus, even the Local key is disabled.
90  !                        The LOCAL command, executed from the controller, is then
100 !                        the only way to return the signal generator to front panel,
110 !                        LOCAL, control.
120 !
130 ! Continue by building upon Example Program #1; HP-IB CHECK:
140 !
150 !*****
160 !
170 Sig_gen=719
180 CLEAR Sig_gen
190 LOCAL Sig_gen
200 REMOTE Sig_gen
210 CLEAR SCREEN
220 OUTPUT Sig_gen;"*RST"
230 PRINT "The signal generator should now be in REMOTE."
240 PRINT
250 PRINT "Verify that all keys are ignored, except the 'Local' and 'Contrast' keys."
260 PRINT
270 PRINT "Verify that pressing 'Local' causes the remote [R] annunciator to go OFF."
280 PRINT
```

Programming Examples  
 Local Lockout Demonstration, Example Program 2

```

290 PRINT "..... Press Continue"
300 PAUSE
310 REMOTE Sig_gen
320 LOCAL LOCKOUT 7
330 PRINT
340 PRINT "Signal generator should now be in LOCAL LOCKOUT mode."
350 PRINT
360 PRINT "Verify that all keys including 'Local' (except Contrast keys) have no ef-
fect."
370 PRINT
380 PRINT "..... Press Continue"
390 PAUSE
400 PRINT
410 LOCAL 7
420 PRINT "Signal generator should now be in LOCAL mode."
430 PRINT
440 PRINT "Verify that the signal generator's front-panel keyboard is functional."
450 PRINT
460 PRINT "Press RUN to start again."
470 END
  
```

**Program Comments**

10 to 160:	Title and program description
170:	Set up a variable to contain the HP-IB address of the source.
180:	Reset the signal generator's parser and clear any pending output from the source.
190:	Place the signal generator into LOCAL mode.
200:	Set the signal generator to REMOTE mode.
210:	Clear the controller's display.
220:	Set the signal generator to a defined state.
230 to 300:	Print messages on the computer's display, then pause.
310:	Place the signal generator into REMOTE mode.
320:	Place the signal generator into LOCAL LOCKOUT mode.
330 to 400:	Print a message on the computer's display, then pause.

410:	Return the signal generator to LOCAL control.
420 to 460:	Print messages on the computer's display.
470:	End the program.

---

## Using Queries, Example Program 3

In this example, query commands are used with response data formats.

CLEAR and RESET the controller and run the following program:

```
10  !*****
20  !
30  ! PROGRAM NAME:   QUERIES                               Rev. 2A796
40  !
50  ! PROGRAM DESCRIPTION:  In this example, query commands are used with response
60  !                       data formats.
70  !
80  ! CLEAR and RESET the controller and RUN the following program:
90  !
100 !*****
110 !
120 DIM A$(10),C$(100),D$(10)
130 INTEGER B
140 Sig_gen=719
150 LOCAL Sig_gen
160 CLEAR Sig_gen
170 CLEAR SCREEN
180 OUTPUT Sig_gen;"*RST"
190 OUTPUT Sig_gen;"FREQ:CW?"
200 ENTER Sig_gen;F
210 PRINT "Present source CW frequency is: ";F/1.E+6;"MHz"
220 PRINT
230 OUTPUT Sig_gen;"POW:AMPL?"
240 ENTER Sig_gen;W
250 PRINT "Current power setting is: ";W;"dBm"
260 PRINT
270 OUTPUT Sig_gen;"FREQ:MODE?"
280 ENTER Sig_gen;A$
290 PRINT "Source's frequency mode is: ";A$
300 PRINT
310 B=0
```

```

320 !OUTPUT Sig_gen;"OUTP:MOD OFF"
330 OUTPUT Sig_gen;"OUTP:MOD:STAT?"
340 ENTER Sig_gen;B
350 IF B>0 THEN
360     PRINT "Modulation is: ON"
370 ELSE
380     PRINT "Modulation is: OFF"
390 END IF
400 OUTPUT Sig_gen;"*IDN?"
410 ENTER Sig_gen;C$
420 PRINT
430 PRINT "This signal generator is a ";C$
440 PRINT
450 OUTPUT Sig_gen;"SYST:COMM:GPIB:ADDR?"
460 ENTER Sig_gen;D$
470 PRINT "The HPIB Address is ";D$
480 PRINT
490 PRINT "Press the 'Local' key to return the instrument to LOCAL control"
500 PRINT "or -- Press RUN to start again."
510 END

```

### Program Comments

10 to 110:	Title and program description
120:	Dimension string A\$, string C\$, and string D\$ variables to contain responses.
130:	Declare variable B as an integer.
140:	Assign the signal generator's HP-IB address to a variable.
150:	Places the signal generator into LOCAL mode.
160:	Reset the signal generator's parser and clear any pending output from the source.
170:	Clear the controller's display.
180:	Set the signal generator to a defined state.
190:	Query the value of the signal generator's CW frequency.
200:	Enter the query response into the variable F. The response is always returned in fundamental units (Hz, in the case of frequency).



Programming Examples  
Using Queries, Example Program 3

210 to 220:	Print the instrument's current source frequency (in MHz) to the controller's display.
230:	Query the value of the signal generator's CW power.
240:	Enter the query response into the variable W.
250 to 260:	Print the instrument's current source power (in dBm) to the controller's display.
270:	Query the value of a discrete function, <code>FREQ:MODE</code> .
280:	Enter the response into A\$. The response will be a string that represents the function's present state.
290 to 300:	Print the value of A\$ on the computer's display.
310:	Assign a value of 0 to B variable.
320:	Commented command for turning off modulation.
330:	Query the on/off state for modulation.
340:	Enter value for the modulation state.
350 to 390:	Determine the on/off modulation state and display the results on the controller's display.
400:	Query the signal generator's identity.
410:	Enter the response into C\$. The response will be a string that represents the signal generator's model and options.
420 to 440:	Print the value of C\$ (the signal generator's model and options) on the computer's display.
450:	Query the signal generator's HP-IB address.
460:	Enter the response into D\$. The response will be a string that represents the signal generator's HP-IB address.
470 to 480:	Print the value of D\$ (the signal generator's HP-IB address) on the computer's display.
490 to 500:	Print messages to the controller's display.
510:	End the program.

---

## Generating a CW Signal, Example Program 4

In this example, a CW signal is generated at a frequency of 500 kHz with a power level of -2.1 dBm. CLEAR and RESET the controller and then run the following program:

```
10  !*****
20  !
30  ! PROGRAM NAME:   SIGNAL_CW                      Rev.  2A796
40  !
50  ! PROGRAM DESCRIPTION:  In this example, a CW signal is generated at a
60  !                       frequency of 500 kHz with a power level of -2.1 dBm.
70  !
80  ! CLEAR and RESET the controller and type in the following commands and RUN
90  ! the program:
100 !
110 !*****
120 !
130 Sig_gen=719
140 LOCAL Sig_gen
150 CLEAR Sig_gen
160 CLEAR SCREEN
170 OUTPUT Sig_gen;"*RST"
180 OUTPUT Sig_gen;"FREQ 500 kHz"
190 PRINT
200 OUTPUT Sig_gen;"POW:AMPL -2.1 dBm"
210 PRINT
220 OUTPUT Sig_gen;"OUTP:STAT ON"
230 PRINT
240 LOCAL Sig_gen
250 PRINT "Press RUN to start again."
260 END
```

### Program Comments

10 to 120:	Title and program description
130:	Assign the signal generator's HP-IB address to a variable.

Programming Examples  
Generating a CW Signal, Example Program 4

140:	Places the signal generator into LOCAL mode.
150:	Reset the signal generator's parser and clear any pending output from the source.
160:	Clear the controller's display.
170:	Set the signal generator to a defined state for programming.
180 to 190:	Set the frequency to 500 kHz.
200 to 210:	Set the signal generator's power level to -2.1 dBm.
220 to 230:	Turn the RF output on.
240:	Return the signal generator to LOCAL mode.
250:	Print a message to the controller's display.
260:	End the program.

---

## Generating an AC-Coupled External FM Signal, Example Program 5

In this example, an AC-coupled FM signal will be generated at a carrier frequency of 700 MHz with a power level of -25 dBm and a deviation of 20 kHz. In order to accomplish this, connect the output of a modulating signal source to the signal generator's EXT 2 connector, set the modulation signal source for the desired FM characteristics, and then run the following program:

```
10      !*****
20      !
30      ! PROGRAM NAME:      EXT_FM      AC-COUPLED                      Rev. 2A796
40      !
50      ! PROGRAM DESCRIPTION:  In this example, an AC-coupled FM signal will be
60      !                       generated at a carrier frequency of 700 MHz with a
70      !                       power level of -25 dBm and a deviation of 20 kHz.
80      !                       In order to accomplish this, connect the output of a
90      !                       modulating signal source to the signal generator's
100     !                       EXTERNAL 2 INPUT connector.
110     !
120     !                       Set the modulation signal source for the desired FM
130     !                       rate and correct 1 Vpk amplitude (HI/LO indication).
140     !
150     ! CLEAR and RESET the controller and type in the following commands and RUN
160     ! the program:
170     !
180     !*****
190     !
200     Sig_gen=719
210     LOCAL Sig_gen
220     CLEAR Sig_gen
230     CLEAR SCREEN
240     OUTPUT Sig_gen;"*RST"
250     OUTPUT Sig_gen;"FM:SOUR EXT2"
260     OUTPUT Sig_gen;"FM:EXT2:COUP AC"
270     OUTPUT Sig_gen;"FM:DEV 20 kHz"
280     OUTPUT Sig_gen;"FREQ 700 MHz"
290     OUTPUT Sig_gen;"POW:AMPL -25 dBm"
```

## Programming Examples

### Generating an AC-Coupled External FM Signal, Example Program 5

```
300 OUTPUT Sig_gen;"FM:STAT ON"  
310 OUTPUT Sig_gen;"OUTP:STAT ON"  
320 PRINT  
330 PRINT "Press RUN to start again."  
340 LOCAL Sig_gen  
350 END
```

### Program Comments

10 to 190:	Title and program description
200:	Assign the signal generator's HP-IB address to a variable.
210:	Places the signal generator into LOCAL mode.
220:	Reset the signal generator's parser and clear any pending output from the source.
230:	Clear the controller's display.
240:	Set the signal generator to a defined state.
250:	Set External 2 source to frequency modulation.
260:	Set FM path 2 coupling to AC.
270:	Set FM path 2 deviation to 20 kHz.
280:	Set the carrier frequency to 700 MHz.
290:	Set the carrier output power level to -25 dBm.
300:	Turn on the frequency modulation.
310:	Turn on the RF output.
320 to 330:	Print a message to the controller's display.
340:	Places the signal generator into LOCAL mode.
360:	End the program.

---

## Generating an AC-Coupled Internal FM Signal, Example Program 6

In this example, an AC-coupled internal FM signal will be generated at a carrier frequency of 900 MHz with a power level of -15 dBm. The FM rate will be 5 kHz and the peak deviation will be 100 kHz. CLEAR and RESET the controller, type the following commands, and RUN the program:

```
10      !*****
20      !
30      ! PROGRAM NAME: INT_FM                      REV. 2A796
40      !
50      ! PROGRAM DESCRIPTION: In this example, an internal FM signal will be
60      !                       generated at a carrier frequency of 900 MHz with a
70      !                       power level of -15 dBm. The FM rate will be 5 kHz
80      !                       and the FM peak deviation will be 100 kHz.
90      !
100     ! CLEAR and RESET the controller and type in the following commands and RUN
110     ! the program:
120     !
130     !*****
140     !
150     Sig_gen=719
160     LOCAL Sig_gen
170     CLEAR Sig_gen
180     CLEAR SCREEN
190     OUTPUT Sig_gen;"*RST"
200     OUTPUT Sig_gen;"FM2:INT:FREQ 5 kHz"
210     OUTPUT Sig_gen;"FM2:DEV 100 kHz"
220     OUTPUT Sig_gen;"FREQ 900 MHz"
230     OUTPUT Sig_gen;"POW -15 dBm"
240     OUTPUT Sig_gen;"FM2:STAT ON"
250     OUTPUT Sig_gen;"OUTP:STAT ON"
260     PRINT
270     PRINT "Press RUN to start again."
280     LOCAL Sig_gen
290     END
```

### Program Comments

10 to 140:	Title and program description
150:	Assign the signal generator's HP-IB address to a variable.
160:	Places the signal generator into LOCAL mode.
170:	Reset the signal generator's parser and clear any pending output from the source.
180:	Clear the controller's display.
190:	Set the signal generator to a defined state for programming.
200:	Set the path 2 internal FM source rate to 5 kHz.
210:	Set the path 2 internal FM source deviation to 100 kHz.
220:	Set the signal generator's carrier frequency to 900 MHz.
230:	Set the signal generator's carrier power to -15 dBm.
240:	Turn on the frequency modulation.
250:	Turn on the signal generator's RF output.
260 to 270:	Print a message to the controller's display.
280:	Places all the signal generator into LOCAL mode.
300:	End the program.

---

## Generating a Step-Swept Signal, Example Program 7

In this example, the signal generator will be programmed to continuously step sweep a defined set of points from 500 MHz to 800 MHz and dwell 500 ms at each of the points. The signal generator will then be set to LOCAL mode to allow the user to make adjustments from the front panel.

CLEAR and RESET the controller, type the following commands, and RUN the program.

```
10      !*****
20      !
30      !  PROGRAM NAME:  STEP_SWEEP                      Rev. 2A796
40      !
50      !  PROGRAM DESCRIPTION:  In this example, the generator will be setup to
60      !                          continuously step sweep a defined set of points from
70      !                          500 MHz to 800 MHz and dwell 500 ms at each of the
80      !                          points.  The generator will then be set for local front
90      !                          panel control to allow the operator to make adjustments.
100     !
110    !  CLEAR and RESET the controller and type in the following commands and RUN
120    !  the program:
130    !
140    !*****
150    !
160 Sig_gen=719
170 LOCAL Sig_gen
180 CLEAR Sig_gen
190 CLEAR SCREEN
200 OUTPUT Sig_gen;"*RST"
210 OUTPUT Sig_gen;"*CLS"
220 OUTPUT Sig_gen;"FREQ:MODE LIST"
230 OUTPUT Sig_gen;"LIST:TYPE STEP"
240 OUTPUT Sig_gen;"FREQ:STAR 500 MHz"
250 OUTPUT Sig_gen;"FREQ:STOP 800 MHz"
260 OUTPUT Sig_gen;"SWE:POIN 10"
270 OUTPUT Sig_gen;"SWE:DWEL .5 S"
280 !OUTPUT Sig_gen;"INIT"          ! Used for single sweep
290 OUTPUT Sig_gen;"INIT:CONT ON"  ! ON for continuous sweep; OFF for single sweep
```



## Programming Examples

### Generating a Step-Swept Signal, Example Program 7

```
300 OUTPUT Sig_gen;"POW:AMPL -5 dBm"
310 OUTPUT Sig_gen;"OUTP:STAT ON"
320 PRINT "The signal generator is continuously step sweeping from"
330 PRINT "500 to 800 MHz. The generator is dwelling 500 ms per step"
340 PRINT "for the defined 10 steps."
350 WAIT 3
360 LOCAL Sig_gen
370 PRINT
380 PRINT "The signal generator is no longer in REMOTE."
390 PRINT
400 WAIT 3
410 PRINT "Press RUN to start again"
420 END
```

### Program Comments

10 to 150:	Title and program description
160:	Assign the signal generator's HP-IB address to a variable.
170:	Places the signal generator into LOCAL mode.
180:	Reset the signal generator's parser and clear any pending output from the source.
190:	Clear the controller's display.
200:	Set the signal generator to a defined state for programming.
210:	Clear the signal generator's Status Byte Register.
220:	Set the signal generator's frequency mode to LIST.
230:	Set the signal generator's LIST type to STEP.
240:	Set the signal generator's start frequency to 500 MHz.
250:	Set the signal generator's stop frequency to 800 MHz.
260:	Set the signal generator's frequency steps at 10 evenly-spaced points, starting at 500 MHz and ending at 800 MHz.
270:	Set the signal generator's output signal to dwell at each of the ten points of 500 ms.
280:	Continuously initialize the signal generator sweep.
290:	Set the signal generator to CONTINUOUS sweep mode.

300:	Set the signal generator's RF output to $-5$ dBm.
310:	Turn on the signal generator's RF output.
320 to 340:	Print a message to the controller's display.
350:	Wait 3 seconds.
360:	Return the signal generator to LOCAL mode.
370 to 390:	Print a message to the controller's display.
400:	Wait 3 seconds.
410:	Print a message to the controller's display.
420:	End the program.

---

## Generating an External DC-Coupled Pulse Modulated Signal, Example Program 8

In this example, a repetitive, externally-triggered, pulse-modulated signal will be generated at a carrier frequency of 5 MHz with a power level of -5 dBm. Connect an external pulse source to the EXT 2 INPUT on the signal generator and set the desired pulse characteristics.

CLEAR and RESET the controller, type the following commands and RUN the program.

```

10  !*****
20  !
30  ! PROGRAM NAME:    EXT_PULSE                      Rev.  2A796
40  !
50  ! PROGRAM DESCRIPTION:  In this example, a repetitive, externally-triggered,
60  !                        dc-coupled pulse modulated signal will be generated
70  !                        at a carrier frequency of 5 MHz with a power level of
80  !                        -5 dBm.  Connect an external pulse source to the EXT 2
90  !                        INPUT on the signal generator and set the desired pulse
100 !                        characteristics.
110 !
120 ! CLEAR and RESET the controller and type in the following commands and RUN
130 ! the program:
140 !
150 !*****
160 !
170 Sig_gen=719
180 LOCAL Sig_gen
190 CLEAR Sig_gen
200 CLEAR SCREEN
210 OUTPUT Sig_gen;"*RST"
220 OUTPUT Sig_gen;"PULM:SOUR EXT2"
230 OUTPUT Sig_gen;"PULM:STAT ON"
240 OUTPUT Sig_gen;"FREQ 5 MHz"
250 OUTPUT Sig_gen;"POW:ALC OFF"
260 OUTPUT Sig_gen;"POW:AMPL -5 dBm"
270 OUTPUT Sig_gen;"POW:ALC:SEAR ON"
280 OUTPUT Sig_gen;"OUTP:STAT ON"

```

Programming Examples  
Generating an External DC-Coupled Pulse Modulated Signal, Example Program 8

```
290 PRINT
300 PRINT "Press RUN to start again."
310 LOCAL Sig_gen
320 END
```

### Program Comments

10 to 160:	Title and program description
170:	Assign the signal generator's HP-IB address to a variable.
180:	Places the signal generator into LOCAL mode.
190:	Reset the signal generator's parser and clear any pending output from the source.
200:	Clear the controller's display.
210:	Set the signal generator to a defined state for programming.
220:	Set the signal generator's pulse modulation source to EXT2.
230:	Turn pulse modulation on.
240:	Set the carrier frequency to 5 MHz.
250:	Turn off the power leveling search to assist in maintaining the required power level.
260:	Set the output power level to -5 dBm.
270:	Turn on the power leveling search to assist in maintaining the required power level.
280:	Turn the RF output on.
290 to 300:	Print a message to the controller's display.
310:	Place the signal generator into LOCAL mode.
320:	End the program.

---

## Saving and Recalling States, Example Program 9

In this example, instrument settings are saved in the signal generator's registers. These settings can then be recalled separately; either from the keyboard or from the source's front panel.

CLEAR and RESET the controller, type the following commands, and RUN the program:

```
10  !*****
20  !
30  ! PROGRAM NAME:   REG_SAV                      Rev.  2A796
40  !
50  ! PROGRAM DESCRIPTION:  In this example, instrument settings are saved in the
60  !                       instrument's registers.  These settings can then be
70  !                       recalled separately either from the keyboard or from
80  !                       the front panel of the instrument.
90  !
100 ! CLEAR and RESET the controller and type in the following commands and RUN
110 ! the program:
120 !
130 !*****
140 !
150 DIM Clear$[10]
160 X=0
170 Sig_gen=719
180 LOCAL Sig_gen
190 CLEAR Sig_gen
200 CLEAR SCREEN
210 OUTPUT Sig_gen;"*RST"
220 OUTPUT Sig_gen;"*CLS"
230 ! *****
240 Sig_in:  !
250 REPEAT
260     X=X+1
270     PRINT
280     PRINT "Configure the INSTRUMENT for the settings to be SAVED."
290     LOCAL Sig_gen
300     PRINT "when the setup is complete, Press Continue..."
```

```
310     PAUSE
320     CLEAR SCREEN
330     OUTPUT Sig_gen;"*SAV ";X
340     INPUT "Are there anymore setups to be SAVED?  Yes/No",No$
350     No$=UPC$(No$[1,1])
360     UNTIL No$="N"
370     WAIT 1
380     PRINT
390     PRINT "You have saved";X;"setups."
400     OUTPUT Sig_gen;"*RST"
410     ! *****
420 Sig_out:  !
430     INPUT "ENTER the Register number to be RECALLED or ENTER 0 to exit.",Reg1
440     CLEAR SCREEN
450     IF Reg1=0 THEN
460         PRINT
470         PRINT "You have requested to exit the program -- the program has been terminated."
480         GOTO Ins_lcl
490     END IF
500     IF Reg1>X THEN
510         PRINT
520         PRINT "You have requested a Register number not recognized by this executive."
530         GOTO Ins_lcl
540     END IF
550     OUTPUT Sig_gen;"*RCL ";Reg1
560     CLEAR SCREEN
570     PRINT
580     PRINT "The instrument has been set to the values from Registrar";Reg1;". "
590     GOTO Sig_out
600     ! *****
610 Ins_lcl:  !
620     LOCAL Sig_gen
630     PRINT
640     WAIT 1
650     PRINT "The instrument has been returned to local control...Press RUN to start
again."
660     END
```

### Program Comments

10 to 140:	Title and program description
150:	Dimension string Clear\$ to contain responses.
160:	Assigns a value of 0 to X variable.
170:	Assign the signal generator's HP-IB address to a variable.
180:	Places the signal generator into LOCAL mode.
190:	Reset the signal generator's parser and clear any pending output from the source.
200:	Clear the controller's display.
210:	Set the signal generator to a defined state for programming.
220:	Clear the signal generator's Status Byte Register.
230:	Program border.
240 to 360:	Subroutine: Sig_in. Allows the operator to configure the measurement and store the settings in save/recall registers. The subroutine will display the total number of instrument settings that were saved.
370:	Wait one second.
380 to 390:	Print a message on the computer's display.
400:	Set the signal generator to a defined state for programming.
410:	Program border
420 to 540:	Subroutine: Sig_out. Assigns keyboard values to the program variable. The subroutine includes conditional statements that will terminate the program depending on the value stored in the program variable Reg1 or transfers execution to a specified program.
550:	Recalls contents of register 1.
560:	Clear the controller's display.
570 to 580:	Print a message to the controller's display.
590:	Transfer program to a specific line.
600:	Program border
610:	Subroutine: Ins_lcl
620 to 630:	Return the signal generator to LOCAL mode.
640:	Wait one second.

650:	Print a message to the controller's display.
660:	End the program.



---

## Reading the Status Byte, Example Program 10

The following example reads the source's status byte and checks for certain conditions: in this case, an unlevelled output condition and an undermodulated output condition are created, and the Status Byte Register is read by the program to determine the questionable condition.

Unlevelled or undermodulated conditions are easy to produce from the front panel. Follow the directions below:

- To create an unlevelled output condition:
  1. Set the source's output amplitude to +20 dBm.
  2. Set the source's output frequency to the maximum value.
  3. Turn on the source's RF output.
  4. Check the source's display for the UNLEVELD annunciator.
- To create an undermodulated output condition:
  1. Select a modulation menu (AM, FM, Phase Modulation).
  2. Select an AC-coupled external source (i.e. Ext 1 AC-Coupled).
  3. Turn on the modulation.
  4. Do not connect any input to the selected external input.
  5. Check the source's display for the UNDERMOD annunciator.

CLEAR and RESET reset the controller, type the following commands, and RUN the program:

```
10      !*****
20      !
30      !  Read Status Byte Example
40      !
50      !*****
60      Sig_gen=719
70      CLEAR Sig_gen
80      CLEAR SCREEN
90      OUTPUT Sig_gen;"*CLS"
100     IF FNStat_con(Sig_gen,"UNLEVELED") THEN PRINT "Sig Gen Power Unleveled"
110     IF FNStat_con(Sig_gen,"MOD 1 UNDERMOD") THEN PRINT "Sig Gen Mod 1 Undermod"
120     IF FNStat_con(Sig_gen,"MOD 1 OVERMOD") THEN PRINT "Sig Gen Mod 1 Overmod"
130     IF FNStat_con(Sig_gen,"MOD 2 UNDERMOD") THEN PRINT "Sig Gen Mod 2 Undermod"
140     IF FNStat_con(Sig_gen,"MOD 2 OVERMOD") THEN PRINT "Sig Gen Mod 2 Overmod"
150     IF FNStat_con(Sig_gen,"OVEN COLD") THEN PRINT "Sig Gen Oven Cold"
160     IF FNStat_con(Sig_gen,"SYNTH UNLOCKED") THEN PRINT "Sig Gen Synth Unlock"
```

```
170 IF FNStat_con(Sig_gen,"10 MHZ REF UNLCK") THEN PRINT "Sig Gen 10 MHz Ref Unlck"
180 IF FNStat_con(Sig_gen,"1 GHZ REF UNLCK") THEN PRINT "Sig Gen 1 GHz LO Unlck"
190 IF FNStat_con(Sig_gen,"BBG SYNTH UNLCK") THEN PRINT "Sig Gen BBG Synth Unlck"
200 !*****
210 END
220 !
230 !
240 Stat_con:DEF FNStat_con(Sig_gen,Condition_name$)
250     SELECT Condition_name$
260     CASE "MOD 1 UNDERMOD","MOD 1 OVERMOD","MOD 2 UNDERMOD","MOD 2 OVERMOD"
270         IF Condition_name$="MOD 1 UNDERMOD" THEN Bit_number=0
280         IF Condition_name$="MOD 1 OVERMOD" THEN Bit_number=1
290         IF Condition_name$="MOD 2 UNDERMOD" THEN Bit_number=2
300         IF Condition_name$="MOD 2 OVERMOD" THEN Bit_number=3
310         OUTPUT Sig_gen;"STAT:QUES:MOD:ENAB 32767"
320         OUTPUT Sig_gen;"STAT:QUES:MOD:COND?"
330         ENTER Sig_gen;Condition
340     CASE "UNLEVELED"
350         Bit_number=1
360         OUTPUT Sig_gen;"STAT:QUES:POW:ENAB 32767"
370         OUTPUT Sig_gen;"STAT:QUES:POW:COND?"
380         ENTER Sig_gen;Condition
390     CASE "OVEN COLD"
400         Bit_number=4
410         OUTPUT Sig_gen;"STAT:QUES:ENAB 32767"
420         OUTPUT Sig_gen;"STAT:QUES:COND?"
430         ENTER Sig_gen;Condition
440     CASE "SYNTH UNLOCKED","10 MHZ REF UNLCK","1 GHZ REF UNLCK","BBG SYNTH UNLCK"
450         IF Condition_name$="SYNTH UNLOCKED" THEN Bit_number=0
460         IF Condition_name$="10 MHZ REF UNLCK" THEN Bit_number=1
470         IF Condition_name$="1 GHZ REF UNLCK" THEN Bit_number=2
480         IF Condition_name$="BBG SYNTH UNLCK" THEN Bit_number=3
490         OUTPUT Sig_gen;"STAT:QUES:FREQ:ENAB 32767"
500         OUTPUT Sig_gen;"STAT:QUES:FREQ:COND?"
510         ENTER Sig_gen;Condition
520     CASE ELSE
530         PRINT "Error in FNStat_con! ";Condition_name$;" is an unknown condition"
```

## Programming Examples

### Reading the Status Byte, Example Program 10

```
540     PAUSE
550     END SELECT
560     State=BIT(Condition,Bit_number)
570     RETURN State
580     FNEND
```

### Program Comments

10-50:	Program title
60:	Assign the signal generator's HP-IB address to a variable.
70 to 80:	Reset the signal generator's parser, clear any pending output, and clear the controller's display.
90:	Reset the signal generator's Status Byte Register.
100:	Check function FNStat_con for an UNLEVELED condition. If the signal generator is unleveled, print a message to the controller's display.
110:	Check function FNStat_con for an External 1 Undermod condition. If the modulation level input at EXT 1 is low, print a message to the controller's display.
120:	Check function FNStat_con for an External 1 Overmod condition. If the modulation level input at EXT 1 is high, print a message to the controller's display.
130:	Check function FNStat_con for an External 2 Undermod condition. If the modulation level input at EXT 2 is low, print a message to the controller's display.
140:	Check function FNStat_con for an External 2 Overmod condition. If the modulation level input at EXT 2 is high, print a message to the controller's display.
150:	Check function FNStat_con for an OVEN COLD condition. If the internal 10 MHz Reference Standard Oven is cold, print a message to the controller's display. NOTE: This condition is valid only for instruments with Option 1E5 High Stability Timebase.
160:	Check function FNStat_con for a synthesizer unlocked condition. If unlocked, print a message to the controller's display.
170:	Check function FNStat_con for a 10 MHz reference unlocked condition. If unlocked, print a message to the controller's display.
180:	Check function FNStat_con for a 1 GHz reference unlocked condition. If unlocked, print a message to the controller's display.
190:	Check function FNStat_con for a baseband synthesizer unlocked condition. If unlocked, print a message to the controller's display. NOTE: This condition is valid only for instruments with Option 1EH.

200-230:	End conditional section.
240-250:	Define the function FNStat_con. This main program passes the signal generator's address and a string indicating what condition to check for to this section of the program. Select/Case statements are used to determine which Data Questionable Status Register is checked. The proper bit within the register is then enabled and read. The state of the bit is then returned to the program (lines 100 - 190).
260:	Selects cases dealing with the Data Questionable Modulation Status Group.
270-300:	Assigns values to the variable <Bit_number> according to the variable <Condition_name\$>.
310:	Enables all bits in the Data Questionable Modulation event register.
320-330:	Queries the Data Questionable Modulation Event Register and stores it in the variable <Condition>.
340:	Selects the case of an UNLEVELED condition.
350-380:	Assigns a value to <Bit_number>. Enables all bit in the Data Questionable Power Event Register, then queries the Data Questionable Event Register and stores it in the variable <Condition>.
390:	Selects cases dealing with an OVEN COLD condition. NOTE: This condition is valid only for instruments with Option 1E5 High Stability Timebase.
400-430:	Assigns a value to the variable <Bit_number>. Enables all bits in the Data Questionable Event Register, the queries the Data Questionable Event Register and stores it in the variable <Condition>.
440:	Selects cases dealing with the Data Questionable Frequency Status Group.
450-480:	Assigns a value to the variable <Bit_number> according to <Condition_name>
490:	Enables all bits in the Data Questionable Frequency Event Register.
500-510:	Queries the Data Questionable Frequency Event Register and stores it in the variable <Condition>.
520-540:	If <Condition_name\$> is unknown, it prints a message to the screen and pauses the program. Note that <Condition_name\$> is passed to the function from the main program.
560-570:	Returns the state of the bit corresponding to the condition queried to the main program. For example, if the source's output power was unleveled and an UNLEVELED condition was queried, then the value of Bit 0 (line 350) of the Data Questionable Power Status Group (line 370-380) would be returned.
580:	End the program.

---

## End of Sweep Service Request, Example Program 11

The following example provides a program that produces a service request to the controller when a specific condition (in this case: end of sweep) is present in the signal generator.

CLEAR and RESET reset the controller, type the following commands, and RUN the program:

```
10  !*****
20  !
30  ! End of Sweep Service Request Example
40  !
50  !*****
60  Sig_gen=719
70  CLEAR Sig_gen
80  CLEAR SCREEN
90  OUTPUT Sig_gen;"*RST"
100 OUTPUT Sig_gen;"*CLS"
110 !*****
120 Setup_srq:  !
130  Hpib=7
140  Mask=2
150  OUTPUT Sig_gen;"STAT:OPER:NTR 8"
160  OUTPUT Sig_gen;"STAT:OPER:PTR 0"
170  OUTPUT Sig_gen;"STAT:OPER:ENAB 8"
180  OUTPUT Sig_gen;"*SRE 128"
190  ON INTR Hpib GOSUB Service_routine
200  ENABLE INTR Hpib;Mask
210  !*****
220 Setup_swp:  !
230  OUTPUT Sig_gen;"FREQ:MODE LIST"
240  OUTPUT Sig_gen;"LIST:TYPE STEP"
250  OUTPUT Sig_gen;"LIST:TRIG:SOUR IMM"
260  OUTPUT Sig_gen;"LIST:MODE AUTO"
270  OUTPUT Sig_gen;"FREQ:STAR 40 MHZ"
280  OUTPUT Sig_gen;"FREQ:STOP 900 MHZ"
290  OUTPUT Sig_gen;"SWE:POIN 25"
300  OUTPUT Sig_gen;"SWE:DWEL .5 S"
```

```
310 OUTPUT Sig_gen;"INIT:CONT OFF"
320 OUTPUT Sig_gen;"TRIG:SOUR IMM"
330 Trigger_swp: !
340 OUTPUT Sig_gen;"INIT"
350 Sweep=1
360 !*****
370 Wait_4_swp_end: !
380 PRINT "Sweeping";
390 WHILE Sweep=1
400 PRINT " .";
410 WAIT 1
420 END WHILE
430 GOTO End
440 !
450 !*****
460 Service_routine: !
470 Ser_poll=SPOLL(Sig_gen)
480 IF BIT(Ser_poll,7) THEN
490 PRINT "The Sig Gen has completed a sweep"
500 Sweep=0
510 ELSE
520 PRINT "A Service Request was received for unknown reasons"
530 END IF
540 ENABLE INTR 7
550 RETURN
560 !*****
570 End: !
580 END
```

## Program Comments

10-50:	Program title
60:	Assign the signal generator's HP-IB address to a variable.
70:	Reset the signal generator's parser and clear any pending output from the source.
80:	Clear the computer's display.
90:	Set the signal generator to a defined state for programming.
100:	Clear the Status Byte; resets all registers to 0.
110:	Program border.
130 to 140:	Set variables Hpib and Mask so that BASIC will cause an interrupt when there is a service request on the HP-IB Interface. Refer to the HP BASIC Interface Reference for details.
150:	Configure the Operation Status Group Negative Transition Filter so that a negative transition in Bit 3 (Sweeping) of the Operation Status Group initiates a corresponding event in the Operation Event Register. This event occurs at the end of a sweep.
160:	Configure the Operation Status Group Positive Transition Filter so that no positive transitions in the Operation Status Group initiate a corresponding event in the Operation Event Register. In other words, a start of sweep does not initiate an event.
170:	Enable Operation Status Event Bit 3 (Sweeping) to report its event to the Summary Bit 7 of the Status Byte Register.
180:	Enable Summary Bit 7 of the Status Byte Register to generate a Service Request (SRQ) on HP-IB.
190 to 200:	Setup HP BASIC to recognize the SRQ and branch to "Service Routine" when the SRQ is received.
210:	Program border.
220-320:	Setup the signal generator for a 40 MHz to 900 MHz stepped single sweep.
330 to 350:	Trigger the sweep.
370 to 420:	An endless loop developed for the sake of this example. The program is stuck here forever or until the signal generator generates an SRQ at the end of sweep. Then an Interrupt is initiated and the program branches to Service_routine (Line 460).
470:	A serial poll is used to read the signal generator Status Byte.
480 to 490:	If Status Byte Bit 7 is true then the sweep is complete. Sets the value of the variable sweep to 0 so the program can exit the endless loop after it returns.

510 to 530:	Indicates that the SRQ cause is unknown.
540 to 550:	Re-enables the Interrupt in HP BASIC then returns to the endless loop. If sweep is still set to 1 the program is again stuck in the endless loop waiting for end-of-sweep. If an end-of-sweep was detected, the program ends.
570 to 580:	End the program.



Programming Examples  
End of Sweep Service Request, Example Program 11

---

## 4 Programming Command Cross Reference

---

This chapter provides tables that cross reference each of the SCPI commands to the corresponding front panel keys, and the HP 8656/57-compatible commands that are equivalent to SCPI commands.

## Front Panel Key Versus Command

Table 4-1. AM Softkeys

Key	SCPI Command
AM Depth	[:SOURce]:AM[1]2[:DEPTh] <val><PCT> [:SOURce]:AM[1]2[:DEPTh]?
AM Depth Couple Off On	[:SOURce]:AM[1]2[:DEPTh]:TRACk ON OFF 1 0 [:SOURce]:AM[1]2[:DEPTh]:TRACk?
AM Dual-Sine Ampl Ratio	[:SOURce]:AM[1]2:INTernal[1]:FREQuency:ALTErnate:AMPLitude:PERCent <val><unit> [:SOURce]:AM[1]2:INTernal[1]:FREQuency:ALTErnate:AMPLitude:PERCent?
AM Off On	[:SOURce]:AM[1]2:STATe ON OFF 1 0 [:SOURce]:AM[1]2:STATe?
AM Path 1 2 WB	[:SOURce]:AM:WIDeband:STATe ON OFF 1 0 [:SOURce]:AM:WIDeband:STATe?
AM Rate	[:SOURce]:AM[1]2:INTernal[1]:FREQuency <val><unit> [:SOURce]:AM[1]2:INTernal[1]:FREQuency?
AM Source	[:SOURce]:AM[1]2:SOURce?
AM Start Rate	[:SOURce]:AM[1]2:INTernal[1]:FREQuency <val><unit> [:SOURce]:AM[1]2:INTernal[1]:FREQuency?
AM Stop Rate	[:SOURce]:AM[1]2:INTernal[1]:FREQuency:ALTErnate <val><unit> [:SOURce]:AM[1]2:INTernal[1]:FREQuency:ALTErnate?
AM Sweep Time	[:SOURce]:AM[1]2:INTernal[1]:SWEep:TIME <val><unit> [:SOURce]:AM[1]2:INTernal[1]:SWEep:TIME?
AM Sweep Trigger	:TRIGger[:SEQuence]:SOURce?
AM Tone 1 Rate	[:SOURce]:AM[1]2:INTernal[1]:FREQuency <val><unit> [:SOURce]:AM[1]2:INTernal[1]:FREQuency?
AM Tone 2 Rate	[:SOURce]:AM[1]2:INTernal[1]:FREQuency:ALTErnate <val><unit> [:SOURce]:AM[1]2:INTernal[1]:FREQuency:ALTErnate?
AM Waveform	[:SOURce]:AM[1]2:INTernal[1]:FUNCTion SHAPE?
Bus	[:SOURce]:AM[1]2:INTernal[1]:SWEep:TRIGger BUS [:SOURce]:AM[1]2:INTernal[1]:SWEep:TRIGger?

**Table 4-1. AM Softkeys**

Key	SCPI Command
Dual-Sine	[:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe DUALsine [:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe?
Ext	[:SOURce]:AM[1]2:INTernal[1]:SWEep:TRIGger EXTernal [:SOURce]:AM[1]2:INTernal[1]:SWEep:TRIGger?
Ext 1 AC-Coupled	[:SOURce]:AM[1]2:SOURce EXTernal[1] [:SOURce]:AM[1]2:EXTernal[1]:COUPling AC [:SOURce]:AM[1]2:EXTernal[1]:COUPling?
Ext 1 DC-Coupled	[:SOURce]:AM[1]2:SOURce EXTernal[1] [:SOURce]:AM[1]2:EXTernal[1]:COUPling DC [:SOURce]:AM[1]2:EXTernal[1]:COUPling?
Ext 2 AC-Coupled	[:SOURce]:AM[1]2:SOURce EXTernal2 [:SOURce]:AM[1]2:EXTernal2:COUPling AC [:SOURce]:AM[1]2:EXTernal2:COUPling?
Ext 2 DC-Coupled	[:SOURce]:AM[1]2:SOURce EXTernal2 [:SOURce]:AM[1]2:EXTernal2:COUPling DC [:SOURce]:AM[1]2:EXTernal2:COUPling?
Immediate	[:SOURce]:AM[1]2:INTernal[1]:SWEep:TRIGger IMMEDIATE [:SOURce]:AM[1]2:INTernal[1]:SWEep:TRIGger?
Internal	[:SOURce]:AM[1]2:SOURce INT[1] [:SOURce]:AM[1]2:SOURce?
Noise	[:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe NOISE [:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe?
Ramp	[:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe RAMP [:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe?
Sine	[:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe SINE [:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe?
Square	[:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe SQUARE [:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe?
Swept-Sine	[:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe SWEPTsine [:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe?
Triangle	[:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe TRIangle [:SOURce]:AM[1]2:INTernal[1]:FUNcTion:SHApe?

**Table 4-1. AM Softkeys**

Key	SCPI Command
Trigger In Polarity Neg Pos	:TRIGger[:SEQuence]:SLOPe POSitive NEGative :TRIGger[:SEQuence]:SLOPe?
Trigger Key	[:SOURce]:AM[1]2:INTernal[1]:SWEep:TRIGger KEY [:SOURce]:AM[1]2:INTernal[1]:SWEep:TRIGger?
Trigger Out Polarity Neg Pos	:TRIGger:OUTPut:POLarity POSitive NEGative :TRIGger:OUTPut:POLarity?

**Table 4-2. Ampl Softkeys**

Key	SCPI Command
ALC BW Normal Narrow	[:SOURce]:DM:EXTernal:ALC:BWIDth BANDwidth NORMal NARRow [:SOURce]:DM:EXTernal:ALC:BWIDth BANDwidth?
ALC Off On	[:SOURce]:POWer:ALC[:STATe] ON OFF 1 0 [:SOURce]:POWer:ALC[:STATe]?
Ampl Offset	[:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet <val><unit> [:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet?
Ampl Ref Set	[:SOURce]:POWer:REFerence <val><unit> [:SOURce]:POWer:REFerence?
Ampl Ref Off On	[:SOURce]:POWer:REFerence:STATe ON OFF 1 0 [:SOURce]:POWer:REFerence:STATe?
Atten Hold Off On	[:SOURce]:POWer:ATTenuation:AUTO ON OFF 1 0 [:SOURce]:POWer:ATTenuation:AUTO?
Do Power Search	[:SOURce]:POWer:ALC:SEARch ONCE
Power Search Manual Auto	[:SOURce]:POWer:ALC:SEARch ON OFF 1 0 [:SOURce]:POWer:ALC:SEARch?

**Table 4-3. Amplitude Hardkey**

Key	SCPI Command
<b>Amplitude (hardkey)</b>	[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] <val><unit> [:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]?

**Table 4-4. Display Contrast Hardkeys**

Key	SCPI Command
<b>Display Contrast (hardkeys)</b>	:DISPlay:CONTrast <value> :DISPlay:CONTrast?

**Table 4-5. FM Softkeys**

Key	SCPI Command
Bus	[:SOURce]:FM[1] 2:INTernal[1]:SWEep:TRIGger BUS [:SOURce]:FM[1] 2:INTernal[1]:SWEep:TRIGger?
DCFM/DCΦM Cal	:CALibration:DCFM
Dual-Sine	[:SOURce]:FM[1] 2:INTernal[1]:FUNcTION:SHAPE DUALsine [:SOURce]:FM[1] 2:INTernal[1]:FUNcTION:SHAPE?
Ext	[:SOURce]:FM[1] 2:INTernal[1]:SWEep:TRIGger EXTernal [:SOURce]:FM[1] 2:INTernal[1]:SWEep:TRIGger?
Ext 1 AC-Coupled	[:SOURce]:FM[1] 2:SOURce EXTernal[1] [:SOURce]:FM[1] 2:EXTernal[1]:COUPling AC [:SOURce]:FM[1] 2:EXTernal[1]:COUPling?
Ext 1 DC-Coupled	[:SOURce]:FM[1] 2:SOURce EXTernal[1] [:SOURce]:FM[1] 2:EXTernal[1]:COUPling DC [:SOURce]:FM[1] 2:EXTernal[1]:COUPling?
Ext 2 AC-Coupled	[:SOURce]:FM[1] 2:SOURce EXTernal2 [:SOURce]:FM[1] 2:EXTernal2:COUPling AC [:SOURce]:FM[1] 2:EXTernal2:COUPling?
Ext 2 DC-Coupled	[:SOURce]:FM[1] 2:SOURce EXTernal2 [:SOURce]:FM[1] 2:EXTernal2:COUPling DC [:SOURce]:FM[1] 2:EXTernal2:COUPling?
FM Dev	[:SOURce]:FM[1] 2[:DEViation] <val><unit> [:SOURce]:FM[1] 2[:DEViation]?
FM Dev Couple Off On	[:SOURce]:FM[1] 2[:DEViation]:TRACk ON OFF 1 0 [:SOURce]:FM[1] 2[:DEViation]:TRACk?
FM Dual-Sine Ampl Ratio	[:SOURce]:FM[1] 2:INTernal[1]:FREQuency:ALTErnate:AMPLitude:PERCent <val><unit> [:SOURce]:FM[1] 2:INTernal[1]:FREQuency:ALTErnate:AMPLitude:PERCent?

**Table 4-5. FM Softkeys**

Key	SCPI Command
FM On Off	[:SOURce]:FM[1] 2:STATe ON OFF 1 0 [:SOURce]:FM[1] 2:STATe?
FM Rate	[:SOURce]:FM[1] 2:INTernal[1]:FREQuency <val><unit> [:SOURce]:FM[1] 2:INTernal[1]:FREQuency?
FM Source	[:SOURce]:FM[1] 2:SOURce?
FM Start Rate	[:SOURce]:FM[1] 2:INTernal[1]:FREQuency <val><unit> [:SOURce]:FM[1] 2:INTernal[1]:FREQuency?
FM Stop Rate	[:SOURce]:FM[1] 2:INTernal[1]:FREQuency:ALTErnate <val><unit> [:SOURce]:FM[1] 2:INTernal[1]:FREQuency:ALTErnate?
FM Sweep Time	[:SOURce]:FM[1] 2:INTernal[1]:SWEep:TIME <val><unit> [:SOURce]:FM[1] 2:INTernal[1]:SWEep:TIME?
FM Sweep Trigger	:TRIGger[:SEQuence]:SOURce?
FM Tone 1 Rate	[:SOURce]:FM[1] 2:INTernal[1]:FREQuency <val><unit> [:SOURce]:FM[1] 2:INTernal[1]:FREQuency?
FM Tone 2 Rate	[:SOURce]:FM[1] 2:INTernal[1]:FREQuency:ALTErnate <val><unit> [:SOURce]:FM[1] 2:INTernal[1]:FREQuency:ALTErnate?
FM Waveform	[:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE?
Immediate	[:SOURce]:FM[1] 2:INTernal[1]:SWEep:TRIGger IMMEDIATE [:SOURce]:FM[1] 2:INTernal[1]:SWEep:TRIGger?
Internal	[:SOURce]:FM[1] 2:SOURce INT[1] [:SOURce]:FM[1] 2:SOURce?
Noise	[:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE NOISE [:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE?
Ramp	[:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE RAMP [:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE?
Sine	[:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE SINE [:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE?
Square	[:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE SQUARE [:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE?
Swept-Sine	[:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE SWEPTSine [:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE?

**Table 4-5. FM Softkeys**

Key	SCPI Command
Triangle	[:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE TRIangle [:SOURce]:FM[1] 2:INTernal[1]:FUNCTion:SHAPE?
Trigger In Polarity Neg Pos	:TRIGger[:SEQuence]:SLOPe POSitive NEGative :TRIGger[:SEQuence]:SLOPe?
Trigger Key	[:SOURce]:FM[1] 2:INTernal[1]:SWEep:TRIGger KEY [:SOURce]:FM[1] 2:INTernal[1]:SWEep:TRIGger?
Trigger Out Polarity Neg Pos	:TRIGger:OUTPut:POLarity POSitive NEGative :TRIGger:OUTPut:POLarity?

**Table 4-6. Freq Softkeys**

Key	SCPI Command
Adjust Phase	[:SOURce]:PHASe:[ADJust] <val><unit> [:SOURce]:PHASe:[ADJust]?
Freq Multiplier	[:SOURce]:FREQuency:MULTIplier <val> [:SOURce]:FREQuency:MULTIplier?
Freq Offset	[:SOURce]:FREQuency:OFFset <val><unit> [:SOURce]:FREQuency:OFFset?
Freq Ref Off On	[:SOURce]:FREQuency:REFeRence:STATe ON OFF 1 0 [:SOURce]:FREQuency:REFeRence:STATe?
Freq Ref Set	[:SOURce]:FREQuency:REFeRence <val><unit> [:SOURce]:FREQuency:REFeRence?
Mode 1 Optimize <10kHz Offset	[:SOURce]:FREQuency:SYNTHeSis 1 [:SOURce]:FREQuency:SYNTHeSis?
Mode 2 Optimize >10kHz Offset	[:SOURce]:FREQuency:SYNTHeSis 2 [:SOURce]:FREQuency:SYNTHeSis?
Phase Ref Set	[:SOURce]:PHASe:REFeRence



**Table 4-7. Frequency Hardkey**

Key	SCPI Command
<b>Frequency (hardkey)</b>	[:SOURce]:FREQuency[:CW] <val><unit> [:SOURce]:FREQuency[:CW]? [:SOURce]:FREQuency:FIXed <val><unit> [:SOURce]:FREQuency:FIXed? [:SOURce]:FREQuency:MODE CW FIXed [:SOURce]:FREQuency:MODE?

**Table 4-8. LF Out Softkeys**

Key	SCPI Command
Bus	[:SOURce]:LFOutput:FUNCTion:SWEep:TRIGger BUS [:SOURce]:LFOutput:FUNCTion:SWEep:TRIGger?
DC	[:SOURce]:LFOutput:FUNCTion:SHAPE DC [:SOURce]:LFOutput:FUNCTion:SHAPE?
Dual-Sine	[:SOURce]:LFOutput:FUNCTion:SHAPE DUALsine [:SOURce]:LFOutput:FUNCTion:SHAPE?
Ext	[:SOURce]:LFOutput:FUNCTion:SWEep:TRIGger EXTernal [:SOURce]:LFOutput:FUNCTion:SWEep:TRIGger?
Function Generator	[:SOURce]:LFOutput:SOURce FUNCTion [:SOURce]:LFOutput:SOURce?
Immediate	[:SOURce]:LFOutput:FUNCTion:SWEep:TRIGger IMMEDIATE [:SOURce]:LFOutput:FUNCTion:SWEep:TRIGger?
Internal	[:SOURce]:LFOutput:SOURce INT[1] [:SOURce]:LFOutput:SOURce?
LF Out Amplitude	[:SOURce]:LFOutput:AMPLitude <val><unit> [:SOURce]:LFOutput:AMPLitude?
LF Out Freq	[:SOURce]:LFOutput:FUNCTion:FREQuency <val><unit> [:SOURce]:LFOutput:FUNCTion:FREQuency?
LF Out Dual-Sine Ampl Ratio	[:SOURce]:LFOutput:FUNCTion:FREQuency:ALTErnate:AMPLitude:PERCent <val><unit> [:SOURce]:LFOutput:FUNCTion:FREQuency:ALTErnate:AMPLitude:PERCent?
LF Out Off On	[:SOURce]:LFOutput:STATe ON OFF 1 0 [:SOURce]:LFOutput:STATe?

**Table 4-8. LF Out Softkeys**

Key	SCPI Command
LF Out Period	[:SOURce]:LFOutput:FUNCTion:PERiod <val><unit> [:SOURce]:LFOutput:FUNCTion:PERiod?
LF Out Source	[:SOURce]:LFOutput:SOURce?
LF Out Start Freq	[:SOURce]:LFOutput:FUNCTion:FREQUency <val><unit> [:SOURce]:LFOutput:FUNCTion:FREQUency?
LF Out Stop Freq	[:SOURce]:LFOutput:FUNCTion:FREQUency:ALTerate <val><unit> [:SOURce]:LFOutput:FUNCTion:FREQUency:ALTerate?
LF Out Sweep Time	[:SOURce]:LFOutput:FUNCTion:SWEep:TIME <val><unit> [:SOURce]:LFOutput:FUNCTion:SWEep:TIME?
LF Out Sweep Trigger	[:SOURce]:LFOutput:FUNCTion:SWEep:TRIGger IMMEDIATE BUS EXTernal KEY [:SOURce]:LFOutput:FUNCTion:SWEep:TRIGger?
LF Out Tone 1 Freq	[:SOURce]:LFOutput:FUNCTion:FREQUency <val><unit> [:SOURce]:LFOutput:FUNCTion:FREQUency?
LF Out Tone 2 Freq	[:SOURce]:LFOutput:FUNCTion:FREQUency:ALTerate <val><unit> [:SOURce]:LFOutput:FUNCTion:FREQUency:ALTerate?
LF Out Waveform	[:SOURce]:LFOutput:FUNCTion:SHAPE?
LF Out Width	[:SOURce]:LFOutput:FUNCTion:PWIDth <val><unit> [:SOURce]:LFOutput:FUNCTion:PWIDth?
Noise	[:SOURce]:LFOutput:FUNCTion:SHAPE NOISE [:SOURce]:LFOutput:FUNCTion:SHAPE?
Ramp	[:SOURce]:LFOutput:FUNCTion:SHAPE RAMP [:SOURce]:LFOutput:FUNCTion:SHAPE?
Sine	[:SOURce]:LFOutput:FUNCTion:SHAPE SINE [:SOURce]:LFOutput:FUNCTion:SHAPE?
Square	[:SOURce]:LFOutput:FUNCTion:SHAPE SQUARE [:SOURce]:LFOutput:FUNCTion:SHAPE?
Swept-Sine	[:SOURce]:LFOutput:FUNCTion:SHAPE SWEPTSINE [:SOURce]:LFOutput:FUNCTion:SHAPE?
Triangle	[:SOURce]:LFOutput:FUNCTion:SHAPE TRIANGLE [:SOURce]:LFOutput:FUNCTion:SHAPE?

**Table 4-8. LF Out Softkeys**

Key	SCPI Command
Trigger In Polarity Neg Pos	:TRIGger[:SEQuence]:SLOPe POSitive NEGative :TRIGger[:SEQuence]:SLOPe?
Trigger Key	[:SOURce]:LFOutput:FUNcTION:SWEep:TRIGger KEY [:SOURce]:LFOutput:FUNcTION:SWEep:TRIGger?
Trigger Out Polarity Neg Pos	:TRIGger:OUTPut:POLarity POSitive NEGative :TRIGger:OUTPut:POLarity?

**Table 4-9. Mod On/Off Hardkey**

Key	SCPI Command
<b>Mod On/Off (hardkey)</b>	:OUTPut:MODulation[:STATe] ON OFF 1 0 :OUTPut:MODulation[:STATe]?

**Table 4-10. Mode - NADC Softkeys**

Key	SCPI Command
4 1's & 4 0's	[:SOURce]:RADio[:NADC]:DATA P4 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom P4 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] P4 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom P4 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] P4
8 1's & 8 0's	[:SOURce]:RADio[:NADC]:DATA P8 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom P8 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] P8 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom P8 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] P8
16 1's & 16 0's	[:SOURce]:RADio[:NADC]:DATA P16 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom P16 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] P16 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom P16 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] P16

**Table 4-10. Mode - NADC Softkeys**

Key	SCPI Command
32 1's & 32 0's	[[:SOURce]:RADio[:NADC]:DATA P32 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom P32 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] P32 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom P32 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] P32
64 1's & 64 0's	[[:SOURce]:RADio[:NADC]:DATA P64 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom P64 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] P64 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom P64 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] P64
All Timeslots	[[:SOURce]:RADio[:NADC]:SOUT ALL
BBG Data Clock Ext Int	[[:SOURce]:RADio[:NADC]:BBCLock INT[1] EXTernal[1] [:SOURce]:RADio[:NADC]:BBCLock?
Begin Frame	[[:SOURce]:RADio[:NADC]:SOUT FRAME
Begin Pattern	[[:SOURce]:RADio[:NADC]:SOUT FRAME
Begin Timeslot	[[:SOURce]:RADio[:NADC]:SOUT SLOT [:SOURce]:RADio[:NADC]:SOUT:SLOT <value> [:SOURce]:RADio[:NADC]:SOUT:SLOT?
Bit Rate	[[:SOURce]:RADio[:NADC]:BRATE <value> [:SOURce]:RADio[:NADC]:BRATE?
Bus	[[:SOURce]:RADio[:NADC]:]TRIGger[:SOURce] BUS
CDVCC	[[:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel:CDVCcode <bit_pattern> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel:CDVCcode? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel:CDVCcode <bit_pattern> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel:CDVCcode?
Configure Down Custom	[[:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom?
Configure Up Custom	[[:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom?

**Table 4-10. Mode - NADC Softkeys**

Key	SCPI Command
Data	[[:SOURce]:RADio[:NADC]:DATA? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA]? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA]?]
Data Format Pattern Framed	[[:SOURce]:RADio[:NADC]:BURSt[:STATe] ON OFF 1 0 [:SOURce]:RADio[:NADC]:BURSt[:STATe]?]
Down Custom	[[:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6[:TYPE] DCUStom]
Down TCH	[[:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6[:TYPE] DTCH]
Down TCH All	[[:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6[:TYPE] DTCH_ALL]
Ext	[[:SOURce]:RADio[:NADC]:DATA EXTernal [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom EXTernal [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] EXTernal [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom EXTernal [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] EXTernal [:SOURce]:RADio[:NADC]:TRIGger[:SOURce] EXTernal]
Ext Data Clock Normal Symbol	[[:SOURce]:RADio[:NADC]:EDCLock SYMBol NORMal [:SOURce]:RADio[:NADC]:EDCLock?]
Ext Delay Bits	[[:SOURce]:RADio[:NADC]:TRIGger[:SOURce]:EXTernal:DELay <value> [:SOURce]:RADio[:NADC]:TRIGger[:SOURce]:EXTernal:DELay?]
Ext Delay Off On	[[:SOURce]:RADio[:NADC]:TRIGger[:SOURce]:EXTernal:DELay:STATe ON OFF 1 0 [:SOURce]:RADio[:NADC]:TRIGger[:SOURce]:EXTernal:DELay:STATe?]
Fall Delay	[[:SOURce]:RADio[:NADC]:BURSt:SHAPe:FALL:DELay <value> [:SOURce]:RADio[:NADC]:BURSt:SHAPe:FALL:DELay? [:SOURce]:RADio[:NADC]:BURSt:SHAPe:FDELay <value> [:SOURce]:RADio[:NADC]:BURSt:SHAPe:FDELay?]
Fall Time	[[:SOURce]:RADio[:NADC]:BURSt:SHAPe:FALL:TIME <value> [:SOURce]:RADio[:NADC]:BURSt:SHAPe:FALL:TIME? [:SOURce]:RADio[:NADC]:BURSt:SHAPe:FTIME <value> [:SOURce]:RADio[:NADC]:BURSt:SHAPe:FTIME?]
Filter Alpha	[[:SOURce]:RADio[:NADC]:ALPHa <value> [:SOURce]:RADio[:NADC]:ALPHa?]

**Table 4-10. Mode - NADC Softkeys**

Key	SCPI Command
Filter RNYQ NYQ	[:SOURce]:RADio[:NADC]:FILTer RNYQuist NYQuist [:SOURce]:RADio[:NADC]:FILTer?
FIX4	[:SOURce]:RADio[:NADC]:DATA FIX4 [:SOURce]:RADio[:NADC]:DATA:FIX4 <0-15> [:SOURce]:RADio[:NADC]:DATA:FIX4? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom FIX4 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom:FIX4 <0-15> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom:FIX4? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] FIX4 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA]:FIX4 <0-15> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA]:FIX4? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom FIX4 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom:FIX4 <0-15> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom:FIX4? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] FIX4 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA]:FIX4 <0-15> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA]:FIX4?
Frame Repeat Single Cont	[:SOURce]:RADio[:NADC]:REPeat SINGLE CONTInuous [:SOURce]:RADio[:NADC]:REPeat?
Frame Trigger	[:SOURce]:RADio[:NADC]:TRIGger[:SOURce]?
NADC Off On	[:SOURce]:RADio[:NADC][:STATe] ON OFF 1 0 [:SOURce]:RADio[:NADC][:STATe]?
Optimize RNYQ For EVM ACP	[:SOURce]:RADio[:NADC]:CHANnel EVM ACP [:SOURce]:RADio[:NADC]:CHANnel?
Pattern Repeat Single Cont	[:SOURce]:RADio[:NADC]:REPeat SINGLE CONTInuous [:SOURce]:RADio[:NADC]:REPeat?
Pattern Trigger	[:SOURce]:RADio[:NADC]:TRIGger[:SOURce]?
Phase Polarity Normal Invert	[:SOURce]:RADio[:NADC]:POLarity[:ALL] NORMAl INVerted [:SOURce]:RADio[:NADC]:POLarity[:ALL]?
PN9	[:SOURce]:RADio[:NADC]:DATA PN9 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom PN9 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] PN9 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom PN9 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] PN9

**Table 4-10. Mode - NADC Softkeys**

Key	SCPI Command
PN9 Mode Normal Quick Option 1EH Only	[[:SOURce]:RADio[:NADC]:BURSt:PN9 NORMAl QUICK [:SOURce]:RADio[:NADC]:BURSt:PN9?
PN15	[[:SOURce]:RADio[:NADC]:DATA PN15 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom PN15 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] PN15 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom PN15 [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] PN15
Rate Full Half	[[:SOURce]:RADio[:NADC]:FRATe FULL HALF [:SOURce]:RADio[:NADC]:FRATe?
Rise Delay	[[:SOURce]:RADio[:NADC]:BURSt:SHAPe:RISE:DELay <value> [:SOURce]:RADio[:NADC]:BURSt:SHAPe:RISE:DELay? [:SOURce]:RADio[:NADC]:BURSt:SHAPe:RDELay <value> [:SOURce]:RADio[:NADC]:BURSt:SHAPe:RDELay?
Rise Time	[[:SOURce]:RADio[:NADC]:BURSt:SHAPe:RISE:TIME <value> [:SOURce]:RADio[:NADC]:BURSt:SHAPe:RISE:TIME? [:SOURce]:RADio[:NADC]:BURSt:SHAPe:RTIME <value> [:SOURce]:RADio[:NADC]:BURSt:SHAPe:RTIME?
SACCH	[[:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel:SACChanel <bit_pattern> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel:SACChanel? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel:SACChanel <bit_pattern> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel:SACChanel?
Select File	[[:SOURce]:RADio[:NADC]:DATA "file name" [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DCUStom "file name" [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel[:DATA] "file name" [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UCUStom "file name" [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel[:DATA] "file name"
SYNC	[[:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel:SWORd <bit_pattern> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:DTCHannel:SWORd? [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel:SWORd <bit_pattern> [:SOURce]:RADio[:NADC]:SLOT[1]2 3 4 5 6:UTCHannel:SWORd?
Sync Out	[[:SOURce]:RADio[:NADC]:SOUT?
Sync Out Offset	[[:SOURce]:RADio[:NADC]:SOUT:OFFSet <value> [:SOURce]:RADio[:NADC]:SOUT:OFFSet?

**Table 4-10. Mode - NADC Softkeys**

Key	SCPI Command
Timeslot Off On	[:SOURce]:RADio[:NADC]:SLOT[1] 2 3 4 5 6:STATe ON OFF 1 0 [:SOURce]:RADio[:NADC]:SLOT[1] 2 3 4 5 6:STATe?
Timeslot Type	[:SOURce]:RADio[:NADC]:SLOT[1] 2 3 4 5 6[:TYPE]?
Trigger Key	[:SOURce]:RADio[:NADC]:TRIGger[:SOURce] KEY
Up Custom	[:SOURce]:RADio[:NADC]:SLOT[1] 2 3 4 5 6[:TYPE] UCUSom
Up TCH	[:SOURce]:RADio[:NADC]:SLOT[1] 2 3 4 5 6[:TYPE] UTCH
Up TCH All	[:SOURce]:RADio[:NADC]:SLOT[1] 2 3 4 5 6[:TYPE] UTCH_ALL

**Table 4-11. Phase Mod Softkeys**

Key	SCPI Command
$\phi$ M Dev	[:SOURce]:PM[1] 2[:DEViation] <val><unit> [:SOURce]:PM[1] 2[:DEViation]?
$\phi$ M Dev Couple Off On	[:SOURce]:PM[1] 2[:DEViation]TRACk ON OFF 1 0 [:SOURce]:PM[1] 2[:DEViation]TRACk?
$\phi$ M Dual-Sine Ampl Ratio	[:SOURce]:PM[1] 2:INTernal[1]:FREQuency:ALTErnate:AMPLitude:PERCent <val><unit> [:SOURce]:PM[1] 2:INTernal[1]:FREQuency:ALTErnate:AMPLitude:PERCent?
$\phi$ M Off On	[:SOURce]:PM[1] 2:STATe ON OFF 1 0 [:SOURce]:PM[1] 2:STATe?
$\phi$ M Rate	[:SOURce]:PM[1] 2:INTernal[1]:FREQuency <val><unit> [:SOURce]:PM[1] 2:INTernal[1]:FREQuency?
$\phi$ M Source	[:SOURce]:PM[1] 2:SOURce?
$\phi$ M Start Rate	[:SOURce]:PM[1] 2:INTernal[1]:FREQuency <val><unit> [:SOURce]:PM[1] 2:INTernal[1]:FREQuency?
$\phi$ M Stop Rate	[:SOURce]:PM[1] 2:INTernal[1]:FREQuency:ALTErnate <val><unit> [:SOURce]:PM[1] 2:INTernal[1]:FREQuency:ALTErnate?
$\phi$ M Sweep Time	[:SOURce]:PM[1] 2:INTernal[1]:SWEep:TIME <val><unit> [:SOURce]:PM[1] 2:INTernal[1]:SWEep:TIME?
$\phi$ M Sweep Trigger	:TRIGger[:SEQUence]:SOURce?



**Table 4-11. Phase Mod Softkeys**

Key	SCPI Command
φM Tone 1 Rate	[:SOURce]:PM[1]2:INTernal[1]:FREQuency <val><unit> [:SOURce]:PM[1]2:INTernal[1]:FREQuency?
φM Tone 2 Rate	[:SOURce]:PM[1]2:INTernal[1]:FREQuency:ALTErnate <val><unit> [:SOURce]:PM[1]2:INTernal[1]:FREQuency:ALTErnate?
φM Waveform	[:SOURce]:PM[1]2:INTernal[1]:FUNCTion:SHAPE?
Bus	[:SOURce]:PM[1]2:INTernal[1]:SWEep:TRIGger BUS [:SOURce]:PM[1]2:INTernal[1]:SWEep:TRIGger?
DCFM/DCφM Cal	:CALibration:DCFM
Dual-Sine	[:SOURce]:PM[1]2:INTernal[1]:FUNCTion:SHAPE DUALsine [:SOURce]:PM[1]2:INTernal[1]:FUNCTion:SHAPE?
Ext	[:SOURce]:PM[1]2:INTernal[1]:SWEep:TRIGger EXTernal [:SOURce]:PM[1]2:INTernal[1]:SWEep:TRIGger?
Ext 1 AC-Coupled	[:SOURce]:PM[1]2:SOURce EXTernal[1] [:SOURce]:PM[1]2:EXTernal[1]:COUPling AC [:SOURce]:PM[1]2:EXTernal[1]:COUPling?
Ext 1 DC-Coupled	[:SOURce]:PM[1]2:SOURce EXTernal[1] [:SOURce]:PM[1]2:EXTernal[1]:COUPling DC [:SOURce]:PM[1]2:EXTernal[1]:COUPling?
Ext 2 AC-Coupled	[:SOURce]:PM[1]2:SOURce EXTernal2 [:SOURce]:PM[1]2:EXTernal2:COUPling AC [:SOURce]:PM[1]2:EXTernal2:COUPling?
Ext 2 DC-Coupled	[:SOURce]:PM[1]2:SOURce EXTernal2 [:SOURce]:PM[1]2:EXTernal2:COUPling DC [:SOURce]:PM[1]2:EXTernal2:COUPling?
FM/φ Normal High BW	[:SOURce]:PM[1]2:BANDwidth BWIDth NORMAlHIGH [:SOURce]:PM[1]2:BANDwidth BWIDth?
Immediate	[:SOURce]:PM[1]2:INTernal[1]:SWEep:TRIGger IMMEDIATE [:SOURce]:PM[1]2:INTernal[1]:SWEep:TRIGger?
Internal	[:SOURce]:PM[1]2:SOURce INTernal[1] [:SOURce]:PM[1]2:SOURce?
Noise	[:SOURce]:PM[1]2:INTernal[1]:FUNCTion:SHAPE NOISE [:SOURce]:PM[1]2:INTernal[1]:FUNCTion:SHAPE?

**Table 4-11. Phase Mod Softkeys**

Key	SCPI Command
Ramp	[:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe RAMP [:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe?
Sine	[:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe SINE [:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe?
Square	[:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe SQUare [:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe?
Swept-Sine	[:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe SWEPTsine [:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe?
Triangle	[:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe TRIangle [:SOURce]:PM[1]2:INTernal[1]:FUNction:SHApe?
Trigger In Polarity Neg Pos	:TRIGger[:SEQuence]:SLOPe POSitive NEGative :TRIGger[:SEQuence]:SLOPe?
Trigger Key	[:SOURce]:PM[1]2:INTernal[1]:SWEep:TRIGger KEY [:SOURce]:PM[1]2:INTernal[1]:SWEep:TRIGger?
Trigger Out Polarity Neg Pos	:TRIGger:OUTPut:POLarity POSitive NEGative :TRIGger:OUTPut:POLarity?

**Table 4-12. Preset Hardkey**

Key	SCPI Command
<b>Preset (hardkey)</b>	:SYSTem:PRESet

**Table 4-13. Pulse Softkeys**

Key	SCPI Command
Ext2 DC-Coupled	[:SOURce]:PULM:SOURce EXTernal2 [:SOURce]:PULM:SOURce?
Internal Pulse	[:SOURce]:PULM:SOURce INT [:SOURce]:PULM:SOURce? [:SOURce]:PULM:INTernal[1]:FUNction:SHApe <PULSe> [:SOURce]:PULM:INTernal[1]:FUNction:SHApe?

**Table 4-13. Pulse Softkeys**

Key	SCPI Command
Internal Square	[:SOURce]:PULM:SOURce INT [:SOURce]:PULM:SOURce? [:SOURce]:PULM:INTernal[1]:FUNction:SHApe <SQUare> [:SOURce]:PULM:INTernal[1]:FUNction:SHApe?
Pulse Off On	:SOURce]:PULM:STATe ON OFF 1 0 :SOURce]:PULM:STATe?
Pulse Period	[:SOURce]:PULM:INTernal[1]:PERiod <val><unit> [:SOURce]:PULM:INTernal[1]:PERiod?
Pulse Rate	:SOURce]:PULM:INTernal[1]:FREQuency <val><unit> :SOURce]:PULM:INTernal[1]:FREQuency?
Pulse Width	[:SOURce]:PULM:INTernal[1]:PWIDth <val><unit> [:SOURce]:PULM:INTernal[1]:PWIDth?

**Table 4-14. Recall Softkeys**

Key	SCPI Command
Recall Reg	*RCL <reg>[, <seq>]

**Table 4-15. RF On/Off Hardkey**

Key	SCPI Command
<b>RF On/Off (hardkey)</b>	:OUTPut[:STATe] ON OFF 1 0 :OUTPut[:STATe]?

**Table 4-16. Save Softkeys**

Key	SCPI Command
Add Comment To Seq[n] Reg[nn]	:MEMory:STATe:COMMeNt <reg_num>,<seq_num>,<comment> :MEMory:STATe:COMMeNt? <reg_num>,<seq_num>
Save Seq[n] Reg[nn]	*SAV <reg>[, <seq>]

**Table 4-17. Sweep/List Softkeys**

Key	SCPI Command
#Points	[:SOURce]:SWEep:POINts <val> [:SOURce]:SWEep:POINts?
Ampl	[:SOURce]:POWer:MODE LIST [:SOURce]:POWer:MODE?
Ampl Start	[:SOURce]:POWer:STARt <val><unit> [:SOURce]:POWer:STARt?
Ampl Stop	[:SOURce]:POWer:STOP <val><unit> [:SOURce]:POWer:STOP?
Bus	[:SOURce]:LIST:TRIGger:SOURce BUS [:SOURce]:LIST:TRIGger:SOURce?
Configure List Sweep	[:SOURce]:LIST:DWELl <val>{, <val>} [:SOURce]:LIST:DWELl? [:SOURce]:LIST:DWELl:POINts? [:SOURce]:LIST:FREQuency <val>{, <val>} [:SOURce]:LIST:FREQuency? [:SOURce]:LIST:FREQuency:POINts? [:SOURce]:LIST:POWer <val>{, <val>} [:SOURce]:LIST:POWer? [:SOURce]:LIST:POWer:POINts?
Dwell Type List Step	[:SOURce]:LIST:DWELl:TYPE <val>{, <val>} [:SOURce]:LIST:DWELl:TYPE LIST STEP [:SOURce]:LIST:DWELl:TYPE?
Ext Neg	[:SOURce]:LIST:TRIGger:SOURce EXTernal [:SOURce]:LIST:TRIGger:SOURce? :TRIGger[:SEQuence]:SOURce EXTernal :TRIGger[:SEQuence]:SOURce? :TRIGger[:SEQuence]:SLOPe NEGative :TRIGger[:SEQuence]:SLOPe?
Ext Pos	[:SOURce]:LIST:TRIGger:SOURce EXTernal [:SOURce]:LIST:TRIGger:SOURce? :TRIGger[:SEQuence]:SOURce EXTernal :TRIGger[:SEQuence]:SOURce? :TRIGger[:SEQuence]:SLOPe POSitive :TRIGger[:SEQuence]:SLOPe?

**Table 4-17. Sweep/List Softkeys**

Key	SCPI Command
Freq	[:SOURce]:FREQuency:MODE LIST [:SOURce]:FREQuency:MODE?
Freq&Ampl	[:SOURce]:POWer:MODE LIST [:SOURce]:POWer:MODE? [:SOURce]:FREQuency:MODE LIST [:SOURce]:FREQuency:MODE?
Freq Start	[:SOURce]:FREQuency:STARt <val><unit> [:SOURce]:FREQuency:STARt?
Freq Stop	[:SOURce]:FREQuency:STOP <val><unit> [:SOURce]:FREQuency:STOP?
Immediate	[:SOURce]:LIST:TRIGger:SOURce IMMEDIATE [:SOURce]:LIST:TRIGger:SOURce?
Load List From Selected File	:MEMory:LOAD:LIST <filename> :MMEMory:LOAD:LIST <filename>
Manual Mode Off On	[:SOURce]:LIST:MODE AUTO MANual [:SOURce]:LIST:MODE?
Manual Point	[:SOURce]:LIST:MANual <val> [:SOURce]:LIST:MANual?
Off	[:SOURce]:FREQuency:MODE CW FIXED [:SOURce]:FREQuency:MODE? [:SOURce]:POWer:MODE FIXED [:SOURce]:POWer:MODE?
Point Trigger	[:SOURce]:LIST:TRIGger:SOURce BUS IMMEDIATE EXTERNAL KEY [:SOURce]:LIST:TRIGger:SOURce?
Single Sweep	:INITiate:CONTInuous[:ALL] ON OFF 1 0 :INITiate[:IMMEDIATE][:ALL]
Step Dwell	[:SOURce]:SWEep:DWELl <val> [:SOURce]:SWEep:DWELl?
Store List to File	:MEMory:STORe:LIST <filename> :MMEMory:STORe:LIST <filename>

**Table 4-17. Sweep/List Softkeys**

Key	SCPI Command
Sweep	[:SOURce]:FREQUency:MODE CW FIXed LIST [:SOURce]:FREQUency:MODE? [:SOURce]:POWer:MODE FIXed LIST [:SOURce]:POWer:MODE?
Sweep Direction Down Up	[:SOURce]:LIST:DIRection UP DOWN [:SOURce]:LIST:DIRection?
Sweep Repeat Single Cont	:INITiate:CONTInuous[:ALL] ON OFF 1 0 :INITiate:CONTInuous[:ALL]?
Sweep Trigger	TRIGger[:SEQUence]:SOURce BUS IMMediate EXTernal KEY :TRIGger[:SEQUence]:SOURce?
Sweep Type List Step	[:SOURce]:LIST:TYPE LIST STEP [:SOURce]:LIST:TYPE?
Trigger Key	[:SOURce]:LIST:TRIGger:SOURce KEY [:SOURce]:LIST:TRIGger:SOURce?
Trigger Out Polarity Neg Pos	:TRIGger:OUTPut:POLarity POSitive NEGative :TRIGger:OUTPut:POLarity?

**Table 4-18. Trigger Hardkey**

Key	SCPI Command
<b>Trigger (hardkey)</b>	:TRIGger[:SEQUence][:IMMediate]

**Table 4-19. Utility Softkeys**

Key	SCPI Command
All	:MEMory:CATalog[:ALL]?
Binary	:MEMory:CATalog:BINary?
Brightness	:DISPlay:BRIGhtness <value> :DISPlay:BRIGhtness?
Clear Error Queue(s)	*CLS

**Table 4-19. Utility Softkeys**

Key	SCPI Command
Delete All Binary Files	:MEMory:DELeTe:BINary
Delete All Files	:MEMory:DELeTe:ALL
Delete All List Files	:MEMory:DELeTe:LIST
Delete All State Files	:MEMory:DELeTe:STATe
Delete File	:MEMory:DELeTe[:NAME] <filename>
Help Mode Single Cont	:SYSTem:HELp:MODE SINGle CONTInuous :SYSTem:HELp:MODE?
HP8648A/B/C/D	:SYSTem:LANGUage "HP8648" :SYSTem:LANGUage? :SYSTem:PRESet:LANGUage "HP8648" :SYSTem:PRESet:LANGUage?
HP-IB Address	:SYSTem:COMMUnicate:GPIB:ADDRess <number> :SYSTem:COMMUnicate:GPIB:ADDRess?
Inverse Video Off On	:DISPlay:INVerse ON OFF 1 0> :DISPlay:INVerse?
List	:MEMory:CATalog:LIST?
Off	:SYSTem:COMMUnicate:SERial:CONTrol:RTS OFF :SYSTem:COMMUnicate:SERial:CONTrol:RTS?
Power On Last Preset	:SYSTem:PON:TYPE PRESet LAST :SYSTem:PON:TYPE?
Preset Language	:SYSTem:PRESet:LANGUage?
Preset Normal User	:SYSTem:PRESet:TYPE NORMAl USER :SYSTem:PRESet:TYPE?
Receive Pace None Xon	:SYSTem:COMMUnicate:SERial:RECeive:PACE XON NONE :SYSTem:COMMUnicate:SERial:RECeive:PACE?
Remote Language	:SYSTem:LANGUage?
Reset RS-232	:SYSTem:COMMUnicate:SERial:RESet

**Table 4-19. Utility Softkeys**

Key	SCPI Command
Reverse Power Protection Normal HP8648	:OUTPut:PROTection:MODE "NORMAL" "HP8648" :OUTPut:PROTection:MODE? :OUTPut:PROTection:TRIPped? :OUTPut:PROTection:CLEar
RS-232 Baud Rate	:SYSTem:COMMunicate:SERial:BAUD <number> :SYSTem:COMMunicate:SERial:BAUD?
RS-232 Echo Off On	:SYSTem:COMMunicate:SERial:ECHO ON OFF 1 0 :SYSTem:COMMunicate:SERial:ECHO?
RTS/CTS	:SYSTem:COMMunicate:SERial:CONTRol:RTS?
RTS/CTS Pacing	:SYSTem:COMMunicate:SERial:CONTRol:RTS STANdard :SYSTem:COMMunicate:SERial:CONTRol:RTS?
RTS On	:SYSTem:COMMunicate:SERial:CONTRol:RTS ON :SYSTem:COMMunicate:SERial:CONTRol:RTS?
Save User Preset	:SYSTem:PRESet[:USER]:SAVE
SCPI	:SYSTem:PRESet:LANGUage "SCPI" :SYSTem:PRESet:LANGUage? :SYSTem:LANGUage "SCPI" :SYSTem:LANGUage?
Screen Saver Delay	:SYSTem:SSAVer:DELay <val> :SYSTem:SSAVer:DELay?
Screen Saver Mode	:SYSTem:SSAVer:MODE LIGHt TEXTernal :SYSTem:SSAVer:MODE?
Screen Saver Off On	:SYSTem:SSAVer:STATe ON OFF 1 0 :SYSTem:SSAVer:STATe?
State	:MEMory:CATalog:STATe?
Transmit Pace None Xon	:SYSTem:COMMunicate:SERial:TRANsmit:PACE XON NONE :SYSTem:COMMunicate:SERial:TRANsmit:PACE?
View Next Error Message	:SYSTem:ERRor[:NEXT]?



---

## HP 8656/57-Compatible Language

The HP ESG Series Signal Generators have the capability of operating in an HP 8656/57-compatible programming mode. Table 4-20 shows the HP8656/57 programming codes that are implemented in the signal generator.

**Table 4-20. HP 8656/57-Compatible Programming Codes**

HP 8565/57 Command <sup>1</sup>	Parameter	Comments	Status
AM	Amplitude Modulation	Function Entry	Implemented
AO	Amplitude Offset	Function Entry	Implemented
AP	Amplitude (carrier)	Function Entry	Implemented
DB	dB	Units Entry	Implemented
DF	dBf	Units Entry	Implemented
DM	dBm	Units Entry	Implemented
DN	Step Down	Function Feature	Not Implemented
EM	EMF	Units Entry	Implemented
FM	Frequency Modulation	Function Entry	Implemented
FR	Frequency (carrier)	Function Entry	Implemented
GT	Flexible Sequence	Feature	Implemented
HI	HI ALC	Function Feature	Not Implemented
HZ	Hz	Units Entry	Implemented
IS	Increment Set	Function Qualifier	Not Implemented
KZ	kHz	Units Entry	Implemented
LO	LO ALC	Function Feature	Not Implemented
MV	mV	Units Entry	Implemented
MZ	MHz	Units Entry	Implemented
PC	Percent <sup>2</sup>	Units Entry	Implemented
PD	Phase Decrement	Function Feature	Implemented
PF	Pulse Modulation (Fast Mode)	Function Entry	Implemented
PI	Phase Increment	Function Feature	Implemented
PM	Pulse Modulation	Function Feature	Implemented
QS	Reverse Sequence	Feature	Implemented
RC	Recall (0-9)	Feature	Implemented

**Table 4-20. HP 8656/57-Compatible Programming Codes**

RL	Recall (0-99)	Feature	Implemented
RP	Reverse Power Protection Reset <sup>3</sup>	Feature	Implemented
R0	Standby	Feature	Not Implemented
R1	On	Feature	Not Implemented
R2	RF Off	Function Feature	Implemented
R3	RF On	Function Feature	Implemented
R5	RF Dead (Full Attenuator)	Function Feature	Implemented
SQ	Sequence	Feature	Implemented
ST	Save (0-9)	Feature	Implemented
SV	Save (0-99)	Feature	Implemented
S1	External Modulation Source	Source Qualifier	Implemented
S2	Internal 400 Hz Modulation Source	Source Qualifier	Implemented
S3	Internal 1 kHz Modulation Source	Source Qualifier	Implemented
S4	Modulation Source Off	Source Qualifier	Implemented
S5	DC FM	Function Entry	Implemented
UP	Step Up	Function Feature	Not Implemented
UV	μV	Units Entry	Implemented
VL	Volts	Units Entry	Implemented
0-9	Numerals 0-9	Data Entries	Implemented
-	Minus Sign	Data Entry	Implemented
.	Decimal Point	Data Entry	Implemented
%	Percent <sup>2</sup>	Units Entry	Implemented

1. Program codes can be either upper or lower case.
2. Either PC or % can be used.
3. The source of reverse power must be removed.

## Command Mapping

The SCPI command, :SYSTem:LANGuage "SCPI"|"COMP", lets you set the signal generator remote (HP-IB) language to one of the following choices:

- "SCPI" - Default language (Standard Commands for Programmable Instruments)
- "COMP" - HP 8656B, HP 8657A/B compatibility

When you operate in an HP 8656/57-compatibility language, internal mapping occurs in the signal generator to effect an equivalent SCPI response. In addition, the modulation sources for the HP 8656/57 are not the same as for the HP ESG series, so when you operate in an HP 8656/57-compatibility language, the following mapping occurs for selecting AM and FM sources:

Modulation Sources	
HP 8656/57	HP ESG Series
AM, Internal	AM1, Internal 1
AM, External	AM2, External 1
FM, Internal	FM1, Internal 1
FM, External	FM2, External 1
AM, Internal and External	AM1, Internal 1 and AM2, External 1
FM, Internal and External	FM1, Internal 1 and FM2, External 1

The HP 8656/57 Signal Generators allow multiple modulations to use the same input. The HP ESG Series Signal Generators do not. If you configure multiple modulations on the same input, the signal generator will respond by automatically disabling modulations.

The mapping between the HP 8656/57-compatible commands and the SCPI commands will change depending on the other parameters that are executed. Refer to the following explanations of the commands:

**AM (Amplitude Modulation)**

- AM becomes the active function.
- AM1 and AM2 depth values are coupled with [:SOURce]:AM[1]2[:DEPTh]:TRACk ON
- If AM was already on, or no modulation was on, the following sequences of SCPI commands are implemented when an AM command is sent with a modulation source command:

	AM Already On	No Modulation Already On
S1	[:SOURce]:AM2:EXTernal[1]:COUPling AC [:SOURce]:AM2:SOURce EXTernal1	[:SOURce]:AM2:EXTernal[1]:COUPling AC [:SOURce]:AM2:SOURce EXTernal1 [:SOURce]:AM2:STATe ON
S2	[:SOURce]:AM[1]:SOURce INT[1] [:SOURce]:AM[1]:INTernal[1]:FREQuency 400 Hz	[:SOURce]:AM[1]:SOURce INT[1] [:SOURce]:AM[1]:INTernal[1]:FREQuency 400 Hz [:SOURce]:AM[1]:STATe ON
S3	[:SOURce]:AM[1]:SOURce INT[1] [:SOURce]:AM[1]:INTernal[1]:FREQuency 1 kHz	[:SOURce]:AM[1]:SOURce INT[1] [:SOURce]:AM[1]:INTernal[1]:FREQuency 1 kHz [:SOURce]:AM[1]:STATe ON

- If FM or pulse modulation was already on, the signal generator attempts to set up AM with the same settings. In addition, the following commands are mapped:

	FM1 Already On	FM2 or Pulse Already On
S1, S2, or S3	[:SOURce]:AM[1]:SOURce INT[1] [:SOURce]:AM[1]:STATe ON [:SOURce]:FM[1]:STATe OFF	[:SOURce]:AM2:SOURce EXTernal1 [:SOURce]:AM2:STATe ON [:SOURce]:FM2:STATe OFF [:SOURce]:PULM:STATe OFF

**FM (Frequency Modulation)**

- FM becomes the active function.
- FM1 and FM2 deviation values are coupled with [:SOURce]:FM[1]|2[:DEViation]:TRACK ON
- If FM was already on, or no modulation was on, the following sequences of SCPI commands are implemented when an FM command is sent with a modulation source command:

	FM Already On	No Modulation Already On
S1	[:SOURce]:FM2:EXTernal[1]:COUPling AC [:SOURce]:FM2:SOURce EXTernal1	[:SOURce]:FM2:EXTernal[1]:COUPling AC [:SOURce]:FM2:SOURce EXTernal1 [:SOURce]:FM2:STATe ON
S2	[:SOURce]:FM[1]:SOURce INT[1] [:SOURce]:FM[1]:INTernal[1]:FREQuency 400 Hz	[:SOURce]:FM[1]:SOURce INT[1] [:SOURce]:FM[1]:INTernal[1]:FREQuency 400 Hz [:SOURce]:FM[1]:STATe ON
S3	[:SOURce]:FM[1]:SOURce INT[1] [:SOURce]:FM[1]:INTernal[1]:FREQuency 1 kHz	[:SOURce]:FM[1]:SOURce INT[1] [:SOURce]:FM[1]:INTernal[1]:FREQuency 1 kHz [:SOURce]:FM[1]:STATe ON

- If AM or pulse modulation was already on, the signal generator attempts to set up FM with the same settings. In addition, the following commands are mapped:

	AM1 Already On	AM2 or Pulse Already On
S1, S2, or S3	[:SOURce]:FM[1]:SOURce INT[1] [:SOURce]:FM[1]:STATe ON [:SOURce]:AM[1]:STATe OFF	[:SOURce]:FM2:SOURce EXTernal1 [:SOURce]:FM2:STATe ON [:SOURce]:AM2:STATe OFF [:SOURce]:PULM:STATe OFF

### **PF (Pulse Modulation - Fast Mode) or PM (Pulse Modulation)**

The HP ESG Series Signal Generators support only one input for pulse: EXTERNAL2. (This is a DC-coupled input.) Internal pulse modulation, therefore, is not supported in HP 8656/57-compatibility mode. The PF (or PM) command is mapped to the following SCPI commands:

```
[[:SOURce]:PULM:SOURce EXTERNAL2
```

```
[[:SOURce]:PULM:STATe ON
```

### **S1 (External Modulation Source)**

No action is taken until a modulation form is selected. Pulse is set to EXTERNAL2. AM and FM are set to EXTERNAL1.

### **S2 (Internal 400 Hz Modulation Source)**

- If AM1 is on, this command is mapped to:

```
[[:SOURce]:AM[1]:INTERNAL[1]:FREQUENCY 400 Hz
```

- If FM1 is on, this command is mapped to:

```
[[:SOURce]:FM[1]:INTERNAL[1]:FREQUENCY 400 Hz
```

### **S3 (Internal 1 kHz Modulation Source)**

- If AM1 is on, this command is mapped to:

```
[[:SOURce]:AM[1]:INTERNAL[1]:FREQUENCY 1 kHz
```

- If FM1 is on, this command is mapped to:

```
[[:SOURce]:FM[1]:INTERNAL[1]:FREQUENCY 1 kHz
```

#### **S4 (Modulation Source Off)**

- If the last active function was PM, (other than any modulation source commands), pulse modulation is disabled by mapping to the following command:

`[[:SOURCE]:PULM:STATe OFF`

- If the last command was S2 or S3, internal modulation is turned off for whatever modulation type is currently active by mapping to the following commands:

`[[:SOURCE]:AM[1]:STATe OFF`

`[[:SOURCE]:FM[1]:STATe OFF`

The S2 or S3 condition is also discarded so that the next time S4 is sent, the entire modulation type will be turned off.

- If the last command was S1, external modulation is turned off for whatever modulation type is currently active by mapping to the following commands:

`[[:SOURCE]:AM2:STATe OFF`

`[[:SOURCE]:FM2:STATe OFF`

The S1 condition is also discarded so that the next time S4 is sent, the entire modulation type will be turned off.

- If the last active function was AM or FM, the entire appropriate modulation type is turned off by mapping to the following commands:

`[[:SOURCE]:AM[1]:STATe OFF`

`[[:SOURCE]:AM2:STATe OFF`

`[[:SOURCE]:FM[1]:STATe OFF`

`[[:SOURCE]:FM2:STATe OFF`

#### **S5 (DC FM)**

- FM becomes the active function.
- In addition, the following commands are mapped:

`[[:SOURCE]:FM2:SOURce EXTernal1`

`[[:SOURCE]:PULM:STATe OFF`

`[[:SOURCE]:AM[1]:STATe OFF`

`[[:SOURCE]:AM2:STATe OFF`

`[[:SOURCE]:FM2:EXTernal[1]:COUPling DC`

`[[:SOURCE]:FM2:STATe ON`

## **Additional Commands**

To allow configuration of the compatibility mode as well as to isolate problems, the following SCPI commands are supported while in HP 8656/57-compatibility mode:

```
:SYSTem:LANGUage "SCPI"|"COMP"|"NADC"|"PDC"|"PHS"
```

```
:SYSTem:LANGUage?
```

```
:SYSTem:ERRor?
```





---

## 5 Language Reference

---

This chapter describes each of the SCPI commands alphabetically, by subsystem. The descriptions include syntax requirements, ranges, restrictions, query responses, and status at instrument preset.

---

## Command Syntax

Following the heading for each programming command entry is a syntax statement showing the proper syntax for the command. An example syntax statement is shown here:

**POWer**[ :LEVel ] MAXimum|MIN

Syntax statements read from left to right. In this example, the **:LEVel** portion of the statement immediately follows the **POWer** portion of the statement with no separating space. A separating space is legal only between the command and its argument. In this example, the portion following the **[ :LEVel ]** portion of the statement is the argument. Additional conventions used in the syntax statements are defined as follows:

- Italics are used to symbolize a program code parameter or query response.
- ::= means “is defined as.”
- | (vertical bar) indicates a choice of one element from a list. For example, **<A> | <B>** indicates **<A>** or **<B>** but not both.
- [ ] (square brackets) indicate that the enclosed items are optional.
- Upper-case lettering indicates that the upper-case portion of the command is the minimum required for the command. For example, in the command **FREQuency**, **FREQ** is the minimum requirement.
- Lower-case lettering indicates that the lower-case portion of the command is optional; it can either be included with the upper-case portion of the command or omitted. For example, in the command **FREQuency**, either **FREQ**, or **FREQUENCY** is correct.
- ? after a subsystem command indicates that the command is a query.

---

## IEEE 488.2 Common Commands

Common commands are generally not measurement related, but are used to manage macros, status registers, synchronization, and data storage. All common commands begin with an asterisk. The common commands are defined by IEEE 488.2

### **\*CLS (Clear Status)**

The \*CLS command clears the status byte, the data questionable event register, the standard event status register, the standard operation status register and any other registers that are summarized in the status byte.

### **\*ESE (Standard Event Status Enable)**

\*ESE <data>

The \*ESE command sets the standard event status enable register.

\*ESE?

\*ESE? queries the status of the standard event status enable register.

### **\*ESR? (Standard Event Status Register)**

\*ESR? queries the value of the standard event status register. This is a destructive read.

### **\*IDN? (Identification)**

The \*IDN? query outputs an identifying string to the HP-IB. The response for the signal generator will be "HEWLETT-PACKARD, US36260150, ESG-D400A, A.01.00" where the actual model number, serial number and firmware revision will be substituted.

### **\*OPC (Operation Complete)**

The \*OPC command sets bit 0 in the standard event status register when all pending operations have finished.

### **\*OPC? (Operation Complete)**

\*OPC? queries bit 0 in the standard event status register. The signal generator will return an ASCII '1' when all pending operations have finished.

### **\*RCL (Recall)**

\*RCL <reg>, <seq>

The \*RCL <reg>, <seq> command recalls the instrument state from the specified memory register <reg> of the specified sequence <seq>. The range of registers <reg> is 0 through 99 and the range of sequences <seq> is 0 through 9.

### **\*RST (Reset)**

The \*RST command resets the instrument to a factory pre-defined condition.

### **\*SAV (Save)**

\*RCL <reg>, <seq>

The \*SAV <reg>, <seq> command saves the instrument state to the specified memory register <reg> of the specified sequence <seq>. The range of registers <reg> is 0 through 99 and the range of sequences <seq> is 0 through 9.

### **\*SRE (Service Request Enable)**

\*SRE <data>

The \*SRE command sets the value of the service request enable register.

### **\*SRE? (Service Request Enable Query)**

\*SRE <data>

The \*SRE? queries the value of the service request enable register.

### **\*STB? (Read Status Byte)**

\*STB? queries the status byte. This is a non-destructive read.

### **\*TRG (Trigger)**

The \*TRG command triggers the device if, and only if, Bus Triggering is the type of trigger event selected. Otherwise, \*TRG is ignored.

**\*TST? (Self-Test)**

The \*TST? query returns the result of the power-up selftest:

- 0 - Passed (no tests failed and at least one test passed)
- 1 - Failed (one or more tests failed)

**\*WAI (Wait-to-continue)**

The \*WAI command causes the instrument to wait until all pending commands are completed, before executing any other commands.

---

## Subsystem Commands

Subsystem commands include all measurement functions and some general purpose functions. Subsystem commands are distinguished by the colon used between keywords, as in **AM: SOURCE**. Each subsystem is a set of commands that roughly corresponds to a functional block of the instrument.

---

## :AM Subsystem

The amplitude modulation subsystem is used to set the modulation controls and the parameters associated with amplitude modulated signals.

### Wideband Amplitude Modulation State

#### Command Mnemonic

**:AM:WIDeband:STATE ON|OFF|1|0**

#### Command Description

This command sets the operating state of the wideband amplitude modulation source. The choices are On (1) or Off (0). AM wideband (Options 1EH, UN3, and UN4 only) provides bandwidth beyond the standard AM with fixed depth (100%). The modulation source is the I input.

**AM Path 1** and **AM Path 2** are summed internally for composite modulation. Either path can be switched to any one of the modulation sources: Int, Ext1, or Ext2. All modulation types can be simultaneously enabled, except FM with  $\Phi$ M. AM, FM, and  $\Phi$ M can sum simultaneous inputs from any two sources (Int, Ext1, and Ext2). Any given source (Int, Ext1, or Ext2) can only be routed to one activated modulation type.

#### Command Query

**:AM:WIDeband:STATE?**

#### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

### External Amplitude Modulation Source Coupling

#### Command Mnemonic

**:AM[1]|2:EXTErnal[1]|2:COUPling AC|DC**



### Command Description

This command sets the external coupling for the amplitude modulation source, if External was selected as the **AM[1] | 2:SOURCE**. The choices are AC or DC coupling. This command does not change the currently active source, nor does it switch the current modulation on or off. The modulating signal may be the sum of several signals, either internal or external sources. If the coupling is set to DC, then both the AC and DC signal components pass. AC coupling passes only AC signal components.

The command **:AM[1] | 2:EXTERNAL[1] | 2:COUPLING AC** lets you input an external, AC-coupled, amplitude modulation signal to the EXT 1 INPUT connector. The modulation signal is tested for voltage and a display annunciator will report a high or low condition if the voltage is  $> \pm 3\%$  of 1 Vpk.

The command **:AM[1] | 2:EXTERNAL[1] | 2:COUPLING DC** lets you input an external, DC-coupled, amplitude modulation signal to the EXT 1 INPUT connector.

The external 1 source has multiple uses but can be used for only one modulation at a time. If, for example, you were using the external 1 source in a frequency modulation configuration for **FM Path 1**, and then you configured **AM Path 2** to also use the external 1 source, the signal generator would turn off **FM Path 1** and assign the external 1 source to your **AM Path 2** configuration.

### Command Query

**:AM[1] | 2:EXTERNAL[1] | 2:COUPLING?**

### Initial Values

At normal preset, this value is set to DC.

At \*RST, this value is set to DC.

At preset power-up, this value is set to DC.

## Internal Amplitude Modulation Source Rate

### Command Mnemonic

**:AM[1] | 2:INTERNAL[1]:FREQUENCY <val><unit>**

### Command Description

This command sets the rate of the internally-generated modulation source. The choices for the variables **<val>** and **<unit>** range from 0.1 Hz (minimum) to 50 kHz (maximum) if the internal waveform is Sine wave. For all other waveforms, the maximum internal amplitude modulation rate is 10 kHz.

Use this command to change the internal modulation frequency for the **AM Path 1** and **AM Path 2** configurations. The minimum increment allowed is 0.1 Hz. Notice that the new value of AM rate applies only to whichever AM configuration (**AM Path 1** or **AM Path 2**) you have currently selected.

### Command Query

```
:AM[1] | 2:INTernal:FREQuency?
```

### Initial Values

At normal preset, this value is set to 400.0 Hz.

At \*RST, this value is set to 400.0 Hz.

At preset power-up, this value is set to 400.0 Hz.

## Internal Amplitude Modulation Alternate Frequency

### Command Mnemonic

```
:AM[1] | 2:INTernal[1]:FREQuency:ALternate <val><unit>
```

### Command Description

This command sets the frequency for the alternate signal. The alternate frequency is the second frequency of a dual-sine or the stop frequency of a swept-sine. The choices for the variables <val> and <unit> range from 0.1 kHz (minimum) to 50.0 kHz (maximum).

### Command Query

```
:AM[1] | 2:INTernal:FREQuency:ALternate?
```

### Initial Values

At normal preset, this value is set to 400.00 Hz.

At \*RST, this value is set to 400.00 Hz.

At preset power-up, this value is set to 400.00 Hz.

## Internal Amplitude Modulation Alternate Frequency Amplitude

### Command Mnemonic

```
:AM[1] | 2:INTernal[1]:FREQuency:ALternate:AMPLitude:PERCent <val><unit>
```

### Command Description

This command sets the amplitude of the alternate frequency as a percentage of the total amplitude. Therefore, if the alternate frequency makes up 30% of the total amplitude, then the primary frequency is 70% of the total amplitude (for dual-sine only). The choices for the variables **<val>** and **<unit>** range from 0.1% (minimum) to 99.9% (maximum).

### Command Query

```
:AM[1] | 2:INTernal:FREQuency:ALternate:AMPLitude:PERCent?
```

### Initial Values

At normal preset, this value is set to 50%.

At \*RST, this value is set to 50%.

At preset power-up, this value is set to 50%.

## Internal Amplitude Modulation Waveform

### Command Mnemonic

```
:AM[1] | 2:INTernal[1]:FUNctIon:SHAPE  
SINE | TRIangle | SQUare | RAMP | NOISE | DUALsine | SWEPTsine
```

### Command Description

This command allows you to assign an AM waveform to your **AM Path 1** and **AM Path 2** configurations. Select from sine, triangle, square, ramp, noise, dual sine, and swept sine waveforms. Notice that your waveform selection applies only to whichever AM path configuration you have currently selected.

Using the command **:AM[1] | 2:INTernal[1]:FUNctIon:SHAPE SINE** lets you specify sine as the amplitude modulation waveform for the **AM Path 1** and **AM Path 2** configurations. Notice that the selected waveform applies only to whichever AM path configuration you have currently selected.

Using the command `:AM[1]|2:INTernal[1]:FUNction:SHApe TRIangle` lets you specify triangle as the amplitude modulation waveform for the **AM Path 1** and **AM Path 2** configurations. Notice that the selected waveform applies only to whichever AM path configuration you have currently selected.

Using the command `:AM[1]|2:INTernal[1]:FUNction:SHApe SQUARE` lets you specify square as the amplitude modulation waveform for the **AM Path 1** and **AM Path 2** configurations. Notice that the selected waveform applies only to whichever AM path configuration you have currently selected.

Using the command `:AM[1]|2:INTernal[1]:FUNction:SHApe RAMP` lets you specify ramp as the amplitude modulation waveform for the **AM Path 1** and **AM Path 2** configurations. Notice that the selected waveform applies only to whichever AM path configuration you have currently selected.

Using the command `:AM[1]|2:INTernal[1]:FUNction:SHApe NOISE` lets you specify noise as the amplitude modulation waveform for the **AM Path 1** and **AM Path 2** configurations. Notice that the selected waveform applies only to whichever AM path configuration you have currently selected.

Using the command `:AM[1]|2:INTernal[1]:FUNction:SHApe DUALsine` allows you to set the dual sine as the amplitude modulation waveform for the **AM Path 1** and **AM Path 2** configurations. In this mode you can set the AM rates for two separate tones. In addition, you can set a percentage of the tone 2 AM depth to the total AM depth.

Using the command `:AM[1]|2:INTernal[1]:FUNction:SHApe SWEPTSine` allows you to set the swept-sine amplitude modulation waveform for the **AM Path 1** and **AM Path 2** configurations. In this mode you can set the start and stop AM rate and the sweep time. You can set the signal generator to a single, externally-triggered sweep on either a negative or positive TTL level or you can choose continuous sweep, triggered immediately. In this menu you can also select either a positive or negative polarity for the TRIGGER OUT signal.

### Command Query

```
:AM[1]|2:INTernal:FUNction:SHApe?
```

### Initial Values

At normal preset, this value is set to Sine.

At \*RST, this value is set to Sine.

At preset power-up, this value is set to Sine.

## Internal Amplitude Modulation Sweep Time

### Command Mnemonic

```
:AM[1]|2:INTernal[1]:SWEep:TIME <val><unit>
```

### Command Description

This command selects the sweep time for a swept-sine, internally-generated signal.

### Command Query

:AM[1] | 2:INTernal:SWEep:TIME?

### Initial Values

At normal preset, this value is set to 100 msec.

At \*RST, this value is set to 100 msec.

At preset power-up, this value is set to 100 msec.

## Internal Amplitude Modulation Sweep Trigger

### Command Mnemonic

:AM[1] | 2:INTernal[1]:SWEep:TRIGger IMMEDIATE | BUS | EXTernal | KEY

### Command Description

This command selects the trigger for the amplitude modulation sweep.

### Command Query

:AM[1] | 2:INTernal:SWEep:TRIGger?

### Initial Values

At normal preset, this value is set to Immediate.

At \*RST, this value is set to Immediate.

At preset power-up, this value is set to Immediate.

## Amplitude Modulation Source

### Command Mnemonic

:AM[1] | 2:SOURce INT[1] | EXT1 | EXT2

### Command Description

This command sets the source that will generate the amplitude modulation. The choices are Internal Source 1, External Source 1, or External Source 2. You can choose internally-generated amplitude modulation or select an externally-applied signal from either the EXT 1 INPUT or EXT 2 INPUT connectors. The internal modulation is always AC-coupled. For the externally-applied signals, you can choose between AC- and DC-coupled modulation. A 1.0 Vpk input is required for calibrated AM depth settings. The **EXT 1 LO/HI** and **EXT 2 LO/HI** display annunciators will turn on if the peak input voltage differs from 1.0 Vpk by more than 3%. (The LO/HI annunciators only function for AC-coupled external inputs.)

The internal and external 1 and 2 sources have multiple uses. You can use them for amplitude, frequency, and phase modulation. However, any given source can only be routed to one enabled modulation at a time. If, for example, you were using the external 1 source in a frequency modulation configuration for **FM Path 1**, and then you configured **AM Path 2** to also use the external 1 source, the signal generator would turn off **FM Path 1** and assign the external 1 source to your **AM Path 2** configuration. Notice that for these purposes the external 1 AC-coupled source is the same as the external 1 DC-coupled source and the external 2 AC-coupled source is the same as the external 2 DC-coupled source.

### Command Query

:AM[1] | 2 : SOURce?

### Initial Values

At normal preset, this value is set to Internal.

At \*RST, this value is set to Internal.

At preset power-up, this value is set to Internal.

### Amplitude Modulation State

#### Command Mnemonic

:AM[1] | 2 : STATe ON | OFF | 1 | 0

### Command Description

This command toggles the amplitude modulation on or off for whichever AM path configuration (**AM Path 1** or **AM Path 2**) you have selected. The choices are On (1) or Off (0). Notice, however that although you can turn on amplitude modulation with this command, the RF carrier is modulated by the enabled modulation only when you have also set **Mod On/Off** to **On**. Whenever amplitude modulation is enabled, the **AM** annunciator is turned on in the display.

There are two paths for AM modulation which can be simultaneously enabled as long as they use different sources (Int, Ext1, or Ext2). **AM Path 2** is limited to a maximum rate of 1 MHz. **AM Path 2** must be set to a deviation less than or equal to **AM Path 1**. The modulation signals from both paths are summed internally for composite modulation.

### Command Query

**:AM[1] | 2:STATE?**

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

## Amplitude Modulation Depth

### Command Mnemonic

**:AM[1] | 2[:DEPTH] <val><unit>**

### Command Description

This command sets the depth of amplitude modulation.

This command is used to set the amplitude modulation depth, in percent, for the **AM Path 1** and **AM Path 2** configurations. The choices for the variables **<val>** and **<unit>** may range from 0.1 PCT to 100 PCT. After executing the command, the current value for AM depth is displayed in the active entry area. The minimum increment allowed is 0.1%. Notice that the new value of AM depth applies only to whichever AM path configuration (**AM Path 1** or **AM Path 2**) you have currently selected.

### Command Query

:AM[1] | 2[:DEPTH]?

### Initial Values

At normal preset, this value is set to 0.1%.

At \*RST, this value is set to 0.1%.

At preset power-up, this value is set to 0.1%.

## Amplitude Modulation Depth Coupling

### Command Mnemonic

:AM[1] | 2[:DEPTH]:TRACK ON|OFF | 1 | 0

### Command Description

This command links the AM depth values of **AM Path 1** and **AM Path 2** and allows the amplitude modulation depth values on both path 1 and path 2 to track each other. If the AM depth coupling function is activated, changing the AM depth on path 1 will cause an equal change in AM depth on path 2, and visa versa. The choices are On (1) or Off (0).

### Command Query

:AM[1] | 2[:DEPTH]:TRACK?

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.



---

## :CALibration Subsystem

The calibration subsystem is used to set the controls and the parameters associated with instrument calibration.

### DCFM/DC $\Phi$ M Calibration

#### Command Mnemonic

:CALibration:DCFM

#### Command Description

This command initiates a DCFM or DC $\Phi$ M calibration (depending on which kind of modulation is currently active) and stores the results in the instrument's firmware. This calibration eliminates the offset in phase (or frequency, as appropriate) modulation so that the carrier phase (or frequency) remains the same with no modulation applied. An external DC-coupled phase (or frequency) modulation must be active when the calibration is performed. After calibration, the DC signal produces no modulation.

#### Command Query

There is no query for this command.

#### Initial Values

There are no initial values associated with this command.

---

## :COMMunicate Subsystem

The communicate subsystem is used to set the controls and the parameters associated with serial system communication.

### GP-IB Address

#### Command Mnemonic

`:SYSTEM:COMMunicate:GPIB:ADDRESS <number>`

#### Command Description

This command sets the source's GP-IB address. The choices for the variable `<number>` are integers 0 through 30.

#### Command Query

`:SYSTEM:COMMunicate:GPIB:ADDRESS?`

#### Initial Values

This is a persistent state set to 19 at the factory.

### RS-232 Baud Rate

#### Command Mnemonic

`:SYSTEM:COMMunicate:SERial:BAUD <number>`

#### Command Description

This command sets the baud rate for the rear panel RS-232 interface (labeled AUXILIARY INTERFACE). The baud rate must be set to 19200 if an optional remote interface box is connected. The choices for the variable are 300, 1200, 2400, 4800, 9600, 19200, 38400.

#### Command Query

`:SYSTEM:COMMunicate:SERial:BAUD?`

#### Initial Values

This is a persistent state set to 19200 at the factory.

## RS-232 RTS Control

### Command Mnemonic

:SYSTem:COMMunicate:SERial:CONTrol:RTS ON|OFF|STANdard

### Command Description

This command controls the state of the RS-232 RTS line. The choices are On, Off, or Standard.

Use On for connecting the optional remote interface box. The instrument will ignore the state of the CTS line.

Use Off for a three-wire remote connection. With this setting, the instrument will ignore the state of the CTS line. The setting is not compatible with the optional remote interface box.

Use Standard to turn on RTS/CTS handshaking. The instrument will monitor the state of the CTS line and discontinue transmitting over RS-232 if it goes false. In addition, the RTS line will be turned off while the receive buffer of the instrument is near overflow. Do not use this setting without a properly configured remote connection which uses hardware handshaking. This setting is not compatible with the optional remote interface box.

### Command Query

:SYSTem:COMMunicate:SERial:CONTrol:RTS?

### Initial Values

This is a persistent state set to On at the factory.

## RS-232 RTS Echo

### Command Mnemonic

:SYSTem:COMMunicate:SERial:ECHO ON|OFF|1|0

### Command Description

This command controls the state of the RS-232 echo. The choices are On (1) or Off (0).

### Command Query

:SYSTem:COMMunicate:SERial:ECHO?

### Initial Values

This is a persistent state set to Off at the factory.

## RS-232 XON Handshake Receive State

### Command Mnemonic

```
:SYSTem:COMMunicate:SERial:RECeive:PACE XON|NONE
```

### Command Description

This command sets XON/XOFF handshaking when the instrument is receiving data. The choices are XON and None.

### Command Query

```
:SYSTem:COMMunicate:SERial:RECeive:PACE?
```

### Initial Values

This is a persistent state, set to None at the factory.

## RS-232 Reset

### Command Mnemonic

```
:SYSTem:COMMunicate:SERial:RESet
```

### Command Description

This command resets the RS-232 buffer. This will discard any unprocessed SCPI input received via the RS-232 port.

### Command Query

There is no query for this command.

### Initial Values

There are no initial values associated with this command.

## RS-232 XON Handshake Transmit State

### Command Mnemonic

```
:SYSTem:COMMunicate:SERial:TRANsmiT:PACE XON|NONE
```

### **Command Description**

This command sets XON/XOFF handshaking when the instrument is transmitting data. The choices are XON and None.

### **Command Query**

`:SYSTEM:COMMunicate:SERial:TRANsmit:PACE?`

### **Initial Values**

This is a persistent state, set to None at the factory.

---

## **:DIAGnostic Subsystem**

The diagnostic subsystem is used to set the controls and the parameters associated with instrument operational and tracking data.

### **Attenuator Cycle Information**

#### **Command Mnemonic**

**:DIAGnostic:INFORMation:CCOunt:ATTenuator?**

#### **Command Description**

This query returns the number of times that the attenuator has been switched.

### **Power Cycle Information**

#### **Command Mnemonic**

**:DIAGnostic:INFORMation:CCOunt:PON?**

#### **Command Description**

This query returns the number of times that the sources line power has been cycled.

### **Reverse Power Protection Trips Information**

#### **Command Mnemonic**

**:DIAGnostic:INFORMation:CCOunt:PROTection?**

#### **Command Description**

This query returns the number of times that the reverse power protection circuitry has been activated.

### **Display Time-On Information**

#### **Command Mnemonic**

**:DIAGnostic:INFORMation:DISPlay:OTIME?**

### **Command Description**

This query returns the number of hours that the source's display has been activated.

### **Option Information**

#### **Command Mnemonic**

:DIAGnostic:INfOrMation:OPTions?

#### **Command Description**

This query returns a list of instrument options.

### **Instrument Time-On Information**

#### **Command Mnemonic**

:DIAGnostic:INfOrMation:OTIME?

#### **Command Description**

This query returns the number of hours that the source has had its line power activated.

### **Instrument Serial Number and Firmware Information**

#### **Command Mnemonic**

:DIAGnostic:INfOrMation:SDATe?

#### **Command Description**

This query returns the instrument's serial number, firmware revision number, and firmware revision date.

---

## :DISPlay Subsystem

The display subsystem is used to set the controls and the parameters associated with the signal source's LCD display.

### Configure Display Brightness

#### Command Mnemonic

`:DISPlay:BRIGhtness <val>`

#### Command Description

This command sets the brightness of the instrument's LCD display. Choices for the variable `<val>` range from 0.0 (dimmiest) to 1.0 (brightest).

#### Command Query

`:DISPlay:BRIGhtness?`

#### Initial Values

This is a persistent state, set to 1 at the factory.

### Configure Display Contrast

#### Command Mnemonic

`:DISPlay:CONTRast <val>`

#### Command Description

This command sets the contrast of the of the instrument's LCD display. Choices for the variable `<val>` range from 0.0 to 1.0.

#### Command Query

`:DISPlay:CONTRast?`

#### Initial Values

This is a persistent state set to 0.768 at the factory.



## **Configure Display Inverse Video**

### **Command Mnemonic**

**:DISPlay:INVerse ON|OFF|1|0**

### **Command Description**

This command sets the source's display to inverse video mode. Choices are On (1) or Off (0).

### **Command Query**

**:DISPlay:INVerse?**

### **Initial Values**

This is a persistent state set to Off at the factory.

---

## :FM Subsystem

The frequency modulation subsystem is used to set the modulation controls and the parameters associated with frequency modulated signals.

### External Frequency Modulation Source Coupling

#### Command Mnemonic

```
:FM[1]|2:EXtErnal[1]|2:COUPling AC|DC
```

#### Command Description

This command sets the external coupling for the frequency modulation source. The choices are AC or DC coupling. This command does not change the currently active source, nor does it switch the current modulation on or off. The modulating signal may be the sum of several signals, either internal or external sources. If the coupling is set to DC, then both the AC and DC signal components pass. AC coupling passes only AC signal components.

The external 1 source has multiple uses but can be used for only one modulation at a time. If, for example, you were using the external 1 source in an amplitude modulation configuration for **AM Path 1**, and then you configured **FM Path 2** to also use the external 1 source, the signal generator would turn off **AM Path 1** and assign the external 1 source to your **FM Path 2** configuration. Notice that for these purposes, the external 1 AC-coupled source is the same as the external 1 DC-coupled source.

#### Command Query

```
:FM[1]|2:EXtErnal[1]|2:COUPling?
```

#### Initial Values

At normal preset, this value is set to DC.

At \*RST, this value is set to DC.

At preset power-up, this value is set to DC.

### Internal Frequency Modulation Source Rate

#### Command Mnemonic

```
:FM[1]|2:INTErnal[1]:FREQuency <val><unit>
```

### Command Description

This command sets the internal modulation frequency for the **FM Path 1** and **FM Path 2** configurations. The current value for FM rate is displayed in the active entry area. The range of values allowed is 0.1 Hz to 10 kHz. (0.1 Hz to 50 kHz is the range allowed if sinewave is selected as the internal waveform.) The minimum increment allowed is 0.1 Hz. Notice that the new value of FM rate applies only to whichever FM path configuration you have currently selected.

### Command Query

**:FM[1] | 2:INTernal:FREQUENCY?**

### Initial Values

At normal preset, this value is set to 400.0 Hz.

At \*RST, this value is set to 400.0 Hz.

At preset power-up, this value is set to 400.0 Hz.

## Internal Frequency Modulation Alternate Frequency

### Command Mnemonic

**:FM[1] | 2:INTernal[1]:FREQUENCY:ALternate <val><unit>**

### Command Description

This command sets the frequency for the alternate signal. The alternate frequency is the second frequency of a dual-sine or the stop frequency of a swept-sine. The choices for the variables **<val>** and **<unit>** range from 0.1 kHz (minimum) to 50.0 kHz (maximum).

### Command Query

**:FM[1] | 2:INTernal:FREQUENCY:ALternate?**

### Initial Values

At normal preset, this value is set to 400.00 Hz.

At \*RST, this value is set to 400.00 Hz.

At preset power-up, this value is set to 400.00 Hz.

## Internal Frequency Modulation Alternate Frequency Amplitude

### Command Mnemonic

```
:FM[1] | 2:INTernal[1]:FREQuency:ALternate:AMPLitude:PERCent <val><unit>
```

### Command Description

This command sets the amplitude of the alternate frequency as a percentage of the total amplitude. Therefore, if the alternate frequency makes up 30% of the total amplitude, then the primary frequency is 70% of the total amplitude (for dual-sine only). The choices for the variables <val> and <unit> range from 0.1% (minimum) to 99.9% (maximum).

### Command Query

```
:FM[1] | 2:INTernal:FREQuency:ALternate:AMPLitude:PERCent?
```

### Initial Values

At normal preset, this value is set to 50%.

At \*RST, this value is set to 50%.

At preset power-up, this value is set to 50%.

## Internal Frequency Modulation Waveform

### Command Mnemonic

```
:FM[1] | 2:INTernal[1]:FUNctIon:SHApe  
SINE | TRIangle | SQUare | RAMP | NOISE | DUALsine | SWEPTsine
```

### Command Description

This command allows you to assign an FM waveform to your **FM Path 1** and **FM Path 2** configurations. Select from sine, triangle, square, ramp, noise, dual sine, and swept sine waveforms. Notice that your waveform selection applies only to whichever AM path configuration you have currently selected.

Using the command `:FM[1] | 2:INTernal[1]:FUNctIon:SHApe SINE` lets you specify sine as the amplitude modulation waveform for the **FM Path 1** and **FM Path 2** configurations. Notice that the selected waveform applies only to whichever FM path configuration you have currently selected.

Using the command `:FM[1]|2:INTERNAL[1]:FUNCTION:SHAPE TRIangle` lets you specify triangle as the amplitude modulation waveform for the **FM Path 1** and **FM Path 2** configurations. Notice that the selected waveform applies only to whichever FM path configuration you have currently selected.

Using the command `:FM[1]|2:INTERNAL[1]:FUNCTION:SHAPE SQUARE` lets you specify square as the amplitude modulation waveform for the **FM Path 1** and **FM Path 2** configurations. Notice that the selected waveform applies only to whichever FM path configuration you have currently selected.

Using the command `:FM[1]|2:INTERNAL[1]:FUNCTION:SHAPE RAMP` lets you specify ramp as the amplitude modulation waveform for the **FM Path 1** and **FM Path 2** configurations. Notice that the selected waveform applies only to whichever FM path configuration you have currently selected.

Using the command `:FM[1]|2:INTERNAL[1]:FUNCTION:SHAPE NOISE` lets you specify noise as the amplitude modulation waveform for the **FM Path 1** and **FM Path 2** configurations. Notice that the selected waveform applies only to whichever FM path configuration you have currently selected.

Using the command `:FM[1]|2:INTERNAL[1]:FUNCTION:SHAPE DUALsine` allows you to set the dual sine as the amplitude modulation waveform for the **FM Path 1** and **FM Path 2** configurations. In this mode you can set the FM rates for two separate tones. In addition, you can set a percentage of the tone 2 FM depth to the total FM depth.

Using the command `:FM[1]|2:INTERNAL[1]:FUNCTION:SHAPE SWEPTSine` allows you to set the swept-sine amplitude modulation waveform for the **FM Path 1** and **FM Path 2** configurations. In this mode you can set the start and stop FM rate and the sweep time. You can set the signal generator to a single, externally-triggered sweep on either a negative or positive TTL level or you can choose continuous sweep, triggered immediately. In this menu you can also select either a positive or negative polarity for the TRIGGER OUT signal.

### Command Query

`:FM[1]|2:INTERNAL:FUNCTION:SHAPE?`

### Initial Values

At normal preset, this value is set to Sine.

At \*RST, this value is set to Sine.

At preset power-up, this value is set to Sine.

## Internal Frequency Modulation Sweep Time

### Command Mnemonic

`:FM[1]|2:INTERNAL[1]:SWEep:TIME <val><unit>`

### Command Description

This command selects the sweep time for a swept-sine, internally-generated signal.

### Command Query

:FM[1] | 2:INTERNAL:SWEep:TIME?

### Initial Values

At normal preset, this value is set to 100 msec.

At \*RST, this value is set to 100 msec.

At preset power-up, this value is set to 100 msec.

## Internal Frequency Modulation Sweep Trigger

### Command Mnemonic

:FM[1] | 2:INTERNAL[1]:SWEep:TRIGger IMMEDIATE|BUS|EXTERNAL|KEY

### Command Description

This command selects the trigger for the frequency modulation sweep.

### Command Query

:FM[1] | 2:INTERNAL:SWEep:TRIGger?

### Initial Values

At normal preset, this value is set to Immediate.

At \*RST, this value is set to Immediate.

At preset power-up, this value is set to Immediate.

## Frequency Modulation Source

### Command Mnemonic

:FM[1] | 2:SOURce INT[1]|EXT1|EXT2

### Command Description

This command sets the source that will generate the frequency modulation. The choices are Internal Source 1, External Source 1, or External Source 2.

You can choose internally-generated frequency modulation or select an externally-applied signal from either the EXT 1 INPUT or EXT 2 INPUT connectors. The internal modulation is always AC-coupled. For the externally-applied signals, you can choose between AC- and DC-coupled modulation. A 1.0 Vpk input is required for calibrated FM deviation settings. The **EXT 1 LO/HI** and **EXT 2 LO/HI** display annunciators will turn on if the peak input voltage differs from 1.0 Vpk by more than 3%. (The LO/HI annunciators only function for AC-coupled external inputs.)

The internal and external 1 and 2 sources have multiple uses. You can use them for amplitude, frequency, and phase modulation. However, any given source can only be routed to one enabled modulation at a time. If, for example, you were using the external 1 source in a frequency modulation configuration for **FM Path 1**, and then you configured **AM Path 2** to also use the external 1 source, the signal generator would turn off **FM Path 1** and assign the external 1 source to your **AM Path 2** configuration. Notice that for these purposes the external 1 AC-coupled source is the same as the external 1 DC-coupled source and the external 2 AC-coupled source is the same as the external 2 DC-coupled source.

### Command Query

```
:FM[1] | 2 :SOURce?
```

### Initial Values

At normal preset, this value is set to Internal.

At \*RST, this value is set to Internal.

At preset power-up, this value is set to Internal.

## Frequency Modulation State

### Command Mnemonic

```
:FM[1] | 2 :STATe ON|OFF|1|0
```

### Command Description

This command toggles the frequency modulation on or off for whichever FM path configuration (**FM Path 1** or **FM Path 2**) you have selected. The choices are On (1) or Off (0). Notice, however that although you can turn on frequency modulation with this command, the RF carrier is modulated by the enabled modulation only when you have also set **Mod On/Off** to **On**. Whenever frequency modulation is enabled, the **FM** annunciator is turned on in the display.

There are two paths for FM modulation which can be simultaneously enabled as long as they use different sources (Int, Ext1, or Ext2). **FM Path 2** is limited to a maximum rate of 1 MHz. **FM Path 2** must be set to a deviation less than or equal to **FM Path 1**. The modulation signals from both paths are summed internally for composite modulation.

### Command Query

:FM[1] | 2:STATe?

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

## Frequency Modulation Deviation

### Command Mnemonic

:FM[1] | 2[:DEVIation] <val><unit>

### Command Description

Use this command to set the frequency modulation deviation for the **FM Path 1** and **FM Path 2** configurations. The current value for FM deviation is displayed in the active entry area. The range of values allowed depends on the carrier frequency. The maximum peak deviation for a frequency is calculated by multiplying N times 10 MHz. (The following table lists the values for N and the resulting maximum peak deviations.)

Carrier Frequency	N	Maximum Peak Deviation
250 kHz to ≤ 249.999 MHz	1	10 MHz
> 249.999 MHz to ≤ 500 MHz	0.5	5 MHz
> 500 MHz to ≤ 1 GHz	1	10 MHz
> 1 GHz to ≤ 2 GHz	2	20 MHz
> 2 GHz to 4 GHz	4	40 MHz

For example, if you choose a carrier frequency of 400 MHz, multiply 0.5 times 10 MHz. This results in a 5 MHz maximum peak deviation.

Notice that the new value of FM deviation applies only to whichever FM path configuration you have currently selected. Also, whenever **FM Path 1** is used with **FM Path 2**, the deviation for **FM Path 1** must be greater than or equal to the deviation for **FM Path 2**.



### Command Query

`:FM[1]|2[:DEVIation]?`

### Initial Values

At normal preset, this value is set to 1.0 kHz.

At \*RST, this value is set to 1.0 kHz.

At preset power-up, this value is set to 1.0 kHz.

## Frequency Modulation Deviation Coupling

### Command Mnemonic

`:FM[1]|2[:DEVIation]:TRACk ON|OFF|1|0`

### Command Description

This command toggles the FM deviation coupling on and off. Turning on FM deviation coupling links the FM deviation values of **FM Path 1** and **FM Path 2**. When the values are coupled, any change you make to one FM deviation value is applied to both FM deviation values.

### Command Query

`:FM[1]|2[:DEVIation]:TRACk?`

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

---

## :FREQUENCY Subsystem

The frequency subsystem is used to set the controls and the parameters associated with carrier signal frequency.

### Fixed Frequency

#### Command Mnemonic

**:FREQUENCY:FIXED <val><unit>**

#### Command Description

This command sets the carrier frequency for the signal generator's RF output. The choices for the variables (depending on your signal generator's model number) **<val>** and **<unit>** range from 100 kHz (minimum) to 1 GHz, 2 GHz, 3 GHz, or 4 GHz (maximum).

#### Command Query

**:FREQUENCY:FIXED?**

#### Initial Values

At normal preset, this value is set to the signal generator's maximum specified output frequency.

At \*RST, this value is set to the signal generator's maximum specified output frequency.

At preset power-up, this value is set to the signal generator's maximum specified output frequency.

### Frequency Mode

#### Command Mnemonic

**:FREQUENCY:MODE CW|FIXED|LIST**

#### Command Description

This command sets the frequency mode of the signal generator. The choices are CW or Fixed, and List.

#### Command Query

**:FREQUENCY:MODE?**

### Initial Values

At normal preset, this value is set to CW.

At \*RST, this value is set to CW.

At preset power-up, this value is set to CW.

## Frequency Multiplier

### Command Mnemonic

**:FREQuency:MUlTiplier <val>**

### Command Description

This command sets the multiplier for the signal generator's carrier frequency. The choices for the variable **<val>** are integers between 1 and 50.

You can multiply the frequency shown on the display without changing the frequency output at the RF OUTPUT connector (simulating the frequency at the output of a harmonic multiplier). For example, set the output frequency to 1 MHz. Then send the command **:FREQuency:MUlTiplier 3**. The display will now show an output frequency of 3 MHz but the actual output frequency will remain at 1 MHz. For any multiplier greater than 1, the **MULT** indicator is shown in the frequency area of the display.

### Command Query

**:FREQuency:MUlTiplier?**

### Initial Values

At normal preset, this value is set to 1.

At \*RST, this value is set to 1.

At preset power-up, this value is set to 1.

## Frequency Offset

### Command Mnemonic

**:FREQuency:OFFSet <val><unit>**

### Command Description

This command sets the frequency offset. The choices for the variables **<val>** and **<unit>** are frequencies between 0.0 Hz and 200.00 GHz.

A frequency offset changes the value shown in the frequency area of the display but does not affect the output frequency. For example, if the current output frequency is 1 MHz and you enter a frequency offset of 3 MHz, the output frequency will remain at 1 MHz but the display will show a frequency of 4 MHz. This feature lets you simulate the frequency at the output of a frequency translating device.

A frequency offset can be entered at any time during normal operation and also when you are operating in frequency reference mode. When an offset has been entered, the **OFFSET** indicator is turned on in the frequency area of the display.

### Command Query

**:FREQUENCY:OFFSET?**

### Initial Values

At normal preset, this value is set to 0.0 Hz.

At \*RST, this value is set to 0.0 Hz.

At preset power-up, this value is set to 0.0 Hz.

## Frequency Reference

### Command Mnemonic

**:FREQUENCY:REFERENCE <val>**

### Command Description

This command sets the current output frequency as a frequency reference value. The choices for the variable **<val>** are frequencies between 0.0 Hz and the signal generator's maximum specified output frequency. All frequency parameters are then set as relative to the reference value.

### Command Query

**:FREQUENCY:REFERENCE?**

### Initial Values

At normal preset, this value is set to 0.0 Hz.

At \*RST, this value is set to 0.0 Hz.

At preset power-up, this value is set to 0.0 Hz.

## Frequency Reference State

### Command Mnemonic

**:FREQuency:REFeRence:StAte ON|OFF|1|0**

### Command Description

This command toggles the frequency reference mode on and off. When frequency reference mode is turned on, the frequency value displayed is equal to the current hardware output frequency minus the reference value set by the **:FREQuency:REFeRence <val>** command. (This command sets the reference value equal to the current output frequency. If you have not yet set the reference value, the preset value for frequency reference is 0 Hz.) The **REF** indicator is turned on in the frequency area of the display. All frequency parameters will now be set as relative to the reference value.

Frequency offsets can be used with frequency reference mode. In this situation, the display will show the frequency calculated as the current hardware output frequency minus the reference value plus the frequency offset.

Frequency reference mode only changes the display; it does not change the RF output frequency. For example, if you set your RF output frequency to 700 MHz, set your reference value to 700 MHz, and then turn frequency relative mode on, your display shows your frequency as 0 Hz but your output frequency remains at 700 MHz.

### Command Query

**:FREQuency:REFeRence:StAte?**

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

## Start Frequency

### Command Mnemonic

**:FREQUENCY:START <val><unit>**

### Command Description

This command sets the frequency of the first point in the sweep. The choices for are the variables **<val>** and **<unit>** may range from 100.0 kHz to the signal generator's maximum specified frequency.

### Command Query

**:FREQUENCY:START?**

### Initial Values

At normal preset, this value is set to the signal generator's maximum specified frequency.

At \*RST, this value is set to the signal generator's maximum specified frequency.

At preset power-up, this value is set to the signal generator's maximum specified frequency.

## Stop Frequency

### Command Mnemonic

**:FREQUENCY:STOP <val><unit>**

### Command Description

This command sets the frequency of the last point in the sweep. The choices for are the variables **<val>** and **<unit>** may range from 100.0 kHz to the signal generator's maximum specified frequency.

### Command Query

**:FREQUENCY:STOP?**

### Initial Values

At normal preset, this value is set to the signal generator's maximum specified frequency.

At \*RST, this value is set to the signal generator's maximum specified frequency.

At preset power-up, this value is set to the signal generator's maximum specified frequency.

## Frequency Optimization

### Command Mnemonic

**:FREQUENCY:SYNTHESIS <val>**

### Command Description

This command sets the PLL (Phase Lock Loop) bandwidth for optimizing phase noise. The choices for the variable <val> may range from 1 (offsets below 10 kHz) to 2 (offsets above 10 kHz).

### Command Query

**:FREQUENCY:SYNTHESIS?**

### Initial Values

At normal preset, this value is set to 2.

At \*RST, this value is set to 2.

At preset power-up, this value is set to 2.

## Continuous Wave Frequency

### Command Mnemonic

**:FREQUENCY[:CW] <val><unit>**

**:FREQUENCY:FIXED <val><unit>**

### Command Description

This command sets the signal generator's CW output frequency. The choices for are the variable <val> may range from 100 kHz to the signal generator's maximum specified output frequency.

The current RF output frequency is always shown in the frequency area of the display (unless you have altered the display by turning on frequency reference mode, entering an offset or a multiplier, or if a frequency sweep is selected). Use this command to change the RF output frequency. Frequency becomes the active function and the current value is also shown in the active entry area of the display.

### Command Query

**:FREQUENCY[:CW]?**

**:FREQUENCY:FIXED?**

### Initial Values

At normal preset, this value is set to the signal generator's maximum specified output frequency.

At \*RST, this value is set to the signal generator's maximum specified output frequency.

At preset power-up, this value is set to the signal generator's maximum specified output frequency.

## Set Phase Reference

### Command Mnemonic

**:PHASe:REFerence**

### Command Description

This command sets the current output phase as a zero reference. Phase adjustments are then set as relative to this zero reference.

## Phase Adjustment

### Command Mnemonic

**:PHASe[:ADJust] <val><unit>**

### Command Description

This command adjusts the phase of the modulating signal. The choices for the variables **<val>** and **<unit>** may range plus or minus 1.416 radians from the initial value.

### Command Query

**:PHASe[:ADJust]?**

### Initial Values

At normal preset, this value is set to 0.000.

At \*RST, this value is set to 0.000.

At preset power-up, this value is set to 0.000.

## Reference Oscillator Source Query

### Command Mnemonic

**:ROSCillator:SOURce?**



### Command Description

This command queries the source of the signal generator's reference oscillator. It returns either **INT** or **EXT**.

---

## :LFOutput Subsystem

The low frequency output subsystem is used to set the controls and the parameters associated with the low frequency output signals.

### Low Frequency Output Amplitude

#### Command Mnemonic

`:LFOutput:AMPLitude <val><unit>`

#### Command Description

This command scales the output of the signal at the LF OUTPUT connector on the front panel. The choices for the variables `<val>` and `<unit>` range from 0.0 volts (minimum) to 5.0 volts peak (maximum).

#### Command Query

`:LFOutput:AMPLitude?`

#### Initial Values

At normal preset, this value is set to 0.0 volts.

At \*RST, this value is set to 0.0 volts.

At preset power-up, this value is set to 0.0 volts.

### Low Frequency Output Frequency

#### Command Mnemonic

`:LFOutput:FUNCTION:FREQUENCY <val><unit>`

#### Command Description

This command sets the modulating frequency for the LF output signal when you have selected the function generator as the internal source. The choices for the variables `<val>` and `<unit>` range from 0.1 Hz (minimum) to 10.0 kHz (maximum). (0.1 Hz to 50 kHz is the range allowed if sinewave is selected as the internal waveform.)

#### Command Query

`:LFOutput:FUNCTION:FREQUENCY?`

### Initial Values

At normal preset, this value is set to 400.0 Hz.

At \*RST, this value is set to 400.0 Hz.

At preset power-up, this value is set to 400.0 Hz.

## Low Frequency Output, Alternate Frequency

### Command Mnemonic

```
:LFOutput:FUNCTION:FREQUENCY:ALternate <val><unit>
```

### Command Description

This command sets the frequency for the alternate LF output signal when you have selected the function generator as the internal source. The alternate frequency is the second frequency of a dual-sine or the stop frequency of a swept-sine. The choices for the variables <val> and <unit> range from 0.1 kHz (minimum) to 50.0 kHz (maximum).

### Command Query

```
:LFOutput:FUNCTION:FREQUENCY:ALternate?
```

### Initial Values

At normal preset, this value is set to 400.0 Hz.

At \*RST, this value is set to 400.0 Hz.

At preset power-up, this value is set to 400.0 Hz.

## Low Frequency Output, Alternate Frequency Amplitude

### Command Mnemonic

```
:LFOutput:FUNCTION:FREQUENCY:ALternate:AMPLitude:PERCent <val><unit>
```

### Command Description

This command sets the amplitude of the alternate frequency as a percentage of the total LF output amplitude. Therefore, if the alternate frequency makes up 30% of the total amplitude, then the primary frequency is 70% of the total LF output amplitude. The choices for the variables <val> and <unit> range from 0.1% (minimum) to 99.9% (maximum).

### Command Query

```
:LFOutput:FUNCTION:FREQUENCY:ALternate:AMPLitude:PERCent?
```

### Initial Values

At normal preset, this value is set to 50%.

At \*RST, this value is set to 50%.

At preset power-up, this value is set to 50%.

## Configure Function Generator Pulse Period

### Command Mnemonic

```
:LFOutput:FUNCTION:PERiod <val><unit>
```

### Command Description

This command sets the period for the internally-generated pulse modulation source. The choices for the variables <val> and <unit> range from 16 microseconds to 30 seconds.

### Command Query

```
:LFOutput:FUNCTION:PERiod?
```

### Initial Values

At normal preset, this value is set to 80.0 usec.

At \*RST, this value is set to 80.0 usec.

At preset power-up, this value is set to 80.0 usec.

## Configure Function Generator Pulse Width

### Command Mnemonic

```
:LFOutput:FUNCTION:PWIDth <val><unit>
```

### Command Description

This command sets the pulse width for the internally-generated pulse source. The choices for the variables <val> and <unit> range from 8.0 to 30.0 microseconds.

### Command Query

**:LFOutput:FUNCTION:PWIDth?**

### Initial Values

At normal preset, this value is set to 40.0 usec.

At \*RST, this value is set to 40.0 usec.

At preset power-up, this value is set to 40.0 usec.

## Low Frequency Output Waveform

### Command Mnemonic

**:LFOutput:FUNCTION:SHAPE**  
**SINE | DUALsine | SWEPTsine | TRIangle | SQUARE | RAMP | PULSE | NOISE | DC**

### Command Description

This command selects the waveform for the low frequency output signal. The choices are Sine, Dual-sine, Swept-sine, Triangle, Ramp, Square, Pulse, Noise, and DC waveforms.

### Command Query

**:LFOutput:FUNCTION:SHAPE?**

### Initial Values

At normal preset, this value is set to Sine.

At \*RST, this value is set to Sine.

At preset power-up, this value is set to Sine.

## Function Generator Sweep Time

### Command Mnemonic

**:LFOutput:FUNCTION:SWEEP:TIME <val><unit>**

### Command Description

This command selects the sweep time for the function generator.

### Command Query

`:LFOutput:FUNCTION:SWEep:TIME?`

### Initial Values

At normal preset, this value is set to 100 msec.

At \*RST, this value is set to 100 msec.

At preset power-up, this value is set to 100 msec.

## Function Generator Sweep Trigger

### Command Mnemonic

`:LFOutput:FUNCTION:SWEep:TRIGger IMMEDIATE|KEY|EXTERNAL|BUS`

### Command Description

This command selects the trigger for the function generator sweep.

### Command Query

`:LFOutput:FUNCTION:SWEep:TRIGger?`

### Initial Values

At normal preset, this value is set to Immediate.

At \*RST, this value is set to Immediate.

At preset power-up, this value is set to Immediate.

## Low Frequency Output Source

### Command Mnemonic

`:LFOutput:SOURce INT[1]|FUNCTION`

### Command Description

This command sets the low frequency source. The choices are Internal or Function Generator. When set to Function Generator, you can select a frequency and shape in addition to selecting the amplitude for the signal that is output at the LF OUTPUT front panel connector. Any modulation with the internal source selected is turned off when the function generator is selected.

When set to Internal, this command allows you to output a signal at the LF OUTPUT connector where the frequency and shape of the signal is set by the internal source as it is being used by a modulation. For example, if the internal source is currently assigned to an AM path configuration and AM is turned on, the signal output at the LF OUTPUT connector will have the frequency and shape of the amplitude modulating signal.

For internal square pulse modulation, the internal source is a sinewave which is later squared by the modulator to generate the pulse squarewave. The LF OUTPUT signal for this internal source, therefore, is a sinewave. For internal pulse modulation, a true variable-width pulse will be seen on the LF OUTPUT.

### Command Query

`:LFOutput:SOURce?`

### Initial Values

At normal preset, this value is set to Internal.

At \*RST, this value is set to Internal.

At preset power-up, this value is set to Internal.

## Low Frequency Output State

### Command Mnemonic

`:LFOutput:SOURce:STATe ON|OFF|1|0`

### Command Description

This command sets the low frequency source state. The choices are On (1) or Off (0).

### Command Query

`:LFOutput:SOURce:STATe?`

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

---

## :LIST Subsystem

The list subsystem is used to set the controls and the parameters associated with frequency and/or power sweeps.

### List Direction

#### Command Mnemonic

`:LIST:DIRection UP|DOWN`

#### Command Description

This command sets the direction of execution of a list sweep. The choices are Up or Down. Choose **UP** to sweep from the first point in the list to the last point, or from the step sweep start frequency and amplitude to the stop frequency and amplitude. Choose **DOWN** to reverse the direction of the sweep.

#### Command Query

`:LIST:DIRection?`

#### Initial Values

At normal preset, this value is set to Up.

At \*RST, this value is set to Up.

At preset power-up, this value is set to Up.

### Dwell List

#### Command Mnemonic

`:LIST:DWELl <val>{, <val>}`

#### Command Description

This command defines a list of dwell times. The list is stored in a file (DWEL\_FILE) which is only re-initialized upon power-up preset. Each value specifies the minimum time that the sweep engine waits at that point's frequency and/or power setting before going to the next setting. Dwell times are only used when the point trigger source is set to Immediate.



### Command Query

`:LIST:DWELL?`

## Dwell List Type

### Command Mnemonic

`:LIST:DWELL:TYPE LIST|STEP`

### Command Description

This command sets the type of dwell for a list sweep. The choices are List or Step. Use this command to toggle the dwell time for the list sweep points between the values defined in the list sweep and the value set for step sweep. Choose **LIST** to sweep with dwell times that you have defined in the list sweep. Choose **STEP** to sweep each point in the list with a dwell time set via the `:SWEep:DWELL <val>` command.

### Command Query

`:LIST:DWELL:TYPE?`

### Initial Values

At normal preset, this value is set to List.

At \*RST, this value is set to List.

At preset power-up, this value is set to List.

## Dwell List Points Query

### Command Mnemonic

`:LIST:DWELL:POINTs?`

### Command Description

This command queries the signal generator for the number of points in the dwell list.

## Frequency List

### Command Mnemonic

`:LIST:FREQuency <val>{, <val>}`

### **Command Description**

This command defines the frequency list. Each value specifies the frequency that the sweep engine uses when sweeping frequency. This list is stored in a file (FREQ\_FILE), which is only re-initialized upon power-up preset.

### **Command Query**

**:LIST:FREQuency?**

## **Frequency List Points Query**

### **Command Mnemonic**

**:LIST:FREQuency:POINTs?**

### **Command Description**

This command queries the signal generator for the number of points in the frequency list.

## **Manual Point**

### **Command Mnemonic**

**:LIST:MANual <val>**

### **Command Description**

This command sets the current element used by the list mode. If list mode is controlling frequency and/or power then the indexed element in the respective list(s) will be used. If the list mode is AUTO, then nothing will happen until the mode is changed to MANUAL. If the point selected is beyond the length of the longest enabled list, then the point will be set to the maximum possible point, and an error will be generated.

### **Command Query**

**:LIST:MANual?**

### **Initial Values**

At normal preset, this value is set to 1.

At \*RST, this value is set to 1.

At preset power-up, this value is set to 1.

## List Mode

### Command Mnemonic

`:LIST:MODE AUTO|MANual`

### Command Description

This command sets the operating mode of a list sweep. The choices are Auto or Manual. When in manual mode, the selected point controls the frequency and amplitude according to the sweep type.

### Command Query

`:LIST:MODE?`

### Initial Values

At normal preset, this value is set to Auto.

At \*RST, this value is set to Auto.

At preset power-up, this value is set to Auto.

## Power List

### Command Mnemonic

`:LIST:POWer <val>{, <val>}`

### Command Description

This command defines the power list. Each value specifies the power level that the instrument uses when it is sweeping. This list is stored in a file (POW\_FILE), which is only re-initialized upon power-up preset.

### Command Query

`:LIST:POWer?`

## Power List Points Query

### Command Mnemonic

`:LIST:POWer:POINTs?`

### Command Description

This command queries the signal generator for the number of points in the power list.

## List Trigger Source

### Command Mnemonic

**:LIST:TRIGger:SOURce** BUS|IMMediate|EXTernal|KEY

### Command Description

This command sets the point-to-point triggering source for the sweep. The choices are Bus (trigger a list or step sweep using the HP-IB), Immediate (immediately triggers the current sweep), External (trigger a list or step sweep on a signal applied to the TRIGGER IN rear panel connector), or Key (triggers the step when you press the **Trigger** front panel key). The sweep must be initiated (and the sweep trigger received) before the point-to-point trigger can cause the sweep to go to the next point.

### Command Query

**:LIST:TRIGger:SOURce?**

### Initial Values

At normal preset, this value is set to Immediate.

At \*RST, this value is set to Immediate.

At preset power-up, this value is set to Immediate.

## List Type

### Command Mnemonic

**:LIST:TYPE** LIST|STEP

### Command Description

This command sets the type of sweep. The choices are List and Step. You create a list sweep by supplying the frequency, amplitude, and dwell time for each point in the sweep. You create a step sweep by supplying the start and stop frequency and amplitude, the number of points, and a dwell time. The signal generator then determines the values of each point between the first and last point in a linear manner.

### Command Query

**:LIST:TYPE?**

### Initial Values

At normal preset, this value is set to List.

Language Reference  
:LIST Subsystem

At \*RST, this value is set to List.

At preset power-up, this value is set to List.

---

## :MEMory and :MMEMory Subsystems

The memory and mass memory subsystems are used manage and access instrument memory and mass storage.

### Binary Memory Catalog

#### Command Mnemonic

**:MEMory:CATalog:BINary?**

#### Command Description

This command outputs a list of the binary files in the “:” directory. The return data will be in the following form:

```
<mem used>,<mem free> {,<file listing>}
```

The signal generator will return the two memory usage parameters and as many file listings as there are files in the directory list. Each file listing parameter will be in the following form:

```
<filename>,<file type>,<file size>
```

The file types are:

- BINary - A binary file
- STATe - A state file
- LIST - A sweep list file

### List Memory Catalog

#### Command Mnemonic

**:MEMory:CATalog:LIST?**

#### Command Description

This command outputs a list of sweep lists files in the “:” directory. The return data will be in the following form:

```
<mem used>,<mem free> {,<file listing>}
```

The signal generator will return the two memory usage parameters and as many file listings as there are files in the directory list. Each file listing parameter will be in the following form:

`<filename>,<file type>, <file size>`

The file types are:

- BINary - A binary file
- STATE - A state file
- LIST - A sweep list file

## State Memory Catalog

### Command Mnemonic

`:MEMory:CATalog:STATe?`

### Command Description

This command outputs a list of the state files in the “:” directory. The return data will be in the following form:

`<mem used>,<mem free> {,<file listing>}`

The signal generator will return the two memory usage parameters and as many file listings as there are files in the directory list. Each file listing parameter will be in the following form:

`<filename>,<file type>, <file size>`

The file types are:

- BINary - A binary file
- STATE - A state file
- LIST - A sweep list file

## All Memory Catalog

### Command Mnemonic

`:MEMory:CATalog[:ALL]?`

`:MMEMory:CATalog[:ALL]? "<file systems>"`

### Command Description

This command outputs all lists of the files in the specified memory subsystem “:” for memory :CATalogue. The return data will be in the following form:

`<mem used>,<mem free> {,<file listing>}`

The signal generator will return the two memory usage parameters and as many file listings as there are files in the directory list. Each file listing parameter will be in the following form:

`<filename>,<file type>, <file size>`

The file types are:

- BINary - A binary file
- STATe - A state file
- LIST - A sweep list file

## Copy Files

### Command Mnemonic

`:MEMory:COPY[:NAME]<filename>, <filename>`

`:MMEMory:COPY <msus>, <msus>`

### Command Description

This command makes duplicates of the requested files. The variable `<msus>` represents `[<file_system>:]<filename>`.

## Memory Data Load

### Command Mnemonic

`:MEMory:DATA <filename>,<datablock>`

`:MMEMory:DATA <msus>, <datablock>`

### Command Description

This command loads `<datablock>` into the memory location `<filename>` or `<msus>`. The variable `<msus>` represents `[<file_system>:]<filename>`.



## Memory Filename Query

### Command Mnemonic

**:MEMory:DATA? <filename>**

**:MMEMory:DATA? <msus>**

### Command Description

This command returns the <datablock> associated with the <filename> or <msus>. The variable <msus> represents [**<file\_system>:**]**<filename>**.

## Block Pattern RAM

### Command Mnemonic

**:MEMory:DATA:PRAM:BLOCK <datablock>**

### Command Description

This command allows writing a block of data directly to the baseband generator pattern memory. The loaded block data provides control of the data and burst of the modulated signal. Any other command which loads the pattern RAM will destroy the user data.

## List Pattern RAM

### Command Mnemonic

```
:MEMory:DATA:PRAM:LIST <value> [,<value>,<...>]
```

### Command Description

This command allows writing a list of data values (bytes) directly into the baseband generator pattern memory. The loaded block data provides control of the data and burst of the modulated signal. Any other command which loads the pattern RAM will destroy the user data. The <value> can be between 0 and 255. The entire data generator memory is controlled by this command. You must first activate a mode either using the front panel keys, or by using a SCPI command to set up the the basebasnd generator and data generator clock rate.

Bit 0 (1)	data value: 0 or 1 as required for a data bit.
Bit 1 (2)	Always 0
Bit 2 (4)	Burst control: 0 for burst off, 1 for burst on. All data values that require power out must have this bit on.
Bit 3 (8)	Always 0
Bit 4 (16)	Always 1
Bit 5 (32)	Always 0
Bit 6 (64)	Event 1 control: 0 or 1, as desired on the EVENT1 output.
Bit 7 (128)	Pattern reset: Reset the pattern to start after this entry is processed.

For example: to control a burst and have a FIX4 of 1100 duration burst, the command may look like this:

```
mem:data:pram:list  
21,21,20,20,21,21,20,20,21,21,20,20,21,21,20,2,21,21,20,20,0,0,0,0,0,0,0,0,0,0,  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,128
```

This will produce a repeating squarewave pattern: RF on and modulated (21,21,20,20,...), RF off (0,0,0,0,...), and repeat pattern (128).

## Delete All

---

**CAUTION:**

---

Using this command can be catastrophic to the functionality of the signal generator. Use caution when applying this command.

### Command Mnemonic

**:MEMory:DELeTe:ALL**

### Command Description

This command clears the user file system of ALL files. This includes the “:”, “LIST:” and “STATE:” directories.

## Delete Binary

### Command Mnemonic

**:MEMory:DELeTe:BINary**

### Command Description

This command deletes all binary files.

## Delete List

### Command Mnemonic

**:MEMory:DELeTe:LIST**

### Command Description

This command deletes all list files.

## Delete State

### Command Mnemonic

**:MEMory:DELeTe:STATE**

### Command Description

This command deletes all state files.

## Delete Filename

### Command Mnemonic

```
:MEMory:DELEte[:NAME]<filename>  
:MMEMory:DELEte[:NAME] <msus>
```

### Command Description

This command clears the user file system of <filename> or <msus>. The variable <msus> represents [`<file_system>:`]`<filename>`.

## Free Memory Query

### Command Mnemonic

```
:MEMory:FREE[:ALL]?
```

### Command Description

This command outputs the number of bytes left in the user file system.

## Load Sweep List

### Command Mnemonic

```
:MEMory:LOAD:LIST <filename>  
:MMEMory:LOAD:LIST <msus>
```

### Command Description

This command loads a sweep list from memory. The variable <msus> represents [`<file_system>:`]`<filename>`.

## State Comment

### Command Mnemonic

```
:MEMory:STAtE:COMMeNt <reg_num>,<seq_num>,<comment>
```

### Command Description

This command allows you to add a descriptive comment to the saved state <reg>, <seq>. Comments can be up to 55 characters long.

### Command Query

:MEMory:STAtE:COMMeNt? <reg\_num>,<seq\_num>

### Store Sweep List

#### Command Mnemonic

:MEMory:STORe:LIST <filename>

:MMEMory:STORe:LIST <msus>

#### Command Description

This command stores the current sweep list to file <filename> or <msus>. The variable <msus> represents [**<file\_system>**:]<filename>.

---

## :OUTPut Subsystem

The RF output subsystem is used to set the controls and the parameters associated with the signal generator's RF output.

### RF Output Modulation State

#### Command Mnemonic

`:OUTPut:MODulation[:STATe] ON|OFF|1|0`

#### Command Description

This command sets the operating state of the signal generator's RF output modulations you have enabled. The choices are On (1) or Off (0). An annunciator is always displayed to indicate whether overall modulation is switched on or off.

#### Command Query

`:OUTPut:MODulation[:STATe]?`

#### Initial Values

At normal preset, this value is set to On.

At \*RST, this value is set to On.

At preset power-up, this value is set to On.

### RF Output Circuit Protection Clear

#### Command Mnemonic

`:OUTPut:PROTection:CLEar`

#### Command Description

This command resets the signal generator's reverse power protection circuitry.

#### Command Query

There is no query for this command.

## RF Output Circuit Protection Mode

### Command Mnemonic

:OUTPut:PROTection:MODE NORMAL|HP8648

### Command Description

This command sets operating mode of the signal generator's reverse power protection circuitry. The choices are Normal and HP8648. When set to Normal, the RPP can only be reset by eliminating the source of the excess reverse power detected at the RF OUTPUT connector, and then pressing **Reset RPP** or sending the remote command **OUTPut:PROTection:CLEar**. When set to HP8648, the RPP will be reset and the caution message removed on any SCPI command. This mode is provided for compatibility with systems that use the HP 8648. It is only available when you have selected HP8648A/B/C/D as the remote language. At all other times reverse power protection is set to Normal.

Operate in this mode with *caution*. Repeatedly tripping the RPP can cause damage to the signal generator. It is still best to eliminate the source of the excess reverse power and then reset the reverse power protection circuit by pressing the **Reset RPP** softkey or sending the remote command **OUTPut:PROTection:CLEar**. You should not ignore RPP trips. Hardware damage may occur before the RPP can protect the instrument.

### Command Query

:OUTPut:PROTection:MODE?

### Initial Values

The RPP reset mode is a persistent state, set to Normal at the factory; it is not affected by an instrument preset or by a power cycle.

## RF Output Circuit Protection Query

### Command Mnemonic

:OUTPut:PROTection:TRIPped?

### Command Description

This command queries the signal generator's reverse power protection circuitry to determine if it has been activated.

## RF Output State

### Command Mnemonic

`:OUTPut[:STATe] ON|OFF|1|0`

### Command Description

This command sets operating state of the signal generator's RF output. The choices are On (1) or Off (0). Although you can configure and engage various modulations, no signal is available at the RF OUTPUT connector until the command `:OUTPut[:STATe] ON` is executed. An annunciator is always displayed to indicate whether RF is switched on or off.

### Command Query

`:OUTPut[:STATe]?`

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.



---

## :PM Subsystem

The phase modulation subsystem is used to set the modulation controls and the parameters associated with phase modulated signals.

### Configure $\Phi$ Modulation Bandwidth

#### Command Mnemonic

**:PM[1] | 2:BA~~N~~Dwidth | BWIDth NORMa1 | HIGH**

#### Command Description

This command toggles between normal phase modulation mode and wideband phase modulation mode. The choices are Normal and Wideband.

#### Command Query

**:PM[1] | 2:BA~~N~~Dwidth | BWIDth?**

#### Initial Values

At normal preset, this value is set to Normal.

At \*RST, this value is set to Normal.

At preset power-up, this value is set to Normal.

### External $\Phi$ Modulation Source Coupling

#### Command Mnemonic

**:PM[1] | 2:EX~~T~~ernal[1] | 2:COU~~P~~ling AC | DC**

#### Command Description

This command sets the external coupling for the phase modulation source if **:PM[1] | 2:SO~~U~~Rce** was selected external. The choices are AC or DC coupling. This command does not change the currently active source, nor does it switch the current modulation on or off. The modulating signal may be the sum of several signals, either internal or external sources. If the coupling is set to DC, then both the AC and DC signal components pass. AC coupling passes only AC signal components.

### Command Query

`:PM[1] | 2:EXtErnal[1] | 2:COUPling?`

### Initial Values

At normal preset, this value is set to DC.

At \*RST, this value is set to DC.

At preset power-up, this value is set to DC.

## Internal $\Phi$ Modulation Source Rate

### Command Mnemonic

`:PM[1] | 2:INTerナル[1]:FREQuency <val><unit>`

### Command Description

This command sets the rate of the internally-generated modulation source. The choices for the variables `<val>` and `<unit>` range from 0.1 Hz (minimum) to 50 kHz (maximum) if the internal waveform is Sine wave. For all other waveforms, the maximum internal amplitude modulation rate is 10 kHz.

### Command Query

`:PM[1] | 2:INTerナル:FREQuency?`

### Initial Values

At normal preset, this value is set to 400.0 Hz.

At \*RST, this value is set to 400.0 Hz.

At preset power-up, this value is set to 400.0 Hz.

## Internal $\Phi$ Modulation Alternate Frequency

### Command Mnemonic

`:PM[1] | 2:INTerナル[FREQuency:ALternate <val><unit>`

### Command Description

This command sets the frequency for the alternate signal. The alternate frequency is the second frequency of a dual-sine or the stop frequency of a swept-sine. The choices for the variables `<val>` and `<unit>` range from 0.1 kHz (minimum) to 50.0 kHz (maximum).

### Command Query

`:PM[1] | 2:INTERNAL:FREQUENCY:ALTERNATE?`

### Initial Values

At normal preset, this value is set to 400.00 Hz.

At \*RST, this value is set to 400.00 Hz.

At preset power-up, this value is set to 400.00 Hz.

## Internal $\Phi$ Modulation Alternate Frequency Amplitude

### Command Mnemonic

`:PM[1] | 2:INTERNAL[1]:FREQUENCY:ALTERNATE:AMPLITUDE:PERCENT <val><unit>`

### Command Description

This command sets the amplitude of the alternate frequency as a percentage of the total amplitude (phase modulation deviation). Therefore, if the alternate frequency makes up 30% of the total amplitude, then the primary frequency is 70% of the total phase modulation deviation amplitude (for dual-sine only). The choices for the variables `<val>` and `<unit>` range from 0.1% (minimum) to 99.9% (maximum).

### Command Query

`:PM[1] | 2:INTERNAL:FREQUENCY:ALTERNATE:AMPLITUDE:PERCENT?`

### Initial Values

At normal preset, this value is set to 50%.

At \*RST, this value is set to 50%.

At preset power-up, this value is set to 50%.

## Internal $\Phi$ Modulation Waveform

### Command Mnemonic

`:PM[1] | 2:INTERNAL[1]:FUNCTION:SHAPE  
SINE | TRIangle | SQUARE | RAMP | NOISE | DUALsine | SWEPTSine`

### Command Description

This command selects the modulation waveform of the internally-generated signal. The choices are Sine, Triangle, Square, Ramp, Noise, Dual-sine, or Swept-sine.

### Command Query

```
:PM[1]|2:INTERNAL:FUNCTION:SHAPE?
```

### Initial Values

At normal preset, this value is set to Sine.

At \*RST, this value is set to Sine.

At preset power-up, this value is set to Sine.

## Internal $\Phi$ Modulation Sweep Time

### Command Mnemonic

```
:PM[1]|2:INTERNAL[1]:SWEep:TIME <val><unit>
```

### Command Description

This command selects the sweep time for a swept-sine, internally-generated signal.

### Command Query

```
:PM[1]|2:INTERNAL:SWEep:TIME?
```

### Initial Values

At normal preset, this value is set to 0.1 Hz.

At \*RST, this value is set to 0.1 Hz.

At preset power-up, this value is set to 0.1 Hz.

## Internal $\Phi$ Modulation Sweep Trigger

### Command Mnemonic

```
:PM[1]|2:INTERNAL[1]:SWEep:TRIGger IMMEDIATE|BUS|EXTERNAL|KEY
```

### Command Description

This command selects the trigger for the phase modulation sweep.

### Command Query

:PM[1] | 2:INTernal:SWEep:TRIGger?

### Initial Values

At normal preset, this value is set to Immediate.

At \*RST, this value is set to Immediate.

At preset power-up, this value is set to Immediate.

## Φ Modulation Source

### Command Mnemonic

:PM[1] | 2:SOURce INT[1]:EXT[1]|EXT2

### Command Description

This command sets the source that will generate the phase modulation. The choices are Internal Source 1, External Source 1, or External Source 2.

### Command Query

:PM[1] | 2:SOURce?

### Initial Values

At normal preset, this value is set to Internal.

At \*RST, this value is set to Internal.

At preset power-up, this value is set to Internal.

## Φ Modulation State

### Command Mnemonic

:PM[1] | 2:STATe ON|OFF|1|0

### Command Description

This command sets the operating state of the phase modulation source. The choices are On (1) or Off (0).

### Command Query

:PM[1] | 2:STATe?

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

### $\Phi$ Modulation Deviation

#### Command Mnemonic

:PM[1]|2[:DEVIation] <val><unit>

#### Command Description

This command sets the deviation of the phase modulation. The choices for are the variables <val> and <unit> depend on the carrier frequency. The maximum peak deviation for a frequency is calculated by multiplying N times 10 MHz. The following table lists the values for N and the resulting maximum peak deviations.

**Maximum Deviation Values for Phase Modulation**

Mode	Maximum Deviation	Maximum Rates (3 dB BW)	
		$\Phi$ M Path 1	$\Phi$ M Path 2
Normal	$N^a \times 90$ radians	100 kHz	100 kHz
High Bandwidth	$N \times 2\pi$ radians	1.5 MHz (typical)	1 MHz (typical)
	$N \times \pi/2$ radians	6 MHz (typical)	1 MHz (typical)

a. For the value of N, refer to the table below.

**Carrier Frequency Bands versus Value of N**

Carrier Frequency	N
250 kHz to $\leq$ 249.999 MHz	1
> 249.999 MHz to $\leq$ 500 MHz	0.5
> 500 MHz to $\leq$ 1 GHz	1
> 1 GHz to $\leq$ 2 GHz	2
> 2 GHz to 4 GHz	4

### Command Query

:PM[1]|2[:DEVIation]?

### Initial Values

At normal preset, this value is set to 0.000 radians.

At \*RST, this value is set to 0.000 radians.

At preset power-up, this value is set to 0.000 radians.

## Φ Modulation Deviation Coupling

### Command Mnemonic

:PM[1]|2[:DEVIation]:TRACk ON|OFF|1|0

### Command Description

This command allows the phase modulation deviation values on both path 1 and path 2 to track each other. The choices are On (1) or Off (0). If the phase modulation deviation coupling function is activated, changing the phase modulation deviation on path 1 will change the in phase modulation deviation on path 2 to match, and visa versa. This is an exact match tracking, not an offset tracking.

### Command Query

:PM[1]|2[:DEVIation]:TRACk?

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

---

## :POWER Subsystem

The RF power subsystem is used to set the controls and the parameters associated with the signal generator's RF output amplitude.

### RF Output Automatic Leveling Circuitry (ALC) Bandwidth

#### Command Mnemonic

:POWER:ALC:BANDwidth|BWIDth NORMAL|NARROW

#### Command Description

This command changes the bandwidth of the ALC loop. Defaults are Normal if IQ modulation is off, Narrow if EXT IQ is on.

#### Command Query

:POWER:ALC:BANDwidth|BWIDth?

#### Initial Values

At normal preset, this value is set to Normal.

At \*RST, this value is set to Normal.

At preset power-up, this value is set to Normal.

### RF Output Automatic Leveling Circuitry (ALC) Search State

#### Command Mnemonic

:POWER:ALC:SEARCH ON|OFF|1|0|ONCE

#### Command Description

This command toggles between the auto and manual modes of power search mode. The choices are On (1) or Off (0). Power search is an internal calibration routine used to achieve calibrated output power when the ALC is off. When you set power search to AUTO, power search will execute automatically with each change to the AM or pulse modulation state.

#### Command Query

:POWER:ALC:SEARCH?



### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

## RF Output Automatic Leveling Circuitry (ALC) State

### Command Mnemonic

`:POWER:ALC:STATE ON|OFF|1|0`

### Command Description

This command sets the operating state of the signal generator's RF output automatic leveling circuitry. The choices are On (1) or Off (0).

### Command Query

`:POWER:ALC:STATE?`

### Initial Values

At normal preset, this value is set to On.

At \*RST, this value is set to On.

At preset power-up, this value is set to On.

## Automatic RF Output Level Attenuation

### Command Mnemonic

`:POWER:ATTenuation:AUTO ON|OFF|1|0`

### Command Description

This command sets the signal generator to attenuator-hold mode where power drop-outs do not occur for power adjustments. The choices are On (1) or Off (0). The maximum power adjustment range is frequency-dependent, but will be at least +7 dBm for increasing power and at least -13 dBm for decreasing power.

### Command Query

`:POWER:ATTenuation:AUTO?`

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

## RF Output Power Mode

### Command Mnemonic

`:POWer:MODE FIXed|LIST`

### Command Description

This command sets the signal generator's RF output power operating mode. The choices are Fixed and List.

### Command Query

`:POWer:MODE?`

### Initial Values

At normal preset, this value is set to Fixed.

At \*RST, this value is set to Fixed.

At preset power-up, this value is set to Fixed.

## RF Output Reference Power

### Command Mnemonic

`:POWer:REFErence <val><unit>`

### Command Description

This command sets the power setting of the signal generator's RF output reference. The choices for the variables <val> and <unit> range from -400 dBm (minimum) to 300 dBm (maximum).

### Command Query

`:POWer:REFErence?`

### Initial Values

At normal preset, this value is set to 0 dBm.

At \*RST, this value is set to 0 dBm.

At preset power-up, this value is set to 0 dBm.

## RF Output Reference Power State

### Command Mnemonic

**:POWER:REFERENCE:STATE ON|OFF|1|0**

### Command Description

This command sets the operating state of the signal generator's RF output reference. The choices are On (1) or Off (0).

### Command Query

**:POWER:REFERENCE:STATE?**

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

## RF Output Start Power

### Command Mnemonic

**:POWER:START <val><unit>**

### Command Description

This command sets the first operating power level in swept power applications. The choices for the variables <val> and <unit> are -135.00 dBm to +20 dBm.

### Command Query

**:POWER:START?**

### Initial Values

At normal preset, this value is set to -135.00 dBm.

At \*RST, this value is set to -135.00 dBm.

At preset power-up, this value is set to -135.00 dBm.

## RF Output Stop Power

### Command Mnemonic

```
:POWer:STOP <val><unit>
```

### Command Description

This command sets the last operating power level in swept power applications. The choices for the variables <val> and <unit> are -135.00 dBm to +20 dBm.

### Command Query

```
:POWer:STOP?
```

### Initial Values

At normal preset, this value is set to -135.00 dBm.

At \*RST, this value is set to -135.00 dBm.

At preset power-up, this value is set to -135.00 dBm.

## RF Output Level Amplitude Offset

### Command Mnemonic

```
:POWer[:LEVel][:IMMediate]:OFFSet <val><unit>
```

### Command Description

This command sets a value for amplitude offset for the RF output power level. An amplitude offset changes the value shown in the amplitude area of the display but does not affect the absolute output power. For example, if the current output power is 0 dBm and you enter an amplitude offset of -3 dBm, the output power will remain at 0 dBm but the display will show an amplitude of -3 dBm. This feature lets you simulate the power level at a test point beyond the RF OUTPUT connector.

### Command Query

```
:POWER[:LEVEL][:IMMEDIATE]:OFFSET?
```

### Initial Values

At normal preset, this value is set to 0.00 dB.

At \*RST, this value is set to 0.00 dB.

At preset power-up, this value is set to 0.00 dB.

## RF Output Level Immediate Amplitude

### Command Mnemonic

```
:POWER[:LEVEL][:IMMEDIATE][:AMPLITUDE] <val><unit>
```

### Command Description

This command sets the RF output power. The choices for variables **<val>** and **<unit>** are power levels between -135.00 (minimum) and +20 dBm (maximum). Amplitude becomes the active function and the current value is shown in the active entry area of the display.

### Command Query

```
:POWER[:LEVEL][:IMMEDIATE][:AMPLITUDE]?
```

### Initial Values

At normal preset, this value is set to -135 dBm.

At \*RST, this value is set to -135 dBm.

At preset power-up, this value is set to -135 dBm.

---

## :PULM Subsystem

The pulse modulation subsystem is used to set the modulation controls and the parameters associated with pulse modulated signals.

### Configure Internal Pulse Waveform

#### Command Mnemonic

`:PULM:INTernal[1]:FUNction:SHAPE <enum>`

#### Command Description

This command toggles the modulation waveform of the internal pulse modulation source between squarewave and variable pulse width wave. The choices for the variables `<enum>` are Square and Pulse.

#### Command Query

`:PULM:INTernal[1]:FUNction:SHAPE?`

#### Initial Values

At normal preset, this value is set to Pulse.

At \*RST, this value is set to Pulse.

At preset power-up, this value is set to Pulse.

### Pulse Modulation Source

#### Command Mnemonic

`:PULM:SOURce INT|EXT2`

#### Command Description

This command sets the source that will generate the pulse modulation. The choices are Internal Square, Internal Pulse, or External Source 2.

#### Command Query

`:PULM:SOURce?`

### Initial Values

At normal preset, this value is set to Internal Square.

At \*RST, this value is set to Internal Square.

At preset power-up, this value is set to Internal Square.

### Pulse Modulation State

#### Command Mnemonic

:PULM:STATe ON|OFF|1|0

#### Command Description

This command sets the operating state of the pulse modulation source. The choices are On (1) or Off (0).

#### Command Query

:PULM:STATe?

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

### Internal Pulse Modulation Source Rate

#### Command Mnemonic

:PULM:INTernal[1]:FREQuency <val><unit>

#### Command Description

This command sets the rate of the internal squarewave pulse modulation source. The choices for the variables <val> and <unit> range from 0.1 Hz (minimum) to 50 kHz (maximum) if the internal waveform is Sine wave. For all other waveforms, the maximum internal amplitude modulation rate is 10 kHz.

#### Command Query

:PULM:INTernal[1]:FREQuency?

### Initial Values

At normal preset, this value is set to 400.0 Hz.

At \*RST, this value is set to 400.0 Hz.

At preset power-up, this value is set to 400.0 Hz.

## Configure Internal Pulse Modulation Pulse Period

### Command Mnemonic

```
:PULM:INTernal[1]:PERiod <val><unit>
```

### Command Description

This command sets the period time for the internally-generated pulse modulation source. The choices for the variables `<val>` and `<unit>` range from 30 seconds to 16 microseconds.

### Command Query

```
:PULM:INTernal[1]:PERiod?
```

### Initial Values

At normal preset, this value is set to 80.0 usec.

At \*RST, this value is set to 80.0 usec.

At preset power-up, this value is set to 80.0 usec.

## Configure Internal Pulse Modulation Pulse Width

### Command Mnemonic

```
:PULM:INTernal[1]:PWIDth <val><unit>
```

### Command Description

This command sets the pulse width for the internally-generated pulse modulation source. The choices for the variables `<val>` and `<unit>` range from 8.0 to 80.0 microseconds.

### Command Query

```
:PULM:INTernal[1]:PWIDth?
```



**Initial Values**

At normal preset, this value is set to 40.0 usec.

At \*RST, this value is set to 40.0 usec.

At preset power-up, this value is set to 40.0 usec.

---

## :STATus Subsystem

The IEEE status subsystem is used to set the controls and the parameters associated with status conditions within the signal generator.

### Standard Operation Status Group Condition Register Query

#### Command Mnemonic

```
:STATus:OPERation:CONDition?
```

#### Command Description

This command returns the decimal value of the sum of the bits in the Standard Operation Status Group Event Register. For example, if a sweep is in progress (bit 3), then the value 8 is returned.

### Standard Operation Status Group Enable

#### Command Mnemonic

```
:STATus:OPERation:ENable <num>
```

#### Command Description

This command determines what bits in the Standard Operation Status Group Event Register will set the Standard Operation Status Group Summary bit (bit 7) in the Status Byte Register. The variable <num> is the sum of the decimal values of the bits you want to enable.

### Standard Operation Status Group Negative Transition Filter Register Enable

#### Command Mnemonic

```
:STATus:OPERation:NTRansition <num>
```

#### Command Description

This command determines what bits in the Standard Operation Status Group Event Register will set the corresponding bit in the Standard Operation Status Group Event Register when that bit has a negative transition (1 to 0). The variable <num> is the sum of the decimal values of the bits that you want to enable.

## Standard Operation Status Group Positive Transition Filter Register Enable

### Command Mnemonic

:STATUS:OPERation:PTRansition <num>

### Command Description

This command determines what bits in the Standard Operation Status Group Event Register will set the corresponding bit in the Standard Operation Status Group Event Register when that bit has a positive transition (0 to 1). The variable <num> is the sum of the decimal values of the bits that you want to enable.

## Standard Operation Status Group Event Register Query

### Command Mnemonic

:STATUS:OPERation[:EVENT]?

### Command Description

This command returns the decimal value of the sum of the bits in the Standard Operation Event Register. For example, if a sweep is in progress (bit 3), then the value 8 is returned. Note that the register requires that the equivalent PTR or NTR filters be set before a condition register bit can set a bit in the event register. Also note that the data in the register is latched until it is queried. Once queried, the data is cleared.

## Status Preset

### Command Mnemonic

:STATUS:PRESet

### Command Description

This command presets all transition filters, enable registers, and all error/event queue enable registers.

## Data Questionable Status Negative Transition Filter Register Enable

### Command Mnemonic

:STATUS:QUESTionable:NTRansition <num>

### Command Description

This command determines what bits in the Data Questionable Status Group Condition Register will set the corresponding bit in the Data Questionable Status Group Event Register when that bit has a negative transition (1 to 0). The variable <num> is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Condition Positive Transition Filter Register Enable

### Command Mnemonic

```
:STATus:QUESTIONable:PTRansition <num>
```

### Command Description

This command determines what bits in the Data Questionable Status Group Condition Register will set the corresponding bit in the Data Questionable Status Group Event Register when that bit has a positive transition (0 to 1). The variable <num> is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Status Group Event Register Query

### Command Mnemonic

```
:STATus:QUESTIONable[:EVENT]?
```

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Event Register. For example, if the instrument has just been connected to line power and the Reference Oscillator Oven (Option 1E5 only) is cold (bit 4), then a value of 16 is returned. Note that the register requires that the equivalent PTR or NTR filters be set before a condition register bit can set a bit in the Event register. Note also that the data in this register is latched until it is queried. Once queried, the data is cleared.

## Data Questionable Calibration Status Group Condition Register Query

### Command Mnemonic

```
:STATus:QUESTIONable:CALibration:CONDition?
```

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Calibration Condition Register. For example, if the DCFM or DCΦM zero calibration fails (bit 0), then a value of 1 is returned. Note that the data in this register is continuously updated and reflects the current conditions.

## Data Questionable Calibration Status Group Enable

### Command Mnemonic

`:STATus:QUESTionable:CALibration:ENable <num>`

### Command Description

This command determines what bits in the Data Questionable Calibration Status Group Event Register will set the Data Questionable Calibration Summary bit (bit 8) in the Data Questionable Status Group Condition Register. The variable `<num>` is the sum of the decimal values of the bits you want to enable.

## Data Questionable Calibration Status Negative Transition Filter Register Enable

### Command Mnemonic

`:STATus:QUESTionable:CALibration:NTRansition <num>`

### Command Description

This command determines what bits in the Data Questionable Calibration Status Group Condition Register will set the corresponding bit in the Data Questionable Calibration Status Group Event Register when that bit has a negative transition (1 to 0). The variable `<num>` is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Calibration Status Positive Transition Filter Register Enable

### Command Mnemonic

`:STATus:QUESTionable:CALibration:PTRansition <num>`

### Command Description

This command determines what bits in the Data Questionable Calibration Status Group Condition Register will set the corresponding bit in the Data Questionable Calibration Status Group Event Register when that bit has a positive transition (0 to 1). The variable `<num>` is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Calibration Status Group Event Register Query

### Command Mnemonic

```
:STATus:QUESTIONable:CALibration[:EVENT]?
```

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Calibration Event Register. For example, if the DCFM or DCΦM zero calibration has failed (bit 0), then a 1 is returned. Note that the register requires that the equivalent PTR or NTR filters be set before a condition register bit can set a bit in the Event register. Note also that the data in this register is latched until it is queried. Once queried, the data is cleared.

## Data Questionable Condition Query

### Command Mnemonic

```
:STATus:QUESTIONable:CONDition?
```

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Condition Register. For example, if a Reference Oscillator Oven (Option 1E5 only) is cold (bit 4), then a value of 16 is returned. Note that the data in this register is continuously updated and reflects current conditions.

## Data Questionable Status Group Enable

### Command Mnemonic

```
:STATus:QUESTIONable:ENABLE <num>
```

### Command Description

This command determines what bits in the Data Questionable Status Group Event Register will set the Data Questionable Status Group Summary bit (bit 3) in the Status Byte Register. The variable <num> is the sum of the decimal values of the bits you want to enable.

## Data Questionable Frequency Status Group Condition Register Query

### Command Mnemonic

```
:STATus:QUESTIONable:FREQuency:CONDition?
```

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Frequency Condition Register. For example, if the 1 GHz internal reference clock is unlocked (bit 2), then a value of 4 is returned. Note that the data in this register is continuously updated and reflects the current conditions.

## Data Questionable Frequency Status Group Enable

### Command Mnemonic

**:STATus:QUESTionable:FREQuency:ENable <num>**

### Command Description

This command determines what bits in the Data Questionable Frequency Status Group Event Register will set the Data Questionable Frequency Summary bit (bit 5) in the Data Questionable Status Group Condition Register. The variable <num> is the sum of the decimal values of the bits you want to enable.

## Data Questionable Frequency Status Negative Transition Filter Register Enable

### Command Mnemonic

**:STATus:QUESTionable:FREQuency:NTRansition <num>**

### Command Description

This command determines what bits in the Data Questionable Frequency Status Group Condition Register will set the corresponding bit in the Data Questionable Frequency Status Group Event Register when that bit has a negative transition (1 to 0). The variable <num> is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Frequency Status Positive Transition Filter Register Enable

### Command Mnemonic

**:STATus:QUESTionable:FREQuency:PTRansition <num>**

### Command Description

This command determines what bits in the Data Questionable Frequency Status Group Condition Register will set the corresponding bit in the Data Questionable Frequency Status Group Event Register when that bit has a positive transition (0 to 1). The variable <num> is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Frequency Status Group Event Register Query

### Command Mnemonic

```
:STATus:QUESTionable:FREQuency[:EVENT]?
```

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Frequency Event Register. For example, if the 1 GHz internal reference clock is unlocked (bit 2), then a value of 4 is returned. Note that the register requires that the equivalent PTR or NTR filters be set before a condition register bit can set a bit in the Event register. Note also that the data in this register is latched until it is queried. Once queried, the data is cleared.

## Data Questionable Modulation Status Group Condition Register Query

### Command Mnemonic

```
:STATus:QUESTionable:MODulation:CONDition?
```

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Modulation Condition Register. Note that the data in this register is continuously updated and reflects the current conditions.

## Data Questionable Modulation Status Group Enable

### Command Mnemonic

```
:STATus:QUESTionable:MODulation:ENable <num>
```

### Command Description

This command determines what bits in the Data Questionable Modulation Status Group Event Register will set the Data Questionable Modulation Summary bit (bit 7) in the Data Questionable Status Group Condition Register. The variable <num> is the sum of the decimal values of the bits you want to enable.

## Data Questionable Modulation Status Negative Transition Filter Register Enable

### Command Mnemonic

```
:STATus:QUESTionable:MODulation:NTRansition <num>
```



### Command Description

This command determines what bits in the Data Questionable Modulation Status Group Condition Register will set the corresponding bit in the Data Questionable Modulation Status Group Event Register when that bit has a negative transition (1 to 0). The variable `<num>` is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Modulation Status Positive Transition Filter Register Enable

### Command Mnemonic

```
:STATus:QUESTionable:MODulation:PTRansition <num>
```

### Command Description

This command determines what bits in the Data Questionable Modulation Status Group Condition Register will set the corresponding bit in the Data Questionable Modulation Status Group Event Register when that bit has a positive transition (0 to 1). The variable `<num>` is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Modulation Status Group Event Register Query

### Command Mnemonic

```
:STATus:QUESTionable:MODulation[:EVENT]?
```

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Modulation Event Register. For example, if the External 1 AC modulation is selected with no modulation connected, a Modulation 1 Undermod condition exists (bit 0), then a value of 1 is returned. Note that the register requires that the equivalent PTR or NTR filters be set before a condition register bit can set a bit in the Event register. Note also that the data in this register is latched until it is queried. Once queried, the data is cleared.

## Data Questionable Power Status Group Condition Register Query

### Command Mnemonic

```
:STATus:QUESTionable:POWer:CONDition?
```

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Power Condition Register. For example, if the RF output signal is unlevelled (bit 1), then a value of 2 is returned. Note that the data in this register is continuously updated and reflects the current conditions.

## Data Questionable Power Status Group Enable

### Command Mnemonic

```
:STATus:QUESTionable:POWer:ENable <num>
```

### Command Description

This command determines what bits in the Data Questionable Power Status Group Event Register will set the Data Questionable Power Summary bit (bit 3) in the Data Questionable Status Group Condition Register. The variable <num> is the sum of the decimal values of the bits you want to enable.

## Data Questionable Power Status Negative Transition Filter Register Enable

### Command Mnemonic

```
:STATus:QUESTionable:POWer:NTRansition <num>
```

### Command Description

This command determines what bits in the Data Questionable Power Status Group Condition Register will set the corresponding bit in the Data Questionable Power Status Group Event Register when that bit has a negative transition (1 to 0). The variable <num> is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Power Status Positive Transition Filter Register Enable

### Command Mnemonic

```
:STATus:QUESTionable:POWer:PTRansition <num>
```

### Command Description

This command determines what bits in the Data Questionable Power Status Group Condition Register will set the corresponding bit in the Data Questionable Power Status Group Event Register when that bit has a positive transition (0 to 1). The variable <num> is the sum of the decimal values of the bits that you want to enable.

## Data Questionable Power Status Group Event Register Query

### Command Mnemonic

`:STATus:QUESTionable:POWer[:EVENT]?`

### Command Description

This command returns the decimal value of the sum of the bits in the Data Questionable Power Event Register. For example, if the RF output signal is unlevelled (bit 1), then a value of 2 is returned. Note that the register requires that the equivalent PTR or NTR filters be set before a condition register bit can set a bit in the Event register. Note also that the data in this register is latched until it is queried. Once queried, the data is cleared.

---

## :SWEep Subsystem

The sweep subsystem is used to set the controls and the parameters associated with a frequency and/or power sweep.

### Sweep Dwell

#### Command Mnemonic

`:SWEep:DWELl <val>`

#### Command Description

This command sets the dwell time for each point in a sweep. The choices for the variable `<val>` are 0.001 seconds to 60 seconds in 1 ms increments. Dwell time is used when the point trigger is **Immediate**. The dwell time is the minimum amount of time the sweep is guaranteed to pause after setting the frequency and power for the current point.

#### Command Query

`:SWEep:DWELl?`

#### Initial Values

At normal preset, this value is set to 0.002 seconds.

At \*RST, this value is set to 0.002 seconds.

At preset power-up, this value is set to 0.002 seconds.

### Sweep Points

#### Command Mnemonic

`:SWEep:POINTs <val>`

#### Command Description

This command defines the number of points in a step sweep. The choices for the variable `<val>` range from 2 to 401. A step sweep must always have a minimum of 2 points and can be configured to have as many as 401 points.

### Command Query

:SWEep:POINTs?

### Initial Values

At normal preset, this value is set to 2.

At \*RST, this value is set to 2.

At preset power-up, this value is set to 2.

---

## :SYSTEM Subsystem

The system subsystem is used to set the controls and the parameters associated with overall system communication.

### Error Information Query

#### Command Mnemonic

:SYSTEM:ERROR[:NEXT]?

#### Command Description

This command queries the signal generator's error queue.

### Help Mode

#### Command Mnemonic

:SYSTEM:HELP:MODE SINGLE|CONTINUOUS

#### Command Description

This command sets the mode of the signal generator's help function. The choices are Single and Continuous. In Single mode, help is provided only for the next key you that you press. In Continuous mode, help is provided for the next key you press and that key's function is also executed (except for the **Preset** key). In either mode, pressing the **Help** key while the help dialog box is displayed will turn help off.

#### Command Query

:SYSTEM:HELP:MODE?

#### Initial Values

At normal preset, this value is set to Single.

At \*RST, this value is set to Single.

At preset power-up, this value is set to Single.

## Remote Language

### Command Mnemonic

**:SYSTem:LANGUage "SCPI" | "COMP" | "NADC" | "PDC" | "PHS" | "HP8648"**

### Command Description

This command sets the remote language for the signal generator. The choices are SCPI, COMP, NADC, PDC, PHS, or HP8648 compatible.

COMP is for HP 8656/8657A/B compatibility.

NADC is for HP 8657D compatibility.

PDC is for HP 8657D compatibility.

PHS is for HP 8657J compatibility.

HP8648 is for HP 8648A/B/C/D compatibility.

### Command Query

**:SYSTem:LANGUage?**

### Initial Values

At normal preset, this value is set based on the value of the **:SYSTem:PRESet:LANGUage** setting.

At \*RST, this value is set based on the value of the **:SYSTem:PRESet:LANGUage** setting.

At preset power-up, this value is set based on the value of the **:SYSTem:PRESet:LANGUage** setting.

## Power On/Preset Conditions

### Command Mnemonic

**:SYSTem:PON:TYPE PRESet | LAST**

### Command Description

This command sets the defined instrument conditions after a power on. The choices are Preset (the factory preset conditions) or Last (the conditions at the time the instrument was powered down).

### Command Query

**:SYSTem:PON:TYPE?**

### Initial Values

This is a persistent state. There are no initial values. The instrument is initially shipped from the factory with this parameter set to Last.

## System Preset

### Command Mnemonic

:SYSTEM:PRESet

### Command Description

This command returns the signal generator to a set of defined conditions.

## Preset Language

### Command Mnemonic

:SYSTEM:PRESet:LANGUage "SCPI" | "COMP" | "HP8648"

### Command Description

This command sets the remote language at the preset condition. The choices are SCPI, COMP, and HP 8648 compatible.

COMP is for HP 8656/8657A/B compatibility.

HP8648 is for HP 8648A/B/C/D compatibility.

### Command Query

:SYSTEM:PRESet:LANGUage?

### Initial Values

At normal preset, this value is set to SCPI.

At \*RST, this value is set to SCPI.

At preset power-up, this value is set to SCPI.

## Preset Type

### Command Mnemonic

:SYSTEM:PRESet:TYPE NORMAl | USER



### Command Description

This command toggles the preset state between factory-defined conditions and user-defined conditions. The choices are Normal and User.

### Command Query

:SYSTEM:PRESet:TYPE?

### Initial Values

This is a persistent state set to Normal at the factory.

## Screen Saver Delay

### Command Mnemonic

:SYSTEM:SSAVer:DElay <val>

### Command Description

This command sets the amount of time (in hours) before the display light or text (see next command) is switched off, if there is no input via the front panel. The choices for the variable <val> are the numbers 1 through 12, in 1 hour increments.

### Command Query

:SYSTEM:SSAVer:DElay?

### Initial Values

At normal preset, this value is set to 1.

At \*RST, this value is set to 1.

At preset power-up, this value is set to 1.

## Screen Saver Mode

### Command Mnemonic

:SYSTEM:SSAVer:MODE LIGHT | TEXT

### Command Description

This command toggles the screen saver mode the choices are Light Only and Light & Text.

### Command Query

**:SYSTEM:SSAVER:MODE?**

### Initial Values

At normal preset, this value is set to Light Only.

At \*RST, this value is set to Light Only.

At preset power-up, this value is set to Light Only.

## Screen Saver State

### Command Mnemonic

**:SYSTEM:SSAVER:STATE ON|OFF|1|0**

### Command Description

This command sets the operating state of the screen saver. The choices are On (1) and Off (0).

### Command Query

**:SYSTEM:SSAVER:STATE?**

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

## SCPI Version

### Command Mnemonic

**:SYSTEM:VERSION?**

### Command Description

This command returns the SCPI version number with which the instrument complies.

---

## :TRIGger Subsystem

The trigger subsystem is used to set the controls and the parameters associated with triggering a sweep in the signal generator.

### Abort

#### Command Mnemonic

:ABORt

#### Command Description

This command causes the sweep in progress to abort and reset. If INIT:CONT is ON, then the sweep will immediately re-initiate. The pending operation flag (affecting \*OPC, \*WAI, and \*OPC?) will undergo a transition once the sweep has been reset.

#### Command Query

There is no query for this command.

### Continuous Sweep

#### Command Mnemonic

:INITiate:CONTinuous[:ALL] ON|OFF|1|0

#### Command Description

This command activates the continuous sweep mode. The choices are On (1) or Off (0).

Modifying this value does not affect the sweep in progress. Executing this command with **ON** causes the sweep to be automatically restarted at the end of the previous sweep. Executing this command with **OFF** causes the sweep to wait until an **:INITiate[:IMMEDIATE]** command is sent to reinitiate the sweep.

### Command Query

`:INITiate:CONTinuous[:ALL]?`

### Initial Values

At normal preset, this value is set to Off.

At \*RST, this value is set to Off.

At preset power-up, this value is set to Off.

### Single Sweep

#### Command Mnemonic

`:INITiate[:IMMediate][:ALL]`

#### Command Description

This command initiates a single sweep if the sweep is on for either frequency or power, and the sweep is not already initiated.

### Trigger Output Polarity

#### Command Mnemonic

`:TRIGger:OUTPut:POLarity POSitive|NEGative`

#### Command Description

This command sets the polarity of the TTL signal present at the TRIGGER OUT connector. The choices are Positive and Negative. The Trigger out is asserted (after the frequency and/or power is set) while the sweep is waiting for its step trigger. Also, the swept-sine sends a pulse to the TRIGGER OUT at the beginning of each sweep.

#### Command Query

`:TRIGger:OUTPut:POLarity?`

#### Initial Values

At normal preset, this value is set to Positive.

At \*RST, this value is set to Positive.

At preset power-up, this value is set to Positive.

## External Trigger On Slope

### Command Mnemonic

**:TRIGger[:SEQuence]:SLOPe POSitive|NEGative**

### Command Description

This command sets the polarity of the slope present at the TRIGGER IN connector that will trigger a sweep in the signal generator. The choices are Positive edge and Negative edge.

### Command Query

**:TRIGger[:SEQuence]:SLOPe?**

### Initial Values

At normal preset, this value is set to Positive.

At \*RST, this value is set to Positive.

At preset power-up, this value is set to Positive.

## Trigger Source

### Command Mnemonic

**:TRIGger[:SEQuence]:SOURce BUS|IMMEDIATE|EXTernal|KEY**

### Command Description

This command sets the trigger source. The choices are Bus (allows you to trigger a list or step sweep via HP-IB), Immediate (immediately triggers the current sweep once it is armed), External (Pos and Neg) (allows you to trigger a list or step sweep on the positive or negative edge of a signal applied to the TRIGGER IN connector), or Key (immediately triggers an armed sweep when you press the **Trigger** hardkey).

### Command Query

**:TRIGger[:SEQuence]:SOURce?**

### Initial Values

At normal preset, this value is set to Immediate.

At \*RST, this value is set to Immediate.

At preset power-up, this value is set to Immediate.

## Immediate Trigger

### Command Mnemonic

`:TRIGger [ :SEQuence ] [ :IMMediate ]`

### Command Description

This command has no effect unless sweep is waiting for a trigger. If sweep is waiting for a trigger (sweep or point-to-point) then this command triggers the sweep, regardless of other settings in the :SWEep subsystem.

### Command Query

There is no query for this command.



---

## 6 Error Messages

**\_\_\_\_\_**  
This chapter explains the error messages that might be shown on the front panel display or transmitted over an interface bus.



---

## Error Messages

If an error condition occurs in the signal generator, it will always be reported to both the front panel display error queue and the SCPI (remote interface) error queue. These two queues are viewed and managed separately.

### The Front Panel Error Queue

This queue is designed in a circular (rotating) fashion. It can hold up to 30 error messages. If the queue is full, and additional error messages arrive, the oldest errors are lost. The previously read messages are not cleared from the queue; they remain in the queue until they are overwritten by a new error message.

The front panel error queue information can be accessed by pressing **Utility, Error Info**. From the Error Info menu, you may choose from **View Previous Error Message, View Next Error Message, or Clear Error Queue(s)**. You can also use the RPG and the arrow keys to review the messages/

If there are any unviewed messages in the front panel error queue, the **ERR** annunciator will be activated on the signal generator's display. you can optionally rotate the RPG or use the arrow keys to view the error messages. To empty the queue, press **Utility, Error Info, Clear Error Queue(s)**.

There are some special error types called permanent errors. These include **unlock, ovencold, hi/lo**, etc. Permanent errors remain in the error queues until the error condition is cleared. Pressing **Utility, Error Info, Clear Error Queue(s)** will empty the front panel error queue, but the permanent errors will be re-reported if the error condition(s) still exist.

### The SCPI Remote Interface Error Queue

This queue is constructed in a linear first-in/first-out fashion. It can hold up to 30 error messages. As errors and events are detected, they are placed in the queue. Unlike the front panel error queue, errors in this queue are not overwritten by the latest incoming error messages. If the queue overflows, the last error in the queue is replaced with the error:

**-350,Queue overflow**

When the queue overflows, the least recent errors remain in the queue, and the most recent error is discarded. Reading an error from the head of the queue removes that error from the queue, and opens a position in the tail of the queue for a new error, if one is subsequently detected.

When all the errors have been read from the queue, further error queries will return:

**0,No error**

The SCPI query **SYSTem:ERROr?** is used to view messages in the SCPI error queue. The error queue will be cleared when any of the following occur (IEEE 488.2, section 11.4.3.4):

- Upon power up
- Upon receipt of a **\*CLS** command
- Upon reading the last item from the queue

In the SCPI error queue, the permanent errors are re-reported after the message **0, No error** is read using the **SYSTem:ERROr?** query and after the **\*CLS** command is executed.

---

## Querying the Error Queue

The queue query message is a request for the next entry from the instrument's error queue. This queue contains an integer that can range from -32768 to 32767. Negative error numbers are reserved by the SCPI standard and are defined in this section. Positive error numbers are instrument-dependent. An error value of zero indicates that no error or event has occurred.

The signal generator responds to the **SYSTEM:ERROR?** (or **STATUS:QUEUE?**) query using the following form:

<error number>, <error description>

The <error number> is a unique error descriptor. Certain standard error numbers are described in this section. The <error description> is a short description of the error, (optionally) followed by additional information regarding the error. Short descriptions of the standard error numbers are described in this section. The information that follows the error message may contain corrective actions that should be followed to correct the error condition.

The <device-dependent info> section of the response may contain information which will allow you to determine the exact error and context. For example:

**-131, Invalid suffix;FREQUENCY:CENT 2.0E+5 dBmV**

The maximum string length of <error description> plus <device-dependent info> is 255 characters. The <error description> will be sent exactly as indicated in this document, including case.

If there has been no error, that is, if the queue is empty, the signal generator will respond with:

**0, No error**

If there has been more than one error, the instrument will respond with the first one in its queue. Subsequent responses to **SYSTEM:ERROR?** will continue with the queue until it is empty.

---

## Error Numbers

The system-defined error numbers are chosen on an enumerated (“1 of N”) basis. The SCPI-defined error numbers and the <error\_description> portions of the error query response are listed here. The first error described in each class (for example, -100, -200, -300, -400) is a “generic” error. In selecting the proper error number to report, more specific error codes are preferred.

---

## No Error

This message indicates that the error queue contains no errors.

**Error Number**    Error Description [description/explanation/examples]

**0**

**No error**

The queue is empty. Every error in the queue has been read or the queue was purposely cleared by power-on or \*CLS.

---

## SCPI Standard Error Messages

### Error Message Description

The list of error messages in this chapter describes all of the SCPI error messages associated with signal generator operation. A sample error message description is provided below to help you understand how information will be presented in this section.

**-222                    Data out of range;value clipped to lower limit.**

Indicates that the user has entered a deviation, depth or internal source frequency that is beyond the specified limits.

The following list explains each element of the sample error message listing shown above.

- SCPI Error Number - The standard SCPI error number (-222 in this example). Standard SCPI error numbers are always negative, with the exception of **0, No error**.
- SCPI Error Message - The SCPI error message is **Data out of range;** in this example.
- Detailed Description - The information that appears after the semicolon (;) provides more detail as to the exact nature of the error. In this example, **value clipped to lower limit** tells you that you have entered a value outside the allowable range and the signal generator has changed the value so that it falls within the allowable limits. If no detailed description exists, it will be omitted from the error message.
- Explanation/Action Required - The text that appears below each error message listing contains an explanation of the error message and, in some cases, corrective actions that should be followed in order to correct the error condition. Though this information is not shown on the signal generator's display, it can be found in the following section.

---

## Command Error

An error number in the range [-199 to -100] indicates that an IEEE 488.2 syntax error has been detected by the instrument's parser. The occurrence of any error in this class will cause the command error bit (bit 5) in the event status register (IEEE 488.2, section 11.5.1) to be set. If this bit is set, one of the following events has occurred:

- An IEEE 488.2 syntax error has been detected by the parser. That is, a control-to-device message was received which is in violation of the IEEE 488.2 standard. Possible violations include a data element which violates device listening formats or whose type is unacceptable to the device.
- An unrecognized header was received. Unrecognized headers include incorrect device-specific headers and incorrect or unimplemented IEEE 488.2 common commands.

Events that generate command errors will not generate execution errors, device-specific errors, or query errors; see the error definitions in this chapter.

### Command Error Message Descriptions

This section lists the signal generator's command error messages and their associated descriptions.

**Error Number**    Error Description [description/explanation/examples]

<b>-100</b>	<b>Command error</b>  This is a generic syntax error for devices that cannot detect more specific errors. The code indicates only that a command error as defined in IEE 488.2, 11.5.1.1.4 has occurred.
<b>-101</b>	<b>Invalid character</b>  A syntactic command contains a character which is invalid for that type. For example, a header containing an ampersand, SETUP&. This error might be used in place of error numbers -114, -121, -141 and some others.
<b>-102</b>	<b>Syntax error</b>  An unrecognized command or data type was encountered. For example, a string was received when the device does not accept strings.
<b>-103</b>	<b>Invalid separator</b>  The parser was expecting a separator and encountered an illegal character. For example, the semicolon was omitted after a program message unit.

- 104**            **Data type error**
- The parser recognized a data element that is not allowed. For example, numeric or string data was expected, but block data was encountered.
- 105**            **GET not allowed**
- A Group Execute Trigger was received within a program message (see IEEE 488.2, 7.7). Correct the HP-IB controller program so that the **GET** does not occur within a line of HP-IB program code.
- 108**            **Parameter not allowed**
- More parameters were received than expected for the header. For example, the **\*ESE** common command only accepts one parameter, so receiving **\*ESE 0,1** is not allowed.
- 109**            **Missing parameter**
- Fewer parameters were received than required for the header. For example, the **\*ESE** common command requires one parameter, so receiving **\*ESE** is not allowed.
- 110**            **Command header error**
- An error was detected in the header. This message is used when the device cannot detect the more specific errors described for errors -111 through -119.
- 111**            **Header separator error**
- A character which is not a legal header separator was encountered while parsing the header.
- 112**            **Program mnemonic too long**
- The header contains more than twelve characters (see IEEE 488.2, 7.6.1.4.1).
- 113**            **Undefined header**
- The header is syntactically correct, but it is undefined for this specific device. For example, **\*XYZ** is not defined for any device.
- 114**            **Header suffix out of range**
- The value of a header suffix attached to a program mnemonic makes the header invalid.
- 120**            **Numeric data error**
- This error, as well as errors -121 through -129, are generated when parsing a data element which appears to be numeric, including non-decimal numeric types. This particular error message is used if the device cannot detect a more specific error.



- 121**            **Invalid character in number**  
An invalid character for the data type being parsed was encountered. For example, an alpha in a decimal numeric or a “9” in octal data.
- 123**            **Exponent too large**  
The magnitude of an exponent was greater than 32000 (see IEEE 488.2, 7.7.2.4.1).
- 124**            **Too many digits**  
The mantissa of a decimal-numeric data element contained more than 255 digits excluding leading zeros (see IEEE 488.2, 7.7.2.4.1).
- 128**            **Numeric data not allowed**  
A legal numeric data element was received, but the device does not accept one in this position for the header.
- 130**            **Suffix error**  
This error, as well as errors -131 through -139, are generated when parsing a suffix. This particular error message is used if the device cannot detect a more specific error.
- 131**            **Invalid suffix**  
The suffix does not follow the syntax described in IEEE 488.2, 7.7.3.2, or the suffix is inappropriate for this device.
- 134**            **Suffix too long**  
The suffix contained more than twelve characters (see IEEE 488.2, 7.7.3.4).
- 138**            **Suffix not allowed**  
A suffix was encountered after a numeric element which does not allow suffixes.
- 140**            **Character data error**  
This error, as well as errors -141 through -149, are generated when parsing a character data element. This particular error message is used if the device cannot detect a more specific error.
- 141**            **Invalid character data**  
Either the character data element contains an invalid character or the particular element received is not valid for the header.
- 144**            **Character data too long**  
The character data element contains more than twelve characters (see IEEE 488.2, 7.7.1.4).

- 148**            **Character data not allowed**  
A legal character data element was encountered where prohibited by the device.
- 150**            **String data error**  
This error, as well as errors -151 through -159, are generated when parsing a string data element. This particular error message is used if the device cannot detect a more specific error.
- 151**            **Invalid string data**  
A string data element was expected, but was invalid (see IEEE 488.2, 7.7.5.2). For example, an **END** message was received before the terminal quote character.
- 158**            **String data not allowed**  
A string data element was encountered, but not allowed by the device at this point in the parsing.
- 160**            **Block data error**  
This error, as well as errors -161 through -169, are generated when parsing a block data element. This particular error message is used if the device cannot detect a more specific error.
- 161**            **Invalid block data**  
A block data element was expected, but was invalid (see IEEE 488.2, 7.7.6.2). For example, an **END** message was received before the end length was satisfied.
- 168**            **Block data not allowed**  
A legal block data element was encountered, but not allowed by the device at this point in the parsing.
- 170**            **Expression data error**  
This error, as well as errors -171 through -179, are generated when parsing an expression data element. This particular error message is used if the device cannot detect a more specific error.
- 171**            **Invalid expression**  
The expression data element was invalid (see IEEE 488.2, 7.7.7.2). For example, unmatched parentheses or an illegal character.
- 178**            **Expression data not allowed**  
A legal expression data was encountered, but was not allowed by the device at this point in parsing.

Error Messages  
Command Error

- 180**            **Macro error**
- This error, as well as errors -181 through -189, are generated when defining a macro or executing a macro. This particular error message is used if the device cannot detect a more specific error.
- 181**            **Invalid outside macro definition**
- Indicates that a macro parameter placeholder (**\$<number>**) was encountered outside of a macro definition.
- 183**            **Invalid inside macro definition**
- Indicates that the program message unit sequence, sent with a **\*DDT** or a **\*DMC** command, is syntactically invalid (see IEEE 488.2, 10.7.6.3).
- 184**            **Macro parameter error**
- Indicates that a command inside the macro definition had the wrong number or type of parameters.

---

## Execution Error

An error number in the range [-299 to -200] indicates that an error has been detected by the instrument's execution control block. The occurrence of any error in this class will cause the execution error bit (bit 4) in the event status register (IEEE 488.2, section 11.5.1) to be set. If this bit is set, one of the following events has occurred:

- A <PROGRAM DATA> element following a header was evaluated by the device as outside of its legal input range or is otherwise inconsistent with the device's capabilities.
- A valid program message could not be properly executed due to some device condition.

Execution errors will be reported by the device after rounding and expression evaluation operations have been completed. Rounding a numeric data element, for example, will not be reported as an execution error. Events that generate execution errors will not generate command errors, device-specific errors, or query errors; see the error definitions in this chapter.

## Execution Error Message Descriptions

This section lists the signal generator's execution error messages and their associated descriptions.

**Error Number**    Error Description [description/explanation/examples]

<b>-200</b>	<b>Execution Error</b>  This is a generic syntax error for devices that cannot detect more specific errors. The code indicates only that an execution error as defined in IEEE 488.2, 11.5.1.1.5 has occurred.
<b>-201</b>	<b>Invalid while in local</b>  Indicates that a command is not executable while the device is in local mode due to a hard local control (see IEEE 488.2, 5.6.1.5). For example, a device with a rotary switch receives a message which would change the switch's state, but the device is in local so the message cannot be executed.
<b>-202</b>	<b>Settings lost due to rtl</b>  Indicates that a setting associated with a hard local control (see IEEE 488.2, 5.6.15) was lost when the device changed to LOCS from REMS or to LWLS from RWLS.
<b>-210</b>	<b>Trigger error</b>  Indicates that a <b>GET</b> , <b>*TRG</b> , or a triggering signal could not be executed due to an error.

- 211**            **Trigger ignored**
- Indicates that a **GET**, **\*TRG**, or triggering signal was received and recognized by the device, but was ignored because of device timing considerations. For example, the device was not ready to respond.
- 212**            **Arm ignored**
- Indicates that an arming signal was received and recognized by the device but was ignored.
- 213**            **Init ignored**
- Indicates that a request for a measurement initiation was ignored as another measurement was already in progress.
- 214**            **Trigger deadlock**
- Indicates that a trigger source for the initiation of a measurement is set to **GET** and a subsequent measurement query is received. The measurement cannot begin until a **GET** is received, but the **GET** would cause an **INTERRUPTED** error.
- 215**            **Arm deadlock**
- Indicates that the arm source for the initiation of a measurement is set to **GET** and a subsequent measurement query is received. The measurement cannot begin until a **GET** is received, but the **GET** would cause an **INTERRUPTED** error.
- 220**            **Parameter error**
- Indicates that a program data element related error has occurred. This particular error message is used if the device cannot detect a more specific errors described for errors -221 through -229.
- 221**            **Settings conflict**
- Indicates that a legal program data element was parsed but could not be executed due to the current device state (see IEEE 488.2 11.5.1.1.5).
- 222**            **Data out of range**
- Indicates that a legal program data element was parsed but could not be executed because the interpreted value was outside the legal range defined by the device (see IEEE 488.2 11.5.1.1.5).
- 223**            **Too much data**
- Indicates that a legal program data element of block, expression or string type was received that contained more data than the device could handle due to memory or related device-specific requirements.

- 224**            **Illegal parameter value**  
Used where exact value, from a list of possibilities, was expected.
- 225**            **Out of memory**  
The device has insufficient memory to perform the requested operation.
- 226**            **Lists not same length**  
Attempted to use LIST structure having individual LISTS of unequal length.
- 230**            **Data corrupt or stale**  
Possibly invalid data. A new reading was started but not completed since last access.
- 231**            **Data questionable**  
Indicates that the measurement accuracy is questionable.
- 232**            **Invalid format**  
Indicates that a legal program data element was parsed but could not be executed because the data format or structure is inappropriate. For example, when loading memory tables or when sending a **SYSTEM:SET** parameter for an unknown instrument.
- 233**            **Invalid version**  
Indicates that a legal program data element was parsed but could not be executed because the version of the data is incorrect to the device. This particular error is used when file or block data elements are recognized by the instrument, but cannot be executed for reasons of version incompatibility. For example, a non- supported file version or a non- supported instrument version.
- 240**            **Hardware error**  
Indicates that a legal program command or query could not be executed because of a hardware problem in the device. The definition of what constitutes a hardware problem is completely device-specific. This error is used when the device cannot detect the more specific errors described for errors -241 through -249.
- 241**            **Hardware missing**  
Indicates that a legal program command or query could not be executed because of missing device hardware. For example, an option was not installed.

- 250**            **Mass storage error**
- Indicates that a mass storage error has occurred. This message is used when a device cannot detect the more specific errors described for errors -251 through -259.
- 252**            **Missing media**
- Indicates that a legal program command or query could not be executed because of missing media, for instance no disk in the disk drive. The definition of what constitutes missing media is device-specific.
- 253**            **Corrupt media**
- Indicates that a legal program command or query could not be executed because of corrupt media, for instance a bad disk or incorrect disk format. The definition of what constitutes corrupt media is device-specific.
- 254**            **Media full**
- Indicates that a legal program command or query could not be executed because the media was full. For example, there is was no space left on the disk. The definition of what constitutes full media is device-specific.
- 255**            **Directory full**
- Indicates that a legal program command or query could not be executed because the media directory was full. The definition of what constitutes a full media directory is device-specific.
- 256**            **File name not found**
- Indicates that a legal program command or query could not be executed because the file name on the device media could not be found. For example, an attempt was made to read or copy a nonexistent file. The definition of what constitutes a file not being found is device-specific.
- 257**            **File name error**
- Indicates that a legal program command or query could not be executed because a file name on the device media was in error. For example, an attempt was made to copy to a duplicate filename. The definition of what constitutes a file name error is device-specific.
- 258**            **Media protected**
- Indicates that the device or user has attempted to write to a read-only memory subsystem (msus). The definition of a protected media is device-specific.

- 260**            **Expression error**
- Indicates that an expression data element-related error occurred. This error message is used when the device cannot detect the more specific errors described for errors -261 through -269.
- 261**            **Math error in expression**
- Indicates that a syntactically legal expression program data element could not be executed due to a math error. For example, a divide-by-zero was attempted. The definition of a math error is device-specific.
- 270**            **Macro error**
- Indicates that a macro-related execution error occurred. This error message is used when the device cannot detect the more specific errors described for errors -271 through -279.
- 271**            **Macro syntax error**
- Indicates that a syntactically legal macro program data sequence, written in accordance with IEEE 488.2, 10.7.2, could not be executed due to a syntax error within the macro definition (see IEEE 488.2, 10.7.6.3).
- 272**            **Macro execution error**
- Indicates that a syntactically legal macro program data sequence could not be executed due to an error within the macro definition (see IEEE 488.2, 10.7.6.3).
- 273**            **Illegal macro label**
- Indicates that the macro label defined in the **\*DMC** command was a legal string syntax, but could not be accepted by the device (see IEEE 488.2, 10.7.3 and 10.7.6.2). For example, the label was too long, the same as a common command header, or contained invalid header syntax.
- 274**            **Macro parameter error**
- Indicates that the macro definition improperly used a macro parameter placeholder (see IEEE 488.2, 10.7.3).
- 275**            **Macro definition too long**
- Indicates that a syntactically legal macro program data sequence could not be executed because the string or block contents were too long for the device to handle (see IEEE 488.2, 10.7.6.1).
- 276**            **Macro recursion error**
- Indicates that a syntactically legal macro program data sequence could not be executed because the device found it to be recursive (see IEEE 488.2, 10.7.6.4).



- 277**            **Macro redefinition not allowed**
- Indicates that the macro label defined in the **\*DMC** command could not be executed because the macro label was already defined (see IEEE 488.2, 10.7.6.4).
- 278**            **Macro header not found**
- Indicates that a syntactically legal macro label in the **\*GMC?** query could not be executed because the header was not previously defined.
- 280**            **Program error**
- Indicates that a downloaded program-related execution error occurred. This error message is used when the device cannot detect the more specific errors described for errors -281 through -289. The syntax used in a program and the mechanism for downloading a program is device-specific.
- 281**            **Cannot create program**
- Indicates that an attempt to create a program was unsuccessful. This may be due to insufficient memory.
- 282**            **Illegal program name**
- Indicates that the name used to reference a program was invalid. For example, redefining an existing program, deleting a nonexistent program, or in general, referencing a nonexistent program.
- 283**            **Illegal variable name**
- Indicates that an attempt was made to reference a nonexistent variable.
- 284**            **Program currently running**
- Indicates that certain operation related to programs may be illegal while the program is running. For example, deleting a running program may be illegal.
- 285**            **Program syntax error**
- Indicates that a syntax error appears within a downloaded program. The syntax used when parsing a downloaded program is device-specific.
- 286**            **Program runtime error**
- Indicates that a runtime error was detected in a downloaded program.
- 290**            **Memory use error**
- Indicates that a user request has directly or indirectly caused an error related to memory or <data\_handles>. This is not the same as “bad” memory.

- 291**            **Out of memory**  
A downloaded program required more memory than was available in the instrument.
- 292**            **Referenced name does not exist**  
A downloaded program attempted to access an undefined element (a variable, constant, filename, etc.).
- 293**            **Referenced name already exists**  
A downloaded program attempted to define an element (a variable, constant, filename, etc.) that had already been defined.
- 294**            **Incompatible type**  
Indicates that the type or structure of a memory item is inadequate.

---

## Device-Specific Error

An error number in the range [-399 to -300] or [1 to 32767] indicates that the instrument has detected an error which is not a command error, a query error, or an execution error; some device operations did not properly complete, possibly due to an abnormal hardware or firmware condition. These codes are also used for self-test response errors. The occurrence of any error in this class will cause the device-specific error bit (bit 3) in the event status register (IEEE 488.2, section 11.5.1) to be set.

---

**NOTE:** For positive error number descriptions see the section titled “ESG Series Signal Generator Instrument-Specific Error Messages.”

---

The meaning of positive error codes is device-dependent and may be enumerated or bit mapped. The <error\_message> string for positive error codes is not defined by SCPI. The occurrence of any error in this class will cause the device-specific error bit (bit 3) in the event status register (IEEE 488.2, section 11.5.1) to be set. Events that generate device-specific errors will not generate command errors, or query errors; see the other error definitions in this section.

## Device-Specific Error Message Descriptions

This section lists the signal generator’s device-specific error messages and their associated descriptions.

<b>Error Number</b>	<b>Error Description [description/explanation/examples]</b>
<b>-300</b>	<b>Device-specific error</b>  This is a generic device-dependent error for devices that cannot detect more specific errors. The code indicates only that a device-dependent error as defined in IEEE 488.2, 11.5.1.1.6 has occurred.
<b>-310</b>	<b>System error</b>  Indicates that an error, termed “system error” by the device, has occurred.
<b>-311</b>	<b>Memory error</b>  Indicates that an error was detected in the device’s memory.
<b>-312</b>	<b>PUD memory lost</b>  Indicates that the protected user data saved by the *PUD command has been lost.
<b>-313</b>	<b>Calibration memory lost</b>  Indicates that non-volatile calibration data has been lost.

- 314**            **Save/recall memory loss**  
Indicates that the non-volatile data saved by the \*SAV? command has been lost.
- 315**            **Configuration memory lost**  
Indicates that non-volatile configuration data saved by the device has been lost. The meaning of this error is device-dependent.
- 320**            **Storage fault**  
Indicates that the firmware detected a fault when using data storage. This error is not an indication of physical damage or failure of any mass storage element.
- 321**            **Out of memory**  
Indicates that an internal operation needed more memory than was available.
- 330**            **Self-test failed**  
Indicates that the device has detected a failure during its self-test procedure.
- 340**            **Calibration failed**  
Indicates that the device has detected a failure during its calibration procedure.
- 350**            **Queue overflow**  
This is a specific code entered into the queue in lieu of the code that caused the error. This message indicates that there is no more room in the queue and an error occurred but was not recorded.
- 360**            **Communication error**  
This is the generic communication error for devices that cannot detect the more specific errors described for errors -361 through -363.
- 361**            **Parity error in program message**  
Indicates that the parity bit was not correct when data was received (for example, an incorrect parity bit on a serial port).
- 362**            **Framing error in program message**  
Indicates that a stop bit was not detected when data was received (for example, a baud rate mismatch).

---

## Query Error

An error number in the range [-499 to -400] indicates that the output queue control of the instrument has detected a problem with the message exchange protocol described in IEEE 488.2, chapter 6. The occurrence of any error in this class will cause the query error bit (bit 2) to be set in the event status register (IEEE 488.2, section 11.5.1). These errors correspond to message exchange protocol errors described in IEEE 488.2, 6.5. If a query error occurs one of the following is true:

- An attempt is being made to read data from the output queue when no output is either present or pending.
- Data in the output queue has been lost.

Events that generate query errors will not generate command errors, execution errors, or device-specific errors; see the other error definitions in this section.

### Query Error Message Descriptions

This section lists the signal generator's query error messages and their associated descriptions.

**Error Number**    Error Description [description/explanation/examples]

<b>-400</b>	<b>Query Error</b>  This is a generic query error for devices that cannot detect more specific errors. The code indicates only that a query error as defined in IEE 488.2, 11.5.1.1.7 and 6.3 has occurred.
<b>-410</b>	<b>Query INTERRUPTED</b>  Indicates that a condition causing an INTERRUPTED query error occurred (see IEEE 488.2, 6.3.2.7). For example, a query was followed by <b>DAB</b> or <b>GET</b> before a response was completely sent.
<b>-420</b>	<b>Query UNTERMINATED</b>  Indicates that a condition causing an UNTERMINATED query error occurred (see IEEE 488.2, 6.3.2.2). For example, the device was addressed to talk and an incomplete program message was received.
<b>-430</b>	<b>Query DEADLOCKED</b>  Indicates that a condition causing a DEADLOCKED query error occurred (see IEEE 488.2, 6.3.1.7). For example, both the input buffer and the output buffer are full and the device cannot continue.

**-440**

**Query UNTERMINATED after indefinite response**

Indicates that a query was received in the same program message after a query requesting an indefinite response was executed (see IEEE 488.2, 6.3.7.5).

---

## ESG Series Signal Generator Instrument-Specific Error Messages

Some instrument-specific error messages incorporate the negative or “generic” SCPI error number with the addition of device-dependent/instrument-specific (ESG Series signal generator-specific) information following the semicolon in the error message.

A positive error number indicates that the instrument has detected an error within the HP-IB system, within the instrument’s firmware or hardware, during the transfer of block data, or during calibration.

**Error Number**    Error Description [description/explanation/examples]

**-100**

**Command error;R0:No standby mode allowed.**

Indicates that, in HP 8656/57 compatibility mode, **R0** was received via HP-IB. This command is not supported by the compatibility mode.

**Command error;Remote active function DN/UP not available.**

Indicates that, in HP 8656/57 compatibility mode, either **DN** or **UP** was received via HP-IB. These commands are not supported by the compatibility mode.

**Command error;LO: No low bandwidth ALC mode allowed.**

Indicates that, in HP 8656/57 compatibility mode, **LO** was received via HP-IB. This command is not supported by the compatibility mode.

**-102**

**Syntax error;Bad HP compatibility language character <character>.**

Indicates that, in HP 8656/57 compatibility mode, illegal language input was received.

**Syntax error;Bad HP compatibility language token <token>.**

Indicates that, in HP 8656/57 compatibility mode, a known command or termination specifier was received when it was not expected. For example, a termination specifier was received with no currently active function.

- 213**            **Init ignored;Unable to sweep due to sweep being in an error state. The sweep error should be fixed.**
- Indicates that the number of list, power, and/or dwell points are in conflict, or a serious system error has occurred in list/sweep. A previous error report should have described the error that is stalling list/sweep.
- Init ignored;Cannot initiate sweep in manual mode.**
- Indicates that the manual mode is on and therefore the instrument cannot sweep.
- Init ignored;Sweep is already initiated.**
- Indicates that the list/sweep is currently initiated and sweeping, therefore the command is not legal according to SCPI.
- Init ignored;Sweep is already continuously initiated.**
- Indicates that the list/sweep is continuously initiated and sweeping, therefore the command is not legal according to SCPI.
- 221**            **Settings conflict;Frequency list and dwell list are of unequal size. Set one list equal to size one, or make their sizes equal.**
- Indicates that the frequency list has more than one element and the dwell list has more than one element, and they are not of equal size. If any of the frequency, power, or dwell lists have more than one element, they must all have the same number of elements. A list of a single element is the same as a list of equal size with the single element repeated the necessary number of times.
- Settings conflict;Frequency list and power list are of unequal size. Turn one list off, set one to size one, or make their sizes equal.**
- Indicates that the frequency list has more than one element and the power list has more than one element, and they are not of equal size. If any of the frequency, power, or dwell lists have more than one element, they must all have the same number of elements. A list of a single element is the same as a list of equal size with the single element repeated the necessary number of times.
- Settings conflict;Power list and dwell list are of unequal size. Set one to size one, or make their sizes equal.**
- Indicates that the dwell list has more than one element and the power list has more than one element, and they are not of equal size. If any of the frequency, power, or dwell lists have more than one element, they must all have the same number of elements. A list of a single element is the same as a list of equal size with the single element repeated the necessary number of times.



**Settings conflict;The selected external trigger setting conflicts with the previous setting.**

Indicates that the external trigger has been set to positive edge for one trigger source and negative edge for another trigger source.

**Settings conflict;FM2/PM2 value set greater than FM1/PM1 value. FM1/PM1 changed to match FM2/PM2 value.**

The deviation of FM2/PM2 must always be less than or equal to the deviation settings for FM1/PM1. This error will be reported to the queue when FM1/PM1 is enabled and FM2/PM2 is also enabled and an adjustment to either FM2/PM2 deviation causes the FM2 or PM2 deviation to be greater than the FM1 or PM1 deviation. It will also be reported when FM2/PM2 is being turned on, and the last FM1/PM1 deviation setting is less than the current FM2/PM2 deviation setting. In both cases the FM1/PM1 deviation will be adjusted to match the FM2/PM2 deviation.

**Settings conflict;FM1/PM1 value set less than FM2/PM2 value. FM2/PM2 changed to match FM1/PM1 value.**

The deviation of FM2/PM2 must always be less than or equal to the deviation settings for FM1/PM1. This error will be reported to the queue when FM2/PM2 is enabled and FM1/PM1 is also enabled and an adjustment to either FM1/PM1 deviation causes the FM1 or PM1 deviation to be less than the FM2 or PM2 deviation. It will also be reported when FM1/PM1 is being turned on, and the last FM2/PM2 deviation setting is greater than the current FM1/PM1 deviation setting. In both cases the FM2/PM2 deviation will be adjusted to match the FM1/PM1 deviation.

**Settings conflict;Enabled mod source conflicts with previously enabled mod source. Previous mod disabled.**

The signal generator has three sources: INT, EXT1, and EXT2 that are shared by the FM1/PM1, AM1/AM2, FM2/PM2, pulse (INT and EXT2), and burst envelope (EXT1 only). Each source can only be used by one of the modulations at a time. If a source is being used by an active modulation, and a request for the source is made by another modulation, the first modulation will be turned off, the second modulation will be turned on.

**Settings conflict;FM & PM not allowed.**

Indicates that there is a hardware conflict between FM and PM. The most recently requested modulation will be turned on, the previous modulation will be turned off.

**Settings conflict;Pattern repeat is changed to continuous because data source is external.**

Indicates that, while in non-bursted data generation, Pattern Repeat was in Single mode and data source was selected to be External. For non-bursted data generation using an external data source, Pattern Repeat must be in Continuous mode. To continue data transmission, Pattern repeat has been changed to Continuous mode.

-222

**Data out of range;value clipped to lower limit.**

Indicates that an input value is below the minimum value allowed. Examples are: frequency setting, reference, or offset; output power; power reference and offset; modulation depth, deviation, or modulation source frequency; number of points and start/stop values for list mode; sequence or register values (save/recall); dwell time.

**Data out of range;value clipped to upper limit.**

Indicates that an input value is above the maximum value allowed. Examples are: frequency setting, reference, or offset; output power; power reference and offset; modulation depth, deviation, or modulation source frequency; number of points and start/stop values for list mode; sequence or register values (save/recall); dwell time.

**Data out of range;Synthesizer: Frequency out of bounds.**

Indicates that the instrument received an internal request for a frequency outside of its supported frequency range. Report the circumstances to the factory.

**Data out of range;Manual point exceeds list sizes. Limiting to maximum point.**

Indicates that the sweep/list manual point has been reassigned to a smaller number value due to the longest list decreasing in size or being turned off. Its new value is the length of the longest enabled list (frequency or power).

**Data out of range;Manual point exceeds frequency list size. Limiting to maximum point.**

Indicates that the sweep/list manual point has been reassigned to a smaller number value due to the longest list decreasing in size or being turned off. Its new value is the length of the frequency list which is the longest enabled list.

**Data out of range;Manual point exceeds power list size. Limiting to maximum point.**

Indicates that the sweep/list manual point has been reassigned to a smaller number value due to the longest list shrinking, or being turned off. Its new value is the length of the power list, which is the longest enabled list.

**-223 Too Much Data;The number of list points exceeds the maximum allowed.**

Indicates that a SCPI list has been entered that is longer than the maximum allowed length, which is also the maximum number of step points; too many points were given for a frequency, amplitude, or dwell time list. This error can also be caused by attempting to copy items in the list editor when the list is already at its maximum length.

**-230 Data corrupt or stale;RAM copy of <filename>.**

The non-volatile RAM copy of a file is either corrupt or is out of date with the EEPROM master copy (if one exists). The system automatically re-initializes the file from EEPROM (if appropriate) or from a default algorithm. A potential cause is a failing backup battery.

**Data corrupt or stale;EEPROM copy of <filename>.**

The EEPROM copy of a file is either corrupt or otherwise unusable. The system automatically updates the non-volatile RAM copy of the EEPROM copy using a default initialization. The actual EEPROM file is left as it is. Report this problem to the factory.

**-231 Data questionable;RAM copy of <filename>.**

Indicates that the non-volatile RAM copy of a file has a correctable error. The system automatically performs the correction. A potential cause is a failing backup battery.

**Data questionable;EEPROM copy of <filename>.**

Indicates that the EEPROM copy of a file has a correctable error. The system automatically performs the correction. A potential cause is a failing EEPROM. Report this problem to the factory.

**-241 Hardware missing; <card\_name>**

Indicates that a test communication to a hardware card failed. The instrument is most likely not functional. Contact the nearest HP Sales and Service office.

**Hardware missing; Installed option boards do not match configuration information.**

Indicates that a set of option boards have been installed that do not match the information that was given to the instrument as part of the installation. If this is the result of a customer installed option, the wrong option was specified during installation. If this is seen at any other time, the likely cause is an EEPROM failure on the option card.

**-250 Mass storage error; EEPROM write timeout on <filename>.**

Indicates that the system was not able to program new data to an EEPROM. The system is still functional, but files written to EEPROM (such as updated calibration data) may be lost when the instrument's line power is cycled. Contact the nearest HP Sales and Service office.

**-253 Corrupt media; User File System**

Indicates that the main memory area used for storing instrument states and sequences as well as other data files is corrupt. The system will automatically clear and reconfigure this memory area. A potential cause is a failing backup battery. Another potential cause could be the loss of line power to the instrument in the middle of a write operation.

**Corrupt media; <media\_name>**

Indicates that a source media (possibly EEPROM) for a data file is corrupt. This error is usually seen in conjunction with errors concerning a certain file.

**-254 Media full; Unable to delete saved state from non-volatile memory. No instrument state change.**

Indicates that the state memory subsystem **STATE:** was unable to delete a register. You must free some memory by deleting a file or register using Catalog. Afterwards, try again.

**Media full; Save a state register ignored.**

Indicates that the state memory subsystem **STATE:** did not have enough room to save a register. You must free some memory by deleting a file or register using Catalog. Afterwards, try again.

**Media full; Save a state register failed. State marked available.**

Indicates that the state memory subsystem **STATE:** did not have enough room to save a register, so the register was lost and is now marked available. You must free some memory by deleting a file or register using Catalog. Afterwards, try again.

**-256** File name not found;The internal list file was not found. There is no list data to return

Indicates that the **DWEL\_FILE**, **FREQ\_FILE**, or **POW\_FILE** has been lost, so a new one will have to be created. These files are the persistent information for list/sweep mode. They contain the dwell list, the frequency list, or the power list. Invoking the list editor will recreate the missing file to a length of one element.

**-257** File name error;Delete empty sequence <sequence\_name>. Delete sequence ignored.

Indicates that the user has attempted to delete a sequence which is empty (all registers unused). This is informational only. Typically this error is reported (several times) when the “Delete All Sequences” command is executed.

**File name error;Delete a non-saved state register. Delete register ignored.**

Indicates that the user has attempted to delete a state which is empty (unused). This is informational only.

**File name error;Directory does not support extenders.**

Indicates that an extender, which is specified by an @ sign followed by a memory subsystem name, has been specified for an explicit memory subsystem which does not allow the @ notation. Only the default (:) memory subsystem allows extenders.

**File name error;Empty filename**

Indicates that a filename of " " was specified. This is not a legal filename.

**File name error;Illegal extender**

Indicates that an illegal memory subsystem name was used after the @. Supported values are **@STATE** and **@LIST**.

**File name error;Illegal filename character**

Indicates that an illegal character was used within a filename. \, :, @ and all non-printable ASCII characters are illegal in filenames.

**File name error;Only one ":" is allowed.**

Indicates that only one colon is allowed in any filename specification. The text before the colon is a user memory subsystem. The valid user choices are **:**, **DEFAULT:**, **STATE:**, and **LIST:**.

- File name error;Only one "@" is allowed.**  
Indicates that only one @ is allowed in any filename specification. It specifies the memory subsystem that a user file actually resides in.
- 286 Program runtime error;Floating-Point Exception**  
Indicates that a floating-point math error (such as a divide by zero) has been detected. The system will attempt to recover automatically. Report the circumstances to the nearest HP Sales and Service office.
- 310 System error;RS232 buffer overflow: character lost.**  
Indicates that the RS232 buffer has been exceeded. The most recent character has been dropped.
- System error;Cannot change manual point until list mode error condition cleared.**  
An error is keeping the sweep/list from being able to set the frequency and/or power. Until the problem is addressed, the manual point cannot be changed.
- System error;Unable to determine which attenuator is installed.**  
Indicates that an invalid attenuator identification code has been detected. Possible causes include a loose attenuator control cable. The instrument will likely not produce the proper output power levels. Report this error to the factory.
- 311 Memory error;Unable to configure Save Recall registers from non-volatile memory. Save Recall registers re-initialized.**  
Indicates that saved states are no longer usable. Delete explicitly using Catalog.
- 315 Configuration memory lost;Persistent state preset. Using factory defaults.**  
Indicates that the persistent state has been forced to return to factory preset values.
- Configuration memory lost;Persistent state version is bad. Using factory defaults.**  
Indicates that the persistent state version is not recognized as valid and is assumed to be corrupt. The persistent state is reinitialized with the factory preset values.
- Configuration memory lost;Persistent state checksum is bad. Using factory defaults.**  
Indicates that the persistent state is corrupt and had to be reinitialized with the factory preset values.

-321

**Out of memory;Unable to verify instrument state file.**

Indicates that an instrument state file could not be accessed and verified because of insufficient memory. Reduce the size of any sweep lists and try again.

**Out of memory;Memory catalog failed.**

Indicates that there is not enough memory to complete a catalog listing. Reduce the size of any sweep lists and try again.

**Out of memory;Unable to display timeslot window.**

Indicates that the instrument was unable to create part of the graphical user interface due to an inability to allocate memory (possibly due to fragmentation). Please report the circumstances to the factory. The instrument is still functional.

**Out of memory;Unable to display protocol window.**

Indicates that the instrument was unable to create part of the graphical user interface due to an inability to allocate memory (possibly due to fragmentation). Please report the circumstances to the factory. The instrument is still functional.

**Out of memory;Unable to display format window.**

Indicates that the instrument was unable to create part of the graphical user interface due to an inability to allocate memory (possibly due to fragmentation). Please report the circumstances to the factory. The instrument is still functional.

**Out of memory;Cannot uncompress file.**

Indicates that a **STATE:** file cannot be uncompressed because there is not enough memory to run the decompression algorithm. Recall will fail and there will be no instrument state change. Reduce the size of any sweep lists and try again.

**Out of memory;Cannot precalculate frequencies. Try fewer frequencies.**

Indicates that memory was exhausted during frequency precalculation (used to speed the process of sweep/list mode). List mode cannot run until either fewer frequencies have been supplied or more memory becomes available and the same set of frequencies are sent again, **FREQ:MODE CW** is executed, or **:FREQ:MODE LIST** is executed.

**Out of memory;Object Memory Area**

Indicates that memory was exhausted during instrument power-on. Report the circumstances to the factory.

**Out of memory;List formation**

The device was unable to allocate space for a lookup table, such as for list mode precalculation. List mode cannot run until either fewer frequencies have been supplied or more memory becomes available and the same set of frequencies are sent again, **FREQ:MODE CW** is executed, or **:FREQ:MODE LIST** is executed.

**Out of memory;Display system out of memory. An abnormal display may result. Memory consumption should be reduced.**

There was not enough memory in the system to properly update the display. Some inconsistencies may be seen. The size of any list/sweep should be reduced, and the source should be preset to clear up any inconsistencies. Report the circumstances to the nearest HP Sales and Service office.

**Out of memory;Unable to check Data Generator memory.**

There was not enough memory in the system to properly complete the data generator memory test. This does NOT imply a data generator memory failure. Check all other error messages to identify possible causes, discontinue list/sweep mode to free some memory, and repeat the test.

-330

**Self-test failed;Power supply self-test failure**

Indicates that the self-test for a particular power supply voltage has failed. The instrument is likely not functional. Contact the nearest HP Sales and Service office.

**Self-test failed;EEPROM header checksum error <card\_name>.**

Indicates that the card identification header for a hardware card is incorrect. If the card is not properly identified, the instrument is likely to be non-functional. Contact the nearest HP Sales and Service office.

**Self-test failed;Data Generator Memory Test @ 0x\_\_\_\_\_**

Indicates that the data generator memory failed. Modulation data produced by the data generator may not be correct. However, if an **Unable to check Data Generator Memory** error was also seen, this result is not conclusive. The address of the first location that failed is reported. Contact the nearest HP Sales and Service office.

**Self-test failed;Burst Generator Memory Test @ 0x\_\_\_\_\_**

Indicates that the burst generator memory failed. Modulation data produced by the burst generator may not be correct. However, if an **Unable to check Burst Generator Memory** error was also seen, this result is not conclusive. The address of the first location that failed is reported. Contact the nearest HP Sales and Service office.



**Self-test failed;Bad address position @ 0x\_\_\_\_\_**

Indicates that the data generator memory failed. Modulation data produced by the data generator may not be correct. An address that appeared to have a failed address line was reported. Contact the nearest HP Sales and Service office.

**Self-test failed;Chips \_\_\_, \_\_\_ aliased @ 0x\_\_\_\_\_**

Indicates that the data generator memory failed. Modulation data produced by the data generator may not be correct. An address that appeared to be aliased across multiple memory chips has been reported. Contact the nearest HP Sales and Service office.

**-430**

**Query DEADLOCKED**

Indicates that a SCPI output queue has filled preventing further SCPI command execution, and there is no more room left in the corresponding SCPI input queue to accept a query to read from the output queue. The system automatically discards output to correct the deadlock.

**201**

**Bad file number;Unable to check Data Generator memory.**

Indicates that the instrument was not able to generate the pattern necessary to perform the data generator memory test. This does NOT imply a data generator memory failure. Report the problem to the factory.

**208**

**I/O error;Unable to delete saved state from non-volatile memory.  
No instrument state change.**

Indicates that a **STATE:** file could not be deleted due to the file not being found, file corruption, or another file-related problem. If the file is displayed by a memory catalog, delete it explicitly.

**I/O error;Save a state register ignored.**

Indicates that a **STATE:** file could not be saved due to insufficient space, file corruption, or another related problem.

**I/O error;Delete empty sequence <sequence\_name>. Delete sequence ignored.**

Indicates that the user has attempted to delete a sequence that is empty. This error message is informational only. Typically, this error is reported several times when the “Delete All Sequences” command is executed. If the file is displayed by Catalog, delete explicitly.

**I/O error;Delete a non-saved state register. Delete register ignored.**

Indicates that the user has attempted to delete an unused (empty) state. This error message is informational only.

**I/O error;Trailing zero found in <filename>. Fixing...**

Indicates that a compressed state file has a zero at its end. This is a sign of file corruption. The device fixes the problem by concealing the zero such that it no longer triggers an error message. The file may be corrupt or unusable.

**I/O error;Unable to recall from non-volatile memory. No instrument state change.**

Indicates that the state file is not readable and the recall was aborted.

**214 Not owner;Unable to delete saved state from non-volatile memory. No instrument state change.**

Indicates that the user has attempted to write to a read-only memory subsystem.

**501 Attenuator hold setting over range;Frequency change forced attenuator adjust.**

Indicates that the firmware has changed the attenuator setting because, while in attenuator hold mode, a change in frequency setting has forced the ALC beyond its range.

**Attenuator hold setting over range;Power set to lower limit.**

Indicates that the firmware has changed the power setting to a value other than the requested value due to the fact that, while in attenuator hold mode, the user has requested a power setting that is below the ALC range for the attenuator setting. The power has been set to the lower limit.

**Attenuator hold setting over range;Power set to upper limit.**

Indicates that the firmware has changed the power setting to a value other than the requested value due to the fact that, while in attenuator hold mode, the user has requested a power setting that is above the ALC range for the attenuator setting.

**508 Synthesizer unlocked**

Indicates that the synthesizer is unlocked. Service may be needed.

**509 Output Section input overdrive**

Internal error: report to factory.

- 511**            **Output unlevelled**  
Indicates that the instrument's output is unlevelled.
- 512**            **Reference unlocked**  
Indicates that the instrument's reference is unlocked. If an external reference is connected, check the frequency and power. It is possible for this to occur during a poor connection/disconnection of an external reference. If this error reoccurs when no external reference is connected, the instrument may require service.
- 513**            **Het VCO unlocked**  
Indicates that the VCO used to generate output frequencies below 250 MHz is unlocked. The instrument may require service.
- 514**            **Reference Oven cold**  
Indicates that the reference oven is not at the required operating temperature. This is normal if the instrument has been powered down for a while. If the error persists, the instrument may require service.
- 515**            **Reference board: 10 Mhz reference signal bad or missing**  
Indicates that the instrument's reference is unlocked. If an external reference is connected, check the frequency and power. It is possible for this to occur during a poor connection/disconnection of an external reference. If this error reoccurs when no external reference is connected, the instrument may require service.
- 517**            **Calibration failure;DCFM DC overrange**  
Indicates that the instrument was unable to perform a DCFM or DC $\Phi$ M calibration due to the input signal being outside of the offset range that can be calibrated for.  
**Calibration failure;Upgrade calibration failed. Data not stored.**  
Indicates that the calibration stage of the instrument upgrade was not executed successfully. The calibration data has not been stored. The upgrade is not functional. Contact the nearest HP Sales and Service office.
- 600**            **RPP has tripped.**  
Indicates that the reverse power protection circuit has been triggered. Repeated tripping of this circuit can cause damage to the instrument.
- 601**            **Power search failed.**  
Indicates that, while executing power search, the level meter circuit failed to return a meaningful value. This event indicates that the power is in a range that the leveling loop cannot properly level. The power will be set to the last properly leveled power.

- 605**            **DSP FW download failed.**
- Indicates that the instrument's firmware was unable to successfully initialize the internal DSP. Report the circumstances to the nearest HP Sales and Service office.
- 606**            **DSP times out.**
- Indicates that the DSP failed to respond within the appropriate amount of time. Report the circumstances to the nearest HP Sales and Service office.
- 607**            **DSP returns error.**
- Indicates that the DSP is in an indeterminate state. Report the circumstances to the nearest HP Sales and Service office.
- 608**            **DSP in use by other process.**
- Indicates that the DSP is in an indeterminate state. Report the circumstances to the nearest HP Sales and Service office.
- 615**            **New wave shape changes limit for internal frequency;frequency changed to new limit.**
- When using the internal modulation source, the upper limit varies for the different waveforms. If the user changes the waveform when the internal source frequency is higher than that allowed for the new waveform, the frequency for the source will be changed, and the user informed of that change with this message.
- 617**            **Configuration error; Data Generator Memory configuration does not match installed board.**
- This indicates that the memory configuration for an option board does not match the known memory limits of the board. If this error has occurred as the result of a customer-installed option, uninstall all options and then reinstall the correct options. If the error persists, contact the factory.
- Configuration error; Installed option boards do not match configuration information.**
- This indicates that the option boards have not been properly installed in the instrument. Verify that the correct option boards have been installed in the correct slots. Reinstall the correct option. If the error persists, contact the factory.
- Configuration error; Invalid Data Generator memory configuration.**
- This indicates that the memory configuration for an option board does not match the known memory limits of any supported option board. If this error has occurred as the result of a customer-installed option, uninstall all options and then reinstall the correct options. If the error persists, contact the factory.

**Configuration error; Invalid option board configuration.**

This indicates that an invalid combination of option boards has been configured. If this error has occurred as the result of a customer-installed option, uninstall all options and then reinstall the correct options. If the error persists, contact the factory.

700

**State Save Recall Error; Recall aborted. Unable to recall the state from non-volatile memory.**

This indicates that the state file was not readable, so the recall was aborted. If state file exists, delete explicitly using the memory catalog.

**State Save Recall Error; Recalled state has a bad checksum. No instrument state change.**

This indicates that the state file was corrupt or out-of-date, so the recall was ignored. If state file exists, delete explicitly using the memory catalog.

**State Save Recall Error; Recall data different from FW revision. No instrument state change.**

Indicates that an attempt was made to recall a state that was saved with an incompatible version of the instrument firmware. This typically occurs when a state file is copied from an instrument with a newer version of firmware to an instrument with an older version of firmware. Newer versions of instrument firmware can read older state files.

**State Save Recall Error; Recall non-saved state register. Recall ignored.**

Indicates that a recall was attempted for a state register that is unused. If state file exists, delete explicitly using catalog.

**State Save Recall Error; Delete sequence <sequence\_name> ignored.**

Indicates that a **STATE:** file in a sequence that is being deleted could not be deleted due to the file not being found, data corruption, etc. If state file exists, delete explicitly using the memory catalog.

**State Save Recall Error; The state file is from a different firmware revision that does not support comments.**

Indicates that an attempt was made to write a comment to a state file revision that does not support comments. Comments in saved state files are not supported by the A.01.00 and A.01.01 releases of the instrument firmware.

---

## Index

### Symbols

\*CLS, 2-42  
\*ESE, 2-45  
\*ESR?, 2-44  
\*RST, 3-3  
\*SRE, 2-41  
\*STB?, 2-41  
<^END>, 2-28  
<new line>, 2-28

### Numerics

1 GHz reference unlocked, 3-28  
10 MHz reference  
  standard, 3-28  
  unlocked, 3-28

### A

abort command, 2-12  
ac coupling  
  setting, 3-14  
ac-coupled  
  external FM signal, 3-13  
  internal FM signal, 3-15  
address, 1-4, 3-2  
  HP-IB, 1-2  
  querying, 3-10  
AH1 mnemonic, 1-3  
AM subsystem, 5-7  
  Amplitude Modulation Depth, 5-14  
  Amplitude Modulation Depth Cou-  
  pling, 5-15  
  Amplitude Modulation Source, 5-12  
  Amplitude Modulation State, 5-13  
  Command Description, 5-7  
  External Amplitude Modulation  
  Source Coupling, 5-7  
  Internal Amplitude Modulation Alter-  
  nate Frequency, 5-9  
  Internal Amplitude Modulation Alter-  
  nate Frequency Amplitude, 5-9  
  Internal Amplitude Modulation  
  Source Rate, 5-8  
  Internal Amplitude Modulation  
  Sweep Time, 5-11  
  Internal Amplitude Modulation  
  Sweep Trigger, 5-12  
  Internal Amplitude Modulation  
  Waveform, 5-10  
  Wideband Amplitude Modulation  
  State, 5-7  
angle brackets, 2-20

annunciators  
  EXT1 HI, 5-13  
  EXT1 LO, 5-13  
  EXT2 HI, 5-13  
  EXT2 LO, 5-13  
assign HP-IB address to variable, 3-9

### B

baseband synthesizer unlocked, 3-28  
boolean parameter, 2-34

### C

C0, C1, C2, C3, C28 mnemonic, 1-3  
cable  
  HP-IB, 1-2  
cable length  
  HP-IB, 1-2  
CALibration subsystem  
  DCFM/DC(phi)M calibration com-  
  mand, 5-16  
carrier  
  frequency, setting, 3-14, 3-16, 3-21  
  output power level, setting, 3-14  
  power, setting, 3-16  
characters  
  white space, 2-24  
check  
  function, 3-28  
checking for conditions, 3-26  
clear  
  display, 1-5, 3-4, 3-6, 3-9  
  output, 1-5, 3-4, 3-6  
  status byte register, 3-18  
clear command, 2-15  
cold  
  oven, 3-28  
command, 2-19  
  commented, 3-10  
  examples, 2-20  
  query, 3-8  
  root level, 2-23  
  types, 2-22  
  wait, 3-19  
command error, 6-8  
command mapping, 4-27  
command statement  
  abort, 2-12  
  clear, 2-15  
  enter, 2-17  
  local, 2-14  
  local lockout, 2-14

  output, 2-16  
  remote, 2-13  
command syntax  
  description, 5-2  
command tree, 2-23  
  levels, 2-23  
  paths, 2-23  
commands  
  common, 2-22, 2-24  
  event, 2-25  
  implied, 2-25  
  query, 2-25  
  subsystem, 2-22  
commented command, 3-10  
common command syntax, 2-29  
common commands, 2-22, 2-24  
COMMunicate subsystem, 5-17  
  GP-IB Address, 5-17  
  RS-232 Baud Rate, 5-17  
  RS-232 Reset, 5-19  
  RS-232 RTS Control, 5-18  
  RS-232 RTS Echo, 5-18  
  RS-232 XON Handshake Receive  
  State, 5-19  
  RS-232 XON Handshake Transmit  
  State, 5-19  
condition register, 2-42  
  data questionable, 2-50  
  data questionable calibration, 2-63  
  data questionable frequency, 2-57  
  data questionable modulation, 2-60  
  data questionable power, 2-54  
  standard operation, 2-47  
connector  
  external, 3-13  
continuous step sweep, 3-17  
controller, 2-11, 2-19  
CW  
  frequency, query value, 3-9  
  power, querying value, 3-10  
  signal, 3-11

### D

data questionable  
  calibration condition register, 2-63  
  calibration event enable register, 2-64  
  calibration status group, 2-62  
  condition register, 2-50  
  event enable register, 2-52  
  frequency condition register, 2-57  
  frequency event enable register, 2-58  
  frequency event register, 3-29

---

## Index

- frequency status group, 2-56
  - modulation condition register, 2-60
  - modulation event enable register, 2-61
  - modulation event register, 3-29
  - modulation status group, 2-59, 3-29
  - power condition register, 2-54
  - power event enable register, 2-55
  - power event register, 3-29
  - power status group, 2-53, 3-29
  - status group, 2-49
  - data types, 2-31, 2-35
  - DC1 mnemonic, 1-3
  - declaring integer variables, 3-9
  - deviation
    - setting, 3-14
  - device-specific error, 6-20
  - DIAGnostic subsystem
    - Attenuator Cycle Information, 5-21
    - Display Time-On Information, 5-21
    - Instrument Serial Number and Firmware Information, 5-22
    - Instrument Time-On Information, 5-22
    - Option Information, 5-22
    - Power Cycle Information, 5-21
    - Reverse Power Protection Trips Information, 5-21
  - dimension string variables, 3-9, 3-24
  - disabling keys, 3-5
  - discrete
    - parameter, 2-33
    - response data, 2-35
  - discrete function
    - querying value, 3-10
  - display
    - clear, 3-9
    - clearing the, 1-5, 3-4
  - DISPlay subsystem
    - Configure Display Brightness, 5-23
    - Configure Display Contrast, 5-23
    - Configure Display Inverse Video, 5-24
  - DT1 mnemonic, 1-3
  - E**
  - E1 mnemonic, 1-3
  - enable register
    - service request, 2-41
  - end of sweep service request, 3-30
  - endless loop, 3-32
  - enter command, 2-17
  - EOI, 2-28
  - error messages
    - overview, 6-1
  - error messages description, 6-2
  - front panel error queue, 6-2
  - querying the error queue, 6-4
  - SCPI remote interface error queue, 6-2
  - SCPI standard, 6-7
  - error numbers, 6-5
  - error, command, 6-8
  - error, device-specific, 6-20
  - error, execution, 6-13
  - error, query, 6-22
  - errors
    - instrument, 2-27
  - errors, none, 6-6
  - errors, signal generator instrument-specific, 6-24
  - event
    - commands, 2-25
    - register, 2-42
  - event enable register, 2-42
  - data questionable, 2-52
    - calibration, 2-64
  - data questionable frequency, 2-58
  - data questionable modulation, 2-61
  - data questionable power, 2-55
  - standard operation, 2-48
- example program
- end of sweep service request, 3-30
  - generating a CW signal, 3-11
  - generating a step-swept signal, 3-17
  - generating an AC-coupled external FM signal, 3-13
  - generating an AC-coupled internal FM signal, 3-15
  - generating an external pulse modulated signal, 3-20
  - local lockout demonstration, 3-5
  - reading the status byte, 3-26
  - saving and recalling states, 3-22
  - using queries, 3-8
- examples
- command, 2-20
  - response, 2-21
- execution error, 6-13
- EXT1 HI annunciator, 5-13
  - EXT1 LO annunciator, 5-13
  - EXT2 HI annunciator, 5-13
  - EXT2 LO annunciator, 5-13
- extended numeric parameter, 2-32
- external
- connector, 3-13
  - input, 3-20
  - pulse modulated signal, 3-20
  - source, 3-14
- F**
- filter
- negative transition, 2-42
  - positive transition, 2-42
- FM
- path, setting, 3-14
  - signal, 3-13, 3-15
  - source deviation, setting, 3-16
  - source rate
    - setting, 3-16
- FM subsystem, 5-25
- external frequency modulation source coupling command, 5-25
  - frequency modulation deviation command, 5-31
  - frequency modulation deviation coupling command, 5-32
  - frequency modulation source command, 5-29
  - frequency modulation state command, 5-30
  - Internal Frequency Modulation Alternate Frequency, 5-26
  - Internal Frequency Modulation Alternate Frequency Amplitude, 5-27
  - internal frequency modulation source rate command, 5-25
  - Internal Frequency Modulation Sweep Time, 5-28
  - Internal Frequency Modulation Sweep Trigger, 5-29
  - Internal Frequency Modulation Waveform, 5-27
- forgiving listening, 2-21, 2-31
- frequency
- mode, setting, 3-18
  - modulation, setting, 3-14, 3-16
  - setting, 3-12, 3-14, 3-16
  - steps, setting, 3-18
- FREQuency subsystem, 5-33
- continuous wave frequency command, 5-38
  - fixed frequency command, 5-33
  - frequency mode command, 5-33

---

## Index

- frequency multiplier command, 5-34
  - frequency offset command, 5-34
  - frequency optimization command, 5-38
  - frequency reference command, 5-35
  - frequency reference state command, 5-36
  - start frequency command, 5-37
  - stop frequency command, 5-37
- G**
- generating
    - AC-coupled external FM signal, 3-13
    - AC-coupled internal FM signal, 3-15
    - CW signal, 3-11
    - external pulse modulated signal, 3-20
    - step-swept signal, 3-17
  - GP-IB Address, 5-17
  - group
    - standard event status, 2-43
    - status byte, 2-39
- H**
- Hewlett-Packard Interface Bus, 1-2
  - HP, 1-2
  - HP 8656/57-compatible commands, 4-25
  - HP Basic, 2-2
  - HP-IB, 1-2
    - address, 1-4, 3-2, 3-3
    - querying address, 3-10
    - select code, 3-2
    - verification, 1-5
  - HP-IB address, 1-2
  - HP-IB cable, 1-2
  - HP-IB cable length, 1-2
  - HP-IB command statements, 2-12
  - HP-IB connector, 1-3
- I**
- IAM subsystem
    - nternal Amplitude Modulation Alternate Frequency Amplitude, 5-9
  - IEEE 488.2 Common Commands
    - \*CLS clear status command, 5-3
    - \*ESE standard event status enable command, 5-3
    - \*ESR? standard event status register query, 5-3
    - \*IDN? identification query, 5-3
    - \*OPC? operation complete command, 5-3
    - \*OPC? operation complete query, 5-4
    - \*RCL recall command, 5-4
    - \*RST reset command, 5-4
    - \*SAV save command, 5-4
    - \*SRE service request enable command, 5-4
    - \*SRE? service request enable query, 5-4
    - \*STB? read status byte query, 5-4
    - \*TRG trigger command, 5-4
    - \*TST? self-test command, 5-5
    - \*WAI wait-to-continue command, 5-5
  - IEEE 488.2 common commands, 5-3
  - IEEE mnemonics, 1-3
  - IEEE standard
    - 488.1, 1-2
    - 488.2, 1-2
  - IEEE standard 488-1978, 1-3
  - implied commands, 2-25
  - instrument, 2-19
  - instrument address, 1-4
  - instrument errors, 2-27
  - instrument-specific errors, 6-24
  - integer response data, 2-35
  - interconnecting cable, 1-2
  - interface bus, 1-2
  - Internal, 5-9
  - internal
    - FM source deviation, setting, 3-16
    - FM source rate, setting, 3-16
  - interrupt, 3-33
- K**
- key versus command table, 4-2
  - keys
    - disabled, 3-5
- L**
- L3 mnemonic, 1-3
  - LEO mnemonic, 1-3
  - leveling
    - setting, 3-21
  - LFOutput subsystem, 5-41
    - low frequency output amplitude command, 5-41
    - low frequency output frequency command, 5-41, 5-42
    - low frequency output source command, 5-45
  - low frequency output state command, 5-46
  - low frequency output waveform command, 5-44
- LIST** subsystem, 5-47
- dwll list command, 5-47
  - dwll list points query command, 5-48
  - dwll list type command, 5-48
  - frequency list command, 5-48
  - frequency list points query command, 5-49
  - list direction command, 5-47
  - list mode command, 5-50
  - list trigger source command, 5-51
  - list type command, 5-51
  - manual list command, 5-49
  - power list command, 5-50
  - power list points query command, 5-50
- list type
- setting, 3-18
- listener, 2-11
- local
- key, 3-5
  - mode, 1-5, 3-4
- local command, 2-14
- local lockout
- demonstration, 3-5
  - mode, 3-6
- local lockout command, 2-14
- local mode, 3-6
- loop
- endless, 3-32
- M**
- MEMory subsystem, 5-53
    - all memory catalog command, 5-54
    - binary memory catalog command, 5-53
    - data memory command, 5-55
    - data memory query command, 5-55, 5-56, 5-57
    - delete all command, 5-58
    - delete filename command, 5-58, 5-59
    - free memory query command, 5-59, 5-60
    - state memory catalog command, 5-54
  - messages
    - program, 2-21
    - response, 2-21
  - mnemonics, 1-3, 2-20
  - mode
-



---

## Index

- Local, 1-5, 3-4, 3-6
  - Local Lockout, 3-6
  - Remote, 1-5, 3-4, 3-6
  - modulation
    - setting, 3-14, 3-16, 3-21
  - modulation state
    - querying, 3-10
  - multiple parameters, 2-24
- N**
- negative transition filter, 2-42
  - numeric parameter, 2-31
- O**
- optional parameters, 2-25
  - output
    - setting, 3-14, 3-16, 3-18
  - output command, 2-16
  - OUTPut subsystem, 5-61
    - RF output circuit protection clear command, 5-61
    - RF output circuit protection mode command, 5-62
    - RF output circuit protection query, 5-62
    - RF output circuit protection query command, 5-62
    - RF output modulation state command, 5-61
    - RF output state command, 5-63
  - oven cold, 3-28
  - overmodulation, 3-28
- P**
- parameter types, 2-31
    - boolean, 2-34
    - discrete, 2-33
    - extended numeric, 2-32
    - numeric, 2-31
  - parameters
    - optional, 2-25
    - separating, 2-24
  - PM subsystem, 5-64
    - external phase modulation source coupling command, 5-64
    - internal phase modulation source rate command, 5-65, 5-66
    - internal phase modulation waveform command, 5-44, 5-45, 5-66, 5-67
    - phase modulation deviation command, 5-69
    - phase modulation deviation coupling command, 5-70
    - phase modulation source command, 5-68
    - phase modulation state command, 5-68
  - positive transition filter, 2-42
  - power level
    - setting, 3-12, 3-14, 3-16, 3-21
  - power leveling
    - setting, 3-21
  - POWER subsystem, 5-71
    - RF output ALC search state command, 5-71
    - RF output ALC state command, 5-72
    - RF output automatic level attenuation command, 5-72
    - RF output level amplitude offset command, 5-75
    - RF output level immediate amplitude command, 5-76
    - RF output power mode command, 5-73
    - RF output reference power command, 5-73
    - RF output reference power state command, 5-74
    - RF output start power command, 5-74
    - RF output stop power command, 5-75
  - PP0 mnemonic, 1-3
  - precise talking, 2-21, 2-31
  - print to display, 1-5, 3-4, 3-6
  - program message, 2-19, 2-21
  - programming, 2-2
    - cross reference, 4-1
  - programming language
    - HP Basic, 2-2
  - PULM subsystem, 5-77
    - internal pulse modulation rate command, 5-43, 5-77, 5-78, 5-79
    - pulse modulation source command, 5-77
    - state command, 5-78
  - pulse modulation, 3-20
    - setting, 3-21
- Q**
- query, 2-19, 2-30, 3-8
    - commands, 2-25
    - response, 3-9
    - values, 3-9
  - query error, 6-22
- R**
- reading instrument errors, 2-27
  - reading status byte, 3-26
  - real response data, 2-35
  - recalling
    - register contents, 3-24
    - states, 3-22
  - register
    - clearing, 3-18
    - condition, 2-42
    - data questionable
      - calibration condition, 2-63
    - data questionable calibration event enable, 2-64
    - data questionable condition, 2-50
    - data questionable event enable, 2-52
    - data questionable frequency condition, 2-57
    - data questionable frequency event enable, 2-58
    - data questionable modulation condition, 2-60
    - data questionable modulation event enable, 2-61
    - data questionable power condition, 2-54
    - data questionable power event enable, 2-55
    - event, 2-42
    - event enable, 2-42
    - recalling contents, 3-24
    - resetting, 3-28
    - service request enable, 2-41
    - standard event status, 2-44
    - standard event status enable, 2-45
    - standard operation condition, 2-47
    - standard operation event enable, 2-48
    - status byte, 2-40
  - register system
    - status byte, 2-38
  - remote
    - annunciator, 3-3
    - mode, 1-5, 3-4, 3-6
  - remote command, 2-13
  - remote operation
    - setup, 1-4
  - remote programming
    - verification, 1-5
  - reset
    - parser, 1-5, 3-4, 3-6
    - status byte register, 3-28
-

---

## Index

- response
  - examples, 2-21
  - message, 2-19, 2-21
  - message syntax, 2-30
- response data
  - discrete, 2-35
  - formats, 3-8
  - integer, 2-35
  - real, 2-35
  - string, 2-36
  - types, 2-35
- responses
  - using query, 3-9
- RF output
  - setting, 3-12, 3-14, 3-16, 3-18, 3-21
- RL1 mnemonic, 1-3
- root level command, 2-23
- ROSCillator subsystem
  - reference oscillator source query command, 5-39
- S**
- saving states, 3-22
- SCPI, 2-19
  - angle brackets, 2-20
  - data types, 2-31
  - examples, 2-20
  - response syntax, 2-28
- SCPI command
  - mnemonics, 2-20
  - syntax, 2-28
  - types, 2-23
- SCPI terms
  - command, 2-19
  - controller, 2-19
  - instrument, 2-19
  - program message, 2-19
  - query, 2-19
  - response message, 2-19
- select code, 3-2
- separating parameters, 2-24
- service request, 3-30, 3-32
- service request enable register, 2-41
- set
  - ac coupling, 3-14
  - carrier frequency, 3-14, 3-16, 3-21
  - carrier output power level, 3-14
  - carrier power, 3-16
  - deviation, 3-14
  - FM path, 3-14
  - frequency, 3-12, 3-14
  - frequency mode, 3-18
  - frequency modulation, 3-14, 3-16
  - frequency steps, 3-18
  - internal FM source deviation, 3-16
  - internal FM source rate, 3-16
  - leveling, 3-21
  - list type, 3-18
  - modulation, 3-14, 3-21
  - output, 3-14, 3-18
  - power level, 3-12, 3-14, 3-21
  - power leveling, 3-21
  - pulse modulation, 3-21
  - RF output, 3-12, 3-14, 3-16, 3-18, 3-21
  - source deviation, 3-16
  - source rate, 3-16
  - start/stop frequency, 3-18
  - steps, 3-18
  - sweep mode, 3-18
- setup for remote operation, 1-4
- SH1 mnemonic, 1-3
- signal generator
  - identity, 3-10
  - model, 3-10
  - options, 3-10
- signal generator instrument-specific errors, 6-24
- signal generator programming, 2-2
- source
  - deviation, setting, 3-16
  - external, 3-14
  - rate, setting, 3-16
- SR1 mnemonic, 1-3
- SRQ, 3-32
- standard event status
  - enable register, 2-45
  - group, 2-43
  - register, 2-44
- standard operation
  - condition register, 2-47
  - event enable register, 2-48
  - status group, 2-46
- start/stop frequency
  - setting, 3-18
- states
  - recalling, 3-22
  - saving, 3-22
- status byte
  - group, 2-39
  - reading, 3-26
  - register, 2-40
  - register system, 2-37, 2-38
  - register, clearing, 3-18
- resetting the register, 3-28
- status enable register
  - standard event, 2-45
- status group
  - data questionable, 2-49
  - data questionable calibration, 2-62
  - data questionable frequency, 2-56
  - data questionable modulation, 2-59
  - data questionable power, 2-53
  - standard event, 2-43
  - standard operation, 2-46
- status register
  - standard event, 2-44
- STATus subsystem, 5-81
  - data questionable calibration status group condition register query command, 5-83
  - data questionable calibration status group event register query command, 5-85
  - data questionable calibration status group summary bit enable command, 5-84
  - data questionable calibration status NTR filter bit enable command, 5-84
  - data questionable calibration status PTR filter bit enable command, 5-84
  - data questionable condition summary bit query command, 5-85
  - data questionable frequency status group condition register query command, 5-85
  - data questionable frequency status group event register query command, 5-87
  - data questionable frequency status group summary bit enable command, 5-86
  - data questionable frequency status NTR filter bit enable command, 5-86
  - data questionable frequency status PTR filter bit enable command, 5-86
  - data questionable modulation status group condition register query command, 5-87
  - data questionable modulation status group event register query command, 5-88

---

## Index

- data questionable modulation status
    - group summary bit enable command, 5-87
  - data questionable modulation status NTR filter bit enable command, 5-87
  - data questionable modulation status PTR filter bit enable command, 5-88
  - data questionable power status group
    - condition register query command, 5-88
  - data questionable power status group event register query command, 5-90
  - data questionable power status group summary bit enable command, 5-89
  - data questionable power status NTR filter bit enable command, 5-89
  - data questionable power status PTR filter bit enable command, 5-89
  - data questionable status event register query command, 5-83
  - data questionable status group summary bit enable command, 5-85
  - data questionable status NTR filter bit enable command, 5-82
  - data questionable status PTR filter bit enable command, 5-83
  - standard operation status group condition register query command, 5-81
  - standard operation status group event register query command, 5-82
  - standard operation status group NTR filter bit enable command, 5-81
  - standard operation status group PTR filter bit enable command, 5-82
  - standard operation status group summary bit enable command, 5-81
  - status preset command, 5-82
- step
- sweep, 3-17
  - swept signal, 3-17
- steps
- setting, 3-18
- string response data, 2-36
- string variable, 3-9
- dimensioning a, 3-24
  - entering response, 3-10
- subsystem commands, 2-22, 5-6
- sweep
- mode, setting, 3-18
  - step, 3-17
  - trigger, 3-32
- SWEep subsystem, 5-91
- sweep dwell command, 5-91
  - sweep points command, 5-91
- syntax
- command, 2-28
  - common command, 2-29
  - drawings, 2-12
  - response, 2-28
  - response message, 2-30
- synthesizer unlocked, 3-28
- SYSTem subsystem, 5-93
- error information query command, 5-93
  - help mode command, 5-93
  - power on/preset conditions command, 5-94
  - preset language command, 5-95
  - preset type command, 5-95
  - remote language command, 5-94
  - screen saver delay command, 5-96
  - screen saver mode command, 5-96
  - screen saver state command, 5-97
  - system preset command, 5-95

### T

- T6 mnemonic, 1-3
- talker, 2-11
- TE0 mnemonic, 1-3
- terminators, 2-23
- TRIGger subsystem, 5-98
  - continuous sweep command, 5-98
  - external trigger on slope command, 5-100
  - single sweep command, 5-99
  - trigger output polarity command, 5-98, 5-99
  - trigger source command, 5-100
- trigger sweep, 3-32

### U

- undermodulation, 3-26, 3-28
- unleveled output, 3-26
- unlocked
  - 1 GHz reference, 3-28
  - 10 MHz reference, 3-28
  - baseband synthesizer, 3-28
  - synthesizer, 3-28
- using queries, 3-8

### V

- verification
  - HP-IB, 1-5
  - of remote programming, 1-5
  - program, 1-5

### W

- wait, 3-19
- white space characters, 2-24
- Wideband, 5-7