SCPI Command Reference

Agilent Technologies
E8257D/67D PSG Signal Generators

This guide applies to the following signal generator models:

E8257D PSG Analog Signal Generator

E8267D PSG Vector Signal Generator

Due to our continuing efforts to improve our products through firmware and hardware revisions, signal generator design and operation may vary from descriptions in this guide. We recommend that you use the latest revision of this guide to ensure you have up-to-date product information. Compare the print date of this guide (see bottom of page) with the latest revision, which can be downloaded from the following website:

http://www.agilent.com/find/psg

Agilent Technologies

Manufacturing Part Number: E8251-90356

Printed in USA

January 2005

Notice

The material in this document is provided "as is," and is subject to change without notice in future editions.

Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied with regard to this manual and to any of the Agilent products to which it pertains, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or any of the Agilent products to which it pertains. Should Agilent have a written contract with the User and should any of the contract terms conflict with these terms, the contract terms shall control.

Questions or Comments about our Documentation?

We welcome any questions or comments you may have about our documentation. Please send us an E-mail at **sources_manuals@am.exch.agilent.com**.

Contents

Contents

Contents

1 Using this Guide

In the following sections, this chapter describes how SCPI information is organized and presented in this guide. An overview of the SCPI language is also provided:

- "How the SCPI Information is Organized" on page 2
- "SCPI Basics" on page 3

How the SCPI Information is Organized

SCPI Listings

The table of contents lists the Standard Commands for Programmable Instruments (SCPI) without the parameters. The SCPI subsystem name will generally have the first part of the command in parenthesis that is repeated in all commands within the subsystem. The title(s) beneath the subsystem name is the remaining command syntax. The following example demonstrates this listing:

Communication Subsystem (:SYSTem:COMMunicate)
    :PMETer:CHANnel
    :SERial:ECHO

The following examples show the complete commands from the above Table of Contents listing:

    :SYSTem:COMMunicate:PMETer:CHANnel
    :SYSTem:COMMunicate:SERial:ECHO

Subsystem Groupings by Chapter

A subsystem is a group of commands used to configure and operate a certain function or feature. Like individual commands, subsystems that share a similar scope or role can also be categorized and grouped together. This guide uses chapters to divide subsystems into the following groups:

- System Commands
- Basic Function Commands
- Analog Modulation Commands
- Digital Modulation Commands

Front Panel Operation Cross Reference

The last section in this book provides an index of hardkeys, softkeys, and data fields used in front panel operation, cross-referenced to their corresponding SCPI command. Key and data field names are sorted in two ways:

- individual softkey, hardkey, or data field name
- SCPI subsystem name with associated key and data field names nested underneath

Supported Models and Options per Command

Within each command section, the Supported heading describes the signal generator configurations supported by the SCPI command. “All” means that all models and options are supported. When “All with Option xxx” is shown next to this heading, only the stated option(s) is supported.

SCPI Basics

This section describes the general use of the SCPI language for the PSG. It is not intended to teach you everything about the SCPI language; the SCPI Consortium or IEEE can provide that level of detailed information. For a list of the specific commands available for the signal generator, refer to the table of contents.

For additional information, refer to the following publications:

- IEEE Standard 488.1-1987, IEEE Standard Digital Interface for Programmable Instrumentation. New York, NY, 1998.
- IEEE Standard 488.2-1992, IEEE Standard Codes, Formats, Protocols and Command Commands for Use with ANSI/IEEE Standard 488.1-1987. New York, NY, 1998.

Common Terms

The following terms are used throughout the remainder of this section:

Command	A command is an instruction in SCPI consisting of mnemonics (keywords), parameters (arguments), and punctuation. You combine commands to form messages that control instruments.
Controller	A controller is any device used to control the signal generator, for example a computer or another instrument.
Event Command	Some commands are events and cannot be queried. An event has no corresponding setting; it initiates an action at a particular time.
Program Message	A program message is a combination of one or more properly formatted commands. Program messages are sent by the controller to the signal generator.
Query	A query is a special type of command used to instruct the signal generator to make response data available to the controller. A query ends with a question mark. Generally you can query any command value that you set.
Response Message	A response message is a collection of data in specific SCPI formats sent from the signal generator to the controller. Response messages tell the controller about the internal state of the signal generator.

Command Syntax

A typical command is made up of keywords prefixed with colons (:). The keywords are followed by parameters. The following is an example syntax statement:

`[:SOURce]:POWer[:LEVel] MAXimum|MINimum`

In the example above, the `[:LEVel]` portion of the command immediately follows the `:POWer` portion with no separating space. The portion following the `[:LEVel]`, `MINimum|MAXimum`, are the parameters (argument for the command statement). There is a separating space (white space) between the command and its parameter.

Additional conventions in syntax statements are shown in Table 1-1 and Table 1-2.

Table 1-1 Special Characters in Command Syntax

<table>
<tr><th>Characters</th><th>Meaning</th><th>Example</th></tr>
<tr><td>|</td><td>A vertical stroke between keywords or parameters indicates alterative choices. For parameters, the effect of the command varies depending on the choice.</td><td>[:SOURce]:AM:
MOD DEEP|NORMal

DEEP or NORMal are the choices.</td></tr>
<tr><td>[ ]</td><td>Square brackets indicate that the enclosed keywords or parameters are optional when composing the command. These implied keywords or parameters will be executed even if they are omitted.</td><td>[:SOURce]:FREQuency[:CW]?

SOURce and CW are optional items.</td></tr>
<tr><td>< ></td><td>Angle brackets around a word (or words) indicate they are not to be used literally in the command. They represent the needed item.</td><td>[:SOURce]:FREQuency:
STARt <val><unit>

In this command, the words <val> and <unit> should be replaced by the actual frequency and unit.

:FREQuency:STARt 2.5GHZ</td></tr>
<tr><td>{ }</td><td>Braces indicate that parameters can optionally be used in the command once, several times, or not at all.</td><td>[:SOURce]:LIST:
POWer <val>{,<val>}

a single power listing:
LIST:POWer 5
a series of power listings:
LIST:POWer 5,10,15,20</td></tr>
</table>

Table 1-2 Command Syntax

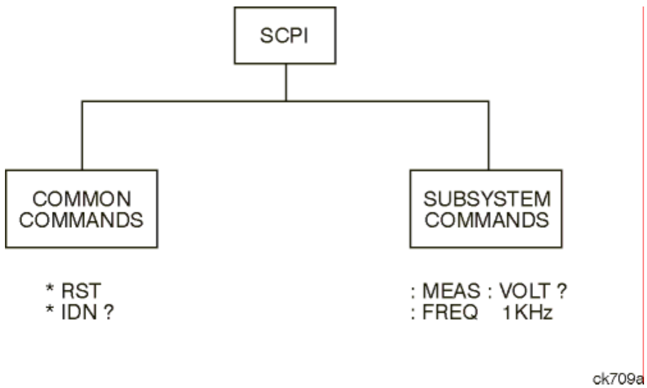
Characters, Keywords, and Syntax	Example
Upper-case lettering indicates the minimum set of characters required to execute the command.	`[:SOURce]:FREQuency[:CW]?`, `FREQ` is the minimum requirement.
Lower-case lettering indicates the portion of the command that is optional; it can either be included with the upper-case portion of the command or omitted. This is the flexible format principle called forgiving listening. Refer to "Command Parameters and Responses" on page 7 for more information.	`:FREQuency` Either `:FREQ`, `:FREQuency`, or `:FREQUENCY` is correct.
When a colon is placed between two command mnemonics, it moves the current path down one level in the command tree. Refer to "Command Tree" on page 6 more information on command paths.	`:TRIGger:OUTPut:POLarity?` `TRIGger` is the root level keyword for this command.
If a command requires more than one parameter, you must separate adjacent parameters using a comma. Parameters are not part of the command path, so commas do not affect the path level.	`[:SOURce]:LIST:` `DWELl <val>{,<val>}`
A semicolon separates two commands in the same program message without changing the current path.	`:FREQ 2.5GHZ;:POW 10DBM`
White space characters, such as `<tab>` and `<space>`, are generally ignored as long as they do not occur within or between keywords. However, you must use white space to separate the command from the parameter, but this does not affect the current path.	`:FREQ uency` or `:POWer :LEVel` are not allowed. A `<space>` between `:LEVel` and `6.2` is mandatory. `:POWer:LEVel 6.2`

Command Types

Commands can be separated into two groups: common commands and subsystem commands. Figure 1-1, shows the separation of the two command groups. Common commands are used to manage macros, status registers, synchronization, and data storage and are defined by IEEE 488.2. They are easy to recognize because they all begin with an asterisk. For example `*IDN?`, `*OPC`, and `*RST` are common commands. Common commands are not part of any subsystem and the signal generator interprets them in the same way, regardless of the current path setting.

Subsystem commands are distinguished by the colon (`:`). The colon is used at the beginning of a command statement and between keywords, as in `:FREQuency[:CW?]`. Each command subsystem is a set of commands that roughly correspond to a functional block inside the signal generator. For example, the power subsystem (`:POWer`) contains commands for power generation, while the status subsystem (:STATus) contains commands for controlling status registers.

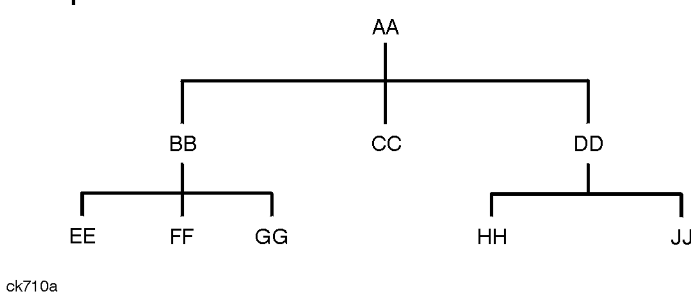
Figure 1-1 Command Types



Command Tree

Most programming tasks involve subsystem commands. SCPI uses a structure for subsystem commands similar to the file systems on most computers. In SCPI, this command structure is called a command tree and is shown in Figure 1-2.

Figure 1-2 Simplified Command Tree



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 commands. In the following example, `:POWer` represents AA, `:ALC` represents BB, `:SOURce` represents GG. The complete command path is `:POWer:ALC:SOURce?` (:AA:BB:GG).

Paths Through the Command Tree

To access commands from different paths in the command tree, you must understand how the signal generator interprets commands. The parser, a part of the signal generator firmware, decodes each message sent to the signal generator. 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) and where it expects to find the next command statement. This is important because the same keyword may appear in different paths. The particular path is determined by the keyword(s) in the command statement.

A message terminator, such as a `<new line>` character, sets the current path to the root. Many programming languages have output statements that automatically send message terminators.

NOTE The current path is set to the root after the line-power is cycled or when `*RST` is sent.

Command Parameters and Responses

SCPI defines different data formats for use in program and response messages. It does this to accommodate the principle of forgiving listening and precise talking. For more information on program data types refer to IEEE 488.2. Forgiving listening means the command and parameter formats are flexible.

For example, with the `:FREQuency:REFerence:STATe ON|OFF|1|0` command, the signal generator accepts `:FREQuency:REFerence:STATe ON`, `:FREQuency:REFerence:STATe 1`, `:FREQ:REF:STAT ON`, `:FREQ:REF:STAT 1` to turn on the frequency reference mode.

Each parameter type has one or more corresponding response data types. A setting that you program using a numeric parameter returns either real or integer response data when queried. Response data (data returned to the controller) is more concise and restricted and is called precise talking.

Precise talking means that the response format for a particular query is always the same.

For example, if you query the power state (`:POWer:ALC:STATe?`) when it is on, the response is always 1, regardless of whether you previously sent `:POWer:ALC:STATe 1` or `:POWer:ALC:STATe ON`.

Table 1-3 Parameter and Response Types

Parameter Types	Response Data Types
Numeric	Real, Integer
Extended Numeric	Real, Integer
Discrete	Discrete
Boolean	Numeric Boolean
String	String

Numeric Parameters

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

If a signal generator setting is programmed with a numeric parameter which can only assume a finite value, it automatically rounds any entered parameter which is greater or less than the finite value. For example, if a signal generator has a programmable output impedance of 50 or 75 ohms, and you specified 76.1 for the output impedance, the value is rounded to 75. The following are 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 the E in exponential
–7.89E–001	use either E or e in exponential
+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.

The following are examples of extended numeric parameters:

Extended Numeric Parameters		**Special Parameters**	
100	any simple numeric value	DEFault	resets parameter to its default value
1.2GHZ	GHZ can be used for exponential (E009)	UP	increments the parameter
200MHZ	MHZ can be used for exponential (E006)	DOWN	decrements the parameter
–100mV	negative 100 millivolts	MINimum	sets parameter to smallest possible value
10DEG	10 degrees	MAXimum	sets parameter to largest possible value

Discrete Parameters

Discrete parameters use mnemonics to represent each valid setting. They have a long and a short form, just like command mnemonics. You can mix upper and lower case letters for discrete parameters.

The following examples of discrete parameters are used with the command
`:TRIGger[:SEQuence]:SOURce BUS|IMMediate|EXTernal`.

BUS	GPIB, LAN, or RS-232 triggering
IMMediate	immediate trigger (free run)
EXTernal	external triggering

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

The following are examples of discrete parameters in commands:

```
TRIGger:SOURce BUS
TRIGger:SOURce IMMediate
TRIGger:SOURce EXTernal
```

Boolean Parameters

Boolean parameters represent a single binary condition that is either true or false. The two-state boolean parameter has four arguments. The following list shows the arguments for the two-state boolean parameter:

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

String Parameters

String parameters allow ASCII strings to be sent as parameters. Single or double quotes are used as delimiters.

The following are examples of string parameters:

```
'This is valid'          "This is also valid"          'SO IS THIS'
```

Real Response Data

Real response data represent decimal numbers in either fixed decimal or scientific notation. Most high-level programming languages that support signal generator input/output (I/O) handle either decimal or scientific notation transparently.

The following are examples of real response data:

+4.000000E+010, –9.990000E+002

–9.990000E+002

+4.0000000000000E+010

+1

0

Integer Response Data

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

0	signs are optional	–100	leading – allowed
+100	leading + 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 only returns the short form of a particular mnemonic, in all upper case letters. The following are examples of discrete response data:

IMM   EXT   INT   NEG

Numeric Boolean Response Data

Boolean response data returns a binary numeric value of one or zero.

String Response Data

String response data are similar to string parameters. The main difference is that string response data returns double quotes, 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. The following are examples of string response data:

```
"This is a string"

"one double quote inside brackets: [""]"

"Hello!"
```

Program Messages

The following commands will be used to demonstrate the creation of program messages:

```
[:SOURce]:FREQuency:STARt      [:SOURce]:FREQuency:STOP
[:SOURce]:FREQuency[:CW]       [:SOURce]:POWer[:LEVel]:OFFSet
```

Example 1

```
:FREQuency:STARt 500MHZ;STOP 1000MHZ
```

This program message is correct and will not cause errors; `STARt` and `STOP` are at the same path level. It is equivalent to sending the following message:

```
FREQuency:STARt 500MHZ;FREQuency:STOP 1000MHZ
```

Example 2

```
:POWer 10DBM;:OFFSet 5DB
```

This program message will result in an error. The message makes use of the default `POWer[:LEVel]` node (root command). When using a default node, there is no change to the current path position. Since there is no command `OFFSet` at the root level, an error results.

The following example shows the correct syntax for this program message:

```
:POWer 10DBM;:POWer:OFFSet 5DB
```

Example 3

`:POWer:OFFSet 5DB;POWer 10DBM`

This program message results in a command error. The path is dropped one level at each colon. The first half of the message drops the command path to the lower level command `OFFSet`; `POWer` does not exist at this level.

The `POWer 10DBM` command is missing the leading colon and when sent, it causes confusion because the signal generator cannot find `POWer` at the `POWer:OFFSet` level. By adding the leading colon, the current path is reset to the root. The following shows the correct program message:

`:POWer:OFFSet 5DB;:POWer 10DBM`

Example 4

`FREQ 500MHZ;POW 4DBM`

In this example, the keyword short form is used. The program message is correct because it utilizes the default nodes of `:FREQ[:CW]` and `:POW[:LEVel]`. Since default nodes do not affect the current path, it is not necessary to use a leading colon before `FREQ` or `POW`.

File Name Variables

File name variables, such as `"<file name>"`, represent three formats, `"<file name>"`, `"<file name@file type>"`, and `"</user/file type/file name>"`. The following shows the file name syntax for the three formats, but uses `"FLATCAL"` as the file name in place of the variable `"<file name>"`:

Format 1	`"FLATCAL"`
Format 2	`"FLATCAL@USERFLAT"`
Format 3	`"/USER/USERFLAT/FLATCAL"`

Format 2 uses the file type extension (@USERFLAT) as part of the file name syntax. Format 3 uses the directory path which includes the file name and file type. Use Formats 2 and 3 when the command does not specify the file type. This generally occurs in the Memory (`:MEMory`) or Mass Memory (`:MMEMory`) subsystems. The following examples demonstrate a command where Format 1 applies:

Command Syntax with the file name variable	`:MEMory:STORe:LIST "<file name>"`
Command Syntax with the file name	`:MEMory:STORe:LIST "SWEEP_1"`

This command has `:LIST` in the command syntax. This denotes that `"SWEEP_1"` will be saved in the :List file type location as a list type file.

The following examples demonstrate a command where Format 2 applies:

Command Syntax with the file name variable

```
:MMEMory:COPY "<filename>","<filename>"
```

Command Syntax with the file name

```
:MMEMory:COPY "FLATCAL@USERFLAT","FLAT_2CAL@USERFLAT"
```

This command cannot distinguish which file type "FLATCAL" belongs to without the file type extension (@USERFLAT). If this command were executed without the extension, the command would assume the file type was Binary.

The following examples demonstrate a command where format 3 applies:

Command Syntax with the file name variable

```
:MMEMory:DATA "/USER/BBG1/WAVEFORM/<file name>",#ABC
```

Command Syntax with the file name

```
:MMEMory:DATA "/USER/BBG1/WAVEFORM/FLATCAL",#ABC
```

This command gives the directory path name where the file `"FLATCAL"` is stored.

`A` the number of decimal digits to follow in B.

`B` a decimal number specifying the number of data bytes in C.

`C` the binary waveform data.

Refer to Table 2-1 on page 66 for a listing of the file systems and types. The entries under file type are used in the directory path.

ARB Waveform File Directories

ARB waveform files can be saved to the following directories:

- `WFM1:` volatile ARB waveform storage. Files located here can be played by the signal generator's arb player, but are volatile and will be lost on a power cycle. The directory can also be specified as `/USER/BBG1/WAVEFORM.`
- `NVWFM:` non-volatile ARB waveform storage. Files must be moved to the `WFM1:` directory before they can be played by the signal generator's Dual ARB player. The directory can also be specified as `/USER/WAVEFORM.`
- `SEQ:` sequence files are stored here and are non-volatile. The directory can also be specified as `/USER/SEQ.`.

MSUS (Mass Storage Unit Specifier) Variable

The variable `"<msus>"` enables a command to be file type specific when working with user files. Some commands use it as the only command parameter, while others can use it in conjunction with a file name when a command is not file type specific. When used with a file name, it is similar to Format 2 in the "File Name Variables" section on page 12. The difference is the file type specifier (msus) occupies its own variable and is not part of the file name syntax.

The following examples illustrate the usage of the variable `"<msus>"` when it is the only command parameter:

Command Syntax with the msus variable

```
:MMEMory:CATalog? "<msus>"
```

Command Syntax with the file system

```
:MMEMory:CATalog? "LIST:"
```

The variable `"<msus>"` is replaced with `"LIST:"`. When the command is executed, the output displays only the files from the List file system.

The following examples illustrate the usage of the variable `"<file name>" with the variable "<msus>"`:

Command Syntax with the file name and msus variables

```
:MMEMory:DELete[:NAME] "<file name>",["<msus>"]
```

Command Syntax with the file name and file system

```
:MMEMory:DELete:NAME "LIST_1","LIST:"
```

The command from the above example cannot discern which file system LIST_1 belongs to without a file system specifier and will not work without it. When the command is properly executed, LIST_1 is deleted from the List file system.

The following example shows the same command, but using Format 2 from the "File Name Variables" section on page 12:

```
:MMEMory:DELete:NAME "LIST_1@LIST"
```

When a file name is a parameter for a command that is not file system specific, either format (`<file name>","<msus>"` or "`<file name@file system>"`) will work.

Refer to Table 1-1 on page 4 for a listing of special syntax characters.

Quote Usage with SCPI Commands

As a general rule, programming languages require that SCPI commands be enclosed in double quotes as shown in the following example:

```
":FM:EXTernal:IMPedance 600"
```

However, when a string is the parameter for a SCPI command, additional quotes or other delimiters may be required to identify the string. Your programming language may use two sets of double quotes, one set of single quotes, or back slashes with quotes to signify the string parameter. The following examples illustrate these different formats:

`"MEMory:LOAD:LIST ""myfile"""` used in BASIC programming languages

`"MEMory:LOAD:LIST \"myfile\""` used in C, C++, Java, and PERL

`"MEMory:LOAD:LIST 'myfile'"` accepted by most programming languages

Consult your programming language reference manual to determine the correct format.

Binary, Decimal, Hexadecimal, and Octal Formats

Command values may be entered using a binary, decimal, hexadecimal, or octal format. When the binary, hexadecimal, or octal format is used, their values must be preceded with the proper identifier. The decimal format (default format) requires no identifier and the signal generator assumes this format when a numeric value is entered without one. The following list shows the identifiers for the formats that require them:

- #B identifies the number as a binary numeric value (base-2).
- #H identifies the number as a hexadecimal alphanumeric value (base-16).
- #Q identifies the number as a octal alphanumeric value (base-8).

The following are examples of SCPI command values and identifiers for the decimal value 45:

#B101101	binary equivalent
#H2D	hexadecimal equivalent
#Q55	octal equivalent

The following example sets the RF output power to 10 dBm (or the equivalent value for the currently selected power unit, `such as DBUV` or `DBUVEMF`) using the hexadecimal value 000A:

```
:POW #H000A
```

A unit of measure, such as `DBM` or `mV`, will not work with the values when using a format other than decimal.

The following example sets the bluetooth board address to FFBF7 (hexadecimal):

```
:RADio:BLUEtooth:ARB:BDADdr #HFFBF7
```

2 System Commands

In the following sections, this chapter provides SCPI descriptions for subsystems dedicated to peripheral signal generator operations common to all PSG models:

Calibration Subsystem (:CALibration)

:DCFM

Supported All with Option UNT

`:CALibration:DCFM`

This command initiates a DCFM or DCΦM calibration depending on the currently active modulation. This calibration eliminates any dc or modulation offset of the carrier signal.

Use this calibration for externally applied signals. While the calibration can also be performed for internally generated signals, dc offset is not a normal characteristic for them.

NOTE If the calibration is performed with a dc signal applied, any deviation provided by the dc signal will be removed and the new zero reference point will be at the applied dc level. The calibration will have to be performed again when the dc signal is removed in order to reset the carrier signal to the correct zero reference.

Key Entry **DCFM / DCΦM Cal**

:IQ

Supported E8267D

`:CALibration:IQ`

This command initiates an I/Q calibration for a range of frequencies and is equivalent to selecting User from the front panel **Calibration Type DC User Full** softkey in the I/Q Calibration menu. For setting range frequencies, refer to ":IQ:STARt" on page 20, and ":IQ:STOP" on page 20.

Key Entry **Execute Cal** **Calibration Type DC User Full**

:IQ:DC

Supported E8267D

`:CALibration:IQ:DC`

This command starts and performs a one– to two–second adjustment that is not traceable to a standard. However, it will minimize errors associated with signal generator internal voltage offsets. This adjustment minimizes errors for the current signal generator setting and at a single frequency. The DC adjustment is volatile and must be repeated with each signal generator setting change. This command can be sent while the RF On/Off is set to Off and the adjustment will still be valid when the RF is enabled.

The I/Q DC adjustment is dependent upon a number of instrument settings. If any of the instrument settings change, the adjustment will become invalid. The dependent instrument settings are:

- RF frequency
- I/Q attenuation level
- Baseband generator settings
- I/Q polarity settings
- Baseband filter settings
- Path settings (Internal I/Q Mux Path 1 or Path 2)
- I/Q calibration (the I/Q DC calibration will be invalidated if any other I/Q calibration is execute)
- Temperature (±5 degrees)

The following instrument states will not invalidate the I/Q DC calibration:

- Power level changes
- I/Q Impairments

***RST** N/A

Key Entry **Execute Cal** **Calibration Type DC User Full**

:IQ:DEFault

Supported E8267D

`:CALibration:IQ:DEFault`

This command will restore the original factory calibration data for the internal I/Q modulator.

Key Entry **Revert to Default Cal Settings**

:IQ:FULL

Supported E8267D

`:CALibration:IQ:FULL`

This command sets and performs a full-frequency range (regardless of the start and stop frequency settings) I/Q calibration and stores the results in the signal generator’s memory.

Start and stop frequencies default to the full frequency range of the signal generator.

Range Depends on the signal generator’s frequency option
Refer to “:FREQuency:CENTer” on page 123

Key Entry **Execute Cal** (**Calibration Type DC User Full** set to **Full**)

:IQ:STARt

Supported E8267D

```
:CALibration:IQ:STARt <val><units>
:CALibration:IQ:STARt?
```

This command sets the start frequency and automatically sets the calibration type to User for an I/Q calibration.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:CAL:IQ:STAR 1GHZ
```

The preceding example sets the signal generator’s start frequency for an IQ calibration to 1 GHz.

Range Depends on the signal generator’s frequency option
Refer to “:FREQuency:CENTer” on page 123

Key Entry **Start Frequency**

:IQ:STOP

Supported E8267D

```
:CALibration:IQ:STOP <val><units>
:CALibration:IQ:STOP?
```

This command sets the stop frequency and automatically sets the calibration type to User for an I/Q calibration. The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:CAL:IQ:STOP 2GHZ
```

The preceding example sets the signal generator’s stop frequency for an IQ calibration to 2 GHz.

Range Depends on the signal generator’s frequency option
Refer to “:FREQuency:CENTer” on page 123

Key Entry **Stop Frequency**

:WBIQ

Supported E8267D with Option 015

```
:CALibration:WBIQ
```

This command initiates a wideband I/Q calibration for a range of frequencies and is equivalent to selecting User from the front panel **Calibration Type DC User Full** softkey. For setting range frequencies, refer to ":WBIQ:STARt" on page 22, and ":WBIQ:STOP" on page 23command descriptions.

Key Entry **Execute Cal**

:WBIQ:DC

Supported E8267D with Option 015

```
:CALibration:WBIQ:DC
```

This command performs a one to two second adjustment that is not traceable to a standard. However, it will minimize errors associated with offset voltages. This adjustment minimizes errors for the current signal generator setting and at a single frequency. The DC adjustment is volatile and must be repeated with each signal generator setting change. This command can be sent while the RF On/Off is set to Off and the adjustment will be valid when RF is enabled.

The wideband I/Q DC adjustment is dependent upon a number of instrument settings. If any of the PSG settings change, the adjustment will become invalid. The dependent instrument settings are:

- RF frequency
- I/Q attenuation level
- Baseband generator settings
- I/Q polarity settings
- Baseband filter settings
- Path settings (Internal I/Q Mux Path 1 or Path 2)
- I/Q calibration (the I/Q DC calibration will be invalidated if any other I/Q calibration is executed)
- Temperature (±5 degrees)

The following instrument states will not invalidate the I/Q DC calibration:

- Power level changes
- I/Q Impairments

***RST** N/A

Key Entry **Execute Cal** **Calibration Type DC User Full**

:WBIQ:DEFault

Supported E8267D with Option 015

```
:CALibration:WBIQ:DEFault
```

This command will restore the original factory calibration data for the internal I/Q modulator.

Key Entry **Revert to Default Cal Settings**

:WBIQ:FULL

Supported E8267D with Option 015

```
:CALibration:WBIQ:FULL
```

This command sets and performs a full-frequency range (regardless of the start and stop frequency settings) wideband I/Q calibration and stores the results in the signal generator's firmware.

Start and stop frequencies will default to the full frequency range of the signal generator.

Range Depends on the signal generator's frequency option
Refer to ":FREQuency:CENTer" on page 123

Key Entry **Execute Cal** **Calibration Type DC User Full**

:WBIQ:STARt

Supported E8267D with Option 015

```
:CALibration:WBIQ:STARt <val><units>
:CALibration:WBIQ:STARt?
```

This command sets the start frequency and automatically sets the calibration type to User for a wideband I/Q calibration. The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:CAL:WBIQ:STAR 1GHZ
```

The preceding example sets the signal generator's start frequency to 1 GHz for a wideband IQ calibration.

Range Depends on the signal generator's frequency option
Refer to ":FREQuency:CENTer" on page 123

Key Entry **Start Frequency**

:WBIQ:STOP

Supported E8267D with Option 015

```
:CALibration:WBIQ:STOP <val><units>
:CALibration:WBIQ:STOP?
```

This command sets the stop frequency and automatically sets the calibration type to User for a wideband I/Q calibration.

The setting enabled by this command is not affected by a signal generator power-on, preset, or *RST command.

Example

```
:CAL:WBIQ:STOP 2GHZ
```

The preceding example sets the signal generator's stop frequency to 2 GHz for a wideband IQ calibration.

Range Depends on the signal generator's frequency option
Refer to ":FREQuency:CENTer" on page 123.

Key Entry **Stop Frequency**

Communication Subsystem (:SYSTem:COMMunicate)

:GPIB:ADDRess

Supported All Models

```
:SYSTem:COMMunicate:GPIB:ADDRess <number>
:SYSTem:COMMunicate:GPIB:ADDRess?
```

This command sets the signal generator's general purpose instrument bus (GPIB) address.

The variable `<number>` is a numeric value between 0 and 30. The signal generator typically uses 19 as the instrument address. The address must be different from other GPIB devices in your system.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:SYST:COMM:GPIB:ADDR 19
```

The preceding example sets the signal generator's GPIB address to 19.

Range 0–30

Key Entry **GPIB Address**

:GTLocal

Supported All

```
:SYSTem:COMMunicate:GTLocal
```

This command sets the signal generator to local mode, enabling front panel operation.

Range N/A

Key Entry **Local**

:LAN:CONFig

Supported All Models

```
:SYSTem:COMMunicate:LAN:CONFig DHCP|MANual
:SYSTem:COMMunicate:LAN:CONFig?
```

This command selects the signal generator's internet protocol (IP) address. The dynamic host communication protocol (DHCP) selection allows the network to assign an IP address. The manual selection allows the user to enter an IP address.

Example

```
:SYST:COMM:LAN:CONF DHCP
```

The preceding example sets up the signal generator LAN configuration to use a DHCP IP address.

Key Entry **LAN Config**

:LAN:GATEway

Supported All Models

```
:SYSTem:COMMunicate:LAN:GATEway "<ipstring>"
:SYSTem:COMMunicate:LAN:GATEway?
```

This command sets the gateway for local area network (LAN) access to the signal generator from outside the current sub-network.

The `"<ipstring>"` string variable is the LAN gateway address, formatted as xxx.xxx.xxx.xxx. Refer to "Quote Usage with SCPI Commands" on page 15 for information on using quotes for different programming languages.

Using an empty string restricts access to the signal generator to local hosts on the LAN.

Example

```
:SYST:COMM:LAN:GATE "203.149.781.101"
```

The preceding example sets the signal generator's LAN gateway address.

Key Entry **Default Gateway**

:LAN:HOSTname

Supported All Models

```
:SYSTem:COMMunicate:LAN:HOSTname "<string>"
:SYSTem:COMMunicate:LAN:HOSTname?
```

This command sets the signal generator's local area network (LAN) connection hostname.

The `"<string>"` variable is the hostname for the signal generator. Refer to "Quote Usage with SCPI Commands" on page 15 for information on using quotes for different programming languages.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:SYST:COMM:LAN:HOSTname "siginst3"
```

The preceding example sets “siginst3” as the signal generator’s LAN hostname.

Key Entry **Hostname**

:LAN:IP

Supported All Models

```
:SYSTem:COMMunicate:LAN:IP "<ipstring>"
:SYSTem:COMMunicate:LAN:IP?
```

This command sets the signal generator’s local area network (LAN) internet protocol (IP) address for your IP network connection.

The `"<ipstring>"` variable is the signal generator’s IP address, formatted as xxx.xxx.xxx.xxx. Refer to “Quote Usage with SCPI Commands” on page 15 for information on using quotes for different programming languages.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:SYST:COMM:LAN:IP "202.195.207.193"
```

The preceding example sets the signal generator’s LAN IP address.

Key Entry **IP Address**

:LAN:SUBNet

Supported All Models

```
:SYSTem:COMMunicate:LAN:SUBNet "<ipstring>"
:SYSTem:COMMunicate:LAN:SUBNet?
```

This command sets the signal generator’s local area network (LAN) subnet mask address for your internet protocol (IP) network connection.

The `"<ipstring>"` variable is the subnet mask for the IP address, formatted as xxx.xxx.xxx.xxx. Refer to “Quote Usage with SCPI Commands” on page 15 for information on using quotes for different programming languages.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:SYST:COMM:LAN:SUBN "203.194.101.111"
```

The preceding example sets the signal generator’s LAN subnet mask.

Key Entry **Subnet Mask**

:PMETer:ADDRess

Supported All Models

```
:SYSTem:COMMunicate:PMETer:ADDRess <val>
:SYSTem:COMMunicate:PMETer:ADDRess?
```

This command sets the instrument address for a power meter that is controlled by the signal generator. The power meter is controlled only through a general purpose instrument bus (GPIB) cable.

The variable `<number>` is an integer numeric value between 0 and 30. The power meter address must be different from the GPIB address of the signal generator and any other GPIB instrument addresses in your system.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:SYST:COMM:PMET:ADDR 14
```

The preceding example sets the address to 14 for the power meter that is connected to and controlled by the signal generator.

Range 0–30

Key Entry **Meter Address**

:PMETer:CHANnel

Supported All Models

```
:SYSTem:COMMunicate:PMETer:CHANnel A|B
:SYSTem:COMMunicate:PMETer:CHANnel?
```

This command sets the measurement channel on a dual channel power meter that is controlled by the signal generator. A single-channel power meter uses channel A and selecting channel B will have no effect.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command. The power meter is controlled only through a general purpose instrument bus (GPIB) cable.

Example

```
:SYST:COMM:PMET:CHAN B
```

The preceding example sets the B measurement channel for the power meter that is connected to and controlled by the signal generator.

Key Entry **Meter Channel A B**

:PMETer:IDN

Supported All Models

```
:SYSTem:COMMunicate:PMETer:IDN E4418B|E4419B|E4416A|E4417A
:SYSTem:COMMunicate:PMETer:IDN?
```

This command sets the model number of the power meter that is controlled by the signal generator. The setting enabled by this command is not affected by a signal generator power-on, preset, or *RST command. The power meter is controlled only through a general purpose instrument bus (GPIB) cable.

Example

```
:SYST:COMM:PMET:IDN E4417A
```

The preceding example sets the model number for the power meter that is connected to and controlled by the signal generator.

Key Entry **Power Meter**

:PMETer:TIMEout

Supported All Models

```
:SYSTem:COMMunicate:PMETer:TIMEout <num>[<time_suffix>]
:SYSTem:COMMunicate:PMETer:TIMEout?
```

This command sets the period of time that the signal generator will wait for a valid reading from the power meter. The variable <num> has a resolution of 0.001.

The variable <num> is the time expressed as a number. The variable `<time_suffix>` are the units of time, for example mS (milliseconds) or S (seconds).

The setting enabled by this command is not affected by a signal generator power-on, preset, or *RST command. The power meter is controlled only through a general purpose instrument bus (GPIB) cable. If a timeout occurs, the signal generator reports an error message.

Example

```
:SYST:COMM:PMET:TIME .1SEC
```

The preceding example sets the timeout to 100 milliseconds for the power meter that is connected to and controlled by the signal generator.

Range 1mS–100S

Key Entry **Meter Timeout**

:SERial:BAUD

Supported All Models

```
:SYSTem:COMMunicate:SERial:BAUD <number>
:SYSTem:COMMunicate:SERial:BAUD?
```

This command sets the baud rate for the rear panel RS-232 interface labeled RS-232. The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

The variable <number> is an integer value corresponding to baud rates: 300, 2400, 4800, 9600, 19200, 38400, and 57600.

Example

```
:SYST:COMM:SER:BAUD 9600
```

The preceding example sets the baud rate for serial communication to 9600.

Key Entry **RS-232 Baud Rate**

:SERial:ECHO

Supported All Models

```
:SYSTem:COMMunicate:SERial:ECHO ON|OFF
:SYSTem:COMMunicate:SERial:ECHO?
```

This command enables or disables the RS-232 echo, and is not affected by a power-on, preset, or `*RST` command. Characters sent to the signal generator are displayed or echoed to the controller display.

Example

```
:SYST:COMM:SER:ECHO ON
```

The preceding example enables RS-232 echoing.

Key Entry **RS-232 ECHO Off On**

:SERial:RESet

Supported All Models

`:SYSTem:COMMunicate:SERial:RESet`

This event command resets the RS-232 buffer and discards unprocessed SCPI input received at the RS-232 port.

Key Entry **Reset RS-232**

:SERial:TOUT

Supported All Models

`:SYSTem:COMMunicate:SERial:TOUT <val>`
`:SYSTem:COMMunicate:SERial:TOUT?`

This command sets the RS-232 serial port timeout value. If further input is not received within the timeout period specified while a SCPI command is processed, the command aborts and clears the input buffer. The variable `<val>` is entered in seconds. The setting is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:SYST:COMM:SER:TOUT 2SEC
```

The preceding example sets the RS-232 timeout for 2 seconds.

Range 1–25

Key Entry **RS-232 Timeout**

Diagnostic Subsystem (:DIAGnostic[:CPU]:INFOrmation)

:BOARds

Supported All Models

```
:DIAGnostic[:CPU]:INFOrmation:BOARds?
```

This query returns a list of the boards installed in the signal generator. The information is returned in the following format:

```
"<board_name,part_number,serial_number,version_number,status>"
```

This information format will repeat for each of the signal generator’s detected boards.

Key Entry **Installed Board Info**

:CCOunt:ATTenuator

Supported E8267D and E8257D with Option 1E1

```
:DIAGnostic[:CPU]:INFOrmation:CCOunt:ATTenuator?
```

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

Key Entry **Diagnostic Info**

:CCOunt:PON

Supported All Models

```
:DIAGnostic[:CPU]:INFOrmation:CCOunt:PON?
```

This query returns the cumulative number of times the signal generator has been powered-on.

Key Entry **Diagnostic Info**

:DISPlay:OTIMe

Supported All Models

```
:DIAGnostic[:CPU]:INFOrmation:DISPlay:OTIMe?
```

This query returns the cumulative number of hours the display has been on.

Key Entry **Diagnostic Info**

:LICENse:AUXiliary

Supported All Models

`:DIAGnostic[:CPU]:INFOrmation:LICense:AUXiliary?`

This query returns a listing of current external software application license numbers for an auxiliary instrument.

Key Entry **Auxiliary Software Options**

:OPTions

Supported All Models

`:DIAGnostic[:CPU]:INFOrmation:OPTions?`

This query returns a list of options installed in the signal generator.

Key Entry **Options Info**

:OPTions:DETail

Supported All Models

`:DIAGnostic[:CPU]:INFOrmation:OPTions:DETail?`

This query returns the options installed, option revision, and digital signal processing (DSP) version if applicable.

Key Entry **Options Info**

:OTIMe

Supported All Models

`:DIAGnostic[:CPU]:INFOrmation:OTIMe?`

This query returns the cumulative number of hours that the signal generator has been on.

Key Entry **Diagnostic Info**

:REVision

Supported All Models

`:DIAGnostic[:CPU]:INFOrmation:REVision?`

This query returns the CPU bootstrap read only memory (boot ROM) revision date. In addition, the query returns the revision, creation date, and creation time for the firmware.

Key Entry **Diagnostic Info**

:SDATe

Supported All Models

`:DIAGnostic[:CPU]:INFOrmation:SDATe?`

This query returns the date and time stamp for the signal generator’s firmware.

Key Entry **Diagnostic Info**

Display Subsystem (:DISPlay)

:ANNotation:AMPLitude:UNIT

Supported All Models

```
:DISPlay:ANNotation:AMPLitude:UNIT DBM|DBUV|DBUVEMF|V|VEMF|DB
:DISPlay:ANNotation:AMPLitude:UNIT?
```

This command sets the displayed front panel amplitude units.

If the amplitude reference state is set to on, the query returns units expressed in dB. Setting any other unit will cause a setting conflict error stating that the amplitude reference state must be set to off. Refer to ":REFerence:STATe" on page 162 for more information.

Example

```
:DISP:ANN:AMPL:UNIT DB
```

The preceding example sets DB as the amplitude units shown on the signal generator's front panel display.

***RST** **dBm**

:ANNotation:CLOCk:DATE:FORMat

Supported All Models

```
:DISPlay:ANNotation:CLOCk:DATE:FORMat MDY|DMY
:DISPlay:ANNotation:CLOCk:DATE:FORMat?
```

This command selects the date format. The choices are month-day-year (MDY) or day-month-year (DMY) format. The date is shown on the signal generator's front panel display.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:DISP:ANN:CLOC:DATA:FORM DMY
```

The preceding example sets the date format shown on the signal generator's front panel display to DMY.

:ANNotation:CLOCk[:STATe]

Supported All Models

```
:DISPlay:ANNotation:CLOCk[:STATe] ON|OFF|1|0
:DISPlay:ANNotation:CLOCk[:STATe]?
```

This command enables or disables the digital clock shown at the lower right side of the front panel display.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:DISP:ANN:CLOC OFF
```

The preceding example disables the digital clock on the signal generator's front panel display.

:BRIGhtness

Supported All Models

```
:DISPlay:BRIGhtness <val>
:DISPlay:BRIGhtness?
```

This command sets the display brightness (intensity). The brightness can be set to the minimum level (0.02), maximum level (1), or in between by using fractional numeric values (0.03–0.99).

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:DISP:BRIG .45
```

The preceding example sets display intensity to .45.

Range 0.02–1

Key Entry **Brightness**

:CAPTure

Supported All Models

```
:DISPlay:CAPTure
```

This command allows the user to capture the current display and store it in the signal generator's memory.

The display capture is stored as DISPLAY.BMP in the Binary file system. This file is overwritten with each subsequent display capture. The file can be down-loaded in the following manner:

1. Log on to the signal generator using file transfer protocol (FTP).
2. Change to the BIN directory using the FTP `cd` command.
3. Retrieve the file by using the FTP `get` command.

:CONTrast

Supported All Models

```
:DISPlay:CONTrast <val>
:DISPlay:CONTrast?
```

This command sets the contrast for the signal generator's display. The variable `<val>` is expressed as a fractional number between 0 and 1. The contrast can be set to the maximum level (1), minimum level (0), or in between by using fractional numeric values (0.001–0.999).

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:DISP:CONT .45
```

The preceding example sets the display contrast to .45.

Range 0–1

Key Entry Display contrast hardkeys are located below the display.

:INVerse

Supported All Models

```
:DISPlay:INVerse ON|OFF|1|0
:DISPlay:INVerse?
```

This command sets the display of the source to inverse video mode. The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:DISP:INV OFF
```

The preceding example sets the display video to normal (not inverse).

Key Entry **Inverse Video Off On**

:REMote

Supported All Models

```
:DISPlay:REMote ON|OFF|1|0
:DISPlay:REMote?
```

This command enables or disables display updating when the signal generator is remotely controlled.

ON (1) This choice updates the signal generator display so that you can see the settings change as the commands are executed, however, this will decrease the signal generator’s response time.

OFF (0) This choice turns off display updating which will optimizing the signal generator’s response time.

The setting enabled by this command is not affected by signal generator preset or `*RST` command. However, cycling the signal generator power will reset it to zero.

Example

```
:DISP:REM 0
```

The preceding example turns off display updating.

Key Entry **Update in Remote Off On**

Display Off On

Supported All Models

```
:DISPlay[:WINDow][:STATe] ON|OFF|1|0
:DISPlay[:WINDow][:STATe]?
```

This command is used to either blank out (OFF or 0) the display screen or turn it on (ON or 1).

A signal generator preset, `*RST` command, or cycling the power will turn the display on.

Example

```
:DISP OFF
```

The preceding example blanks out the signal generator’s display.

IEEE 488.2 Common Commands

*CLS

Supported All Models

`*CLS`

The Clear Status (CLS) command clears the Status Byte register, the Data Questionable Event register, the Standard Event Status register, and the Standard Operation Status register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

*ESE

Supported All Models

`*ESE <val>`

This command enables bits in the Standard Event Enable register. Bits enabled and set in this register will set the Standard Event Status Summary bit (bit 5) in the Status Byte register. When bit 5 (decimal 32) in the Status Byte register is set, you can read the Standard Event register using the `*ESR` command and determine the cause.

The Standard Event Enable register state (bits enabled with this command) is not affected by signal generator preset or `*RST`. The register will be cleared when the signal generator is turned off unless the command "*PSC" is used before turning it off.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

`*ESE 129`

This command enables bit 0 (decimal 1, Operation Complete) and bit 7 (decimal 128, Power On) in the Standard Event Status Enable register.

Range 0–255

*ESE?

Supported All Models

`*ESE?`

This query returns the decimal sum of the enabled bits in the Standard Event Enable register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

*ESR?

Supported All Models

NOTE	This is a destructive read. The data in the register is latched until it is queried. Once queried, the data is cleared. Refer to the *PSG Programming Guide* for more information.

`*ESR?`

This query returns the decimal sum of the bits set in the Standard Event register.

*IDN?

Supported All Models

`*IDN?`

This query requests an identification string from the signal generator. The `IDN` string consists of the following information:

`<company_name>, <model_number>, <serial_number>, <firmware_revision>`

The identification information can be modified. Refer to ":IDN" on page 101 for more information.

Key Entry **Diagnostic Info**

*OPC

Supported All Models

`*OPC`

The Operation Complete (OPC) command sets bit 0 in the Standard Event register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

*OPC?

Supported All Models

`*OPC?`

The Operation Complete (OPC) query returns the ASCII character 1 in the Standard Event register indicating completion of all pending operations.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

*PSC

Supported All Models

`*PSC ON|OFF|1|0`

The power-on Status Clear (PSC) command controls the automatic power-on clearing of the Service Request Enable register, the Standard Event Status Enable register, and the device-specific event enable registers.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

ON (1)	This choice enables the power-on clearing of the listed registers.
OFF (0)	This choice disables the clearing of the listed registers and they retain their status when a power-on condition occurs.

Example

`*PSC ON`

This command clears all listed registers at power-on.

*PSC?

Supported All Models

`*PSC?`

The power-on Status Clear (PSC) query returns the flag (1 or 0) setting as enabled by the *PSC command.

*RCL

Supported All Models

```
*RCL <reg>,<seq>
```

The Recall (RCL) command recalls the state from the specified memory register <reg> in the specified sequence <seq>.

Range *registers*: 0–99 *Sequences*: 0–9

Key Entry **RECALL Reg** **Select Seq:**

*RST

Supported All Models

```
*RST
```

The Reset (RST) command resets most signal generator functions to a factory-defined state.

Each command description in this reference shows the *RST value if the signal generator’s setting is affected.

*SAV

Supported All Models

```
*SAV <reg>,<seq>
```

The Save (SAV) command saves the state of the signal generator to the specified memory register `<reg>` of the specified sequence `<seq>`. Settings such as frequency, attenuation, power, and settings that do not survive a power cycle or an instrument reset can be saved. Data formats, arb setups, list sweep values, table entries, and so forth are not stored. Only a reference to the data file name is saved. Refer to the *PSG User’s Guide* and *PSG Programming Guide* for more information on saving and recalling instrument states.

Range *registers*: 0–99 *Sequences*: 0–9

Key Entry **Save Reg** **Save Seq[n] Reg[nn]**

*SRE

Supported All Models

```
*SRE <val>
```

The Service Request Enable (SRE) command enables bits in the Service Request Enable register. Bits enabled and set in this register will set bits in the Status Byte register.

The variable `<val>` is the decimal sum of the bits that are enabled. Bit 6 (value 64) is not available in this register and therefore cannot be enabled by this command. Because bit 6 is not available, entering values from 64 to 127 is equivalent to entering values from 0 to 63.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

The setting enabled by this command is not affected by signal generator preset or `*RST`. However, cycling the signal generator power will reset this register to zero.

Range 0–63, 128–191

*SRE?

Supported All Models

`*SRE?`

The Service Request Enable (SRE) query returns the decimal sum of bits enabled in the Service Request Enable register. Bit 6 (decimal 64) is not available in this register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–63, 128–191

*STB?

Supported All Models

`*STB?`

This command reads the decimal sum of the bits set in the Status Byte register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–255

*TRG

Supported All Models

`*TRG`

The Trigger (TRG) command triggers the device if BUS is the selected trigger source, otherwise, *TRG is ignored. Refer to “:TRIGger[:SEQuence]:SOURce” on page 116 for more information on triggers.

*TST?

Supported All Models

`*TST?`

The Self-Test (TST) query initiates the internal self-test and returns one of the following results:

`0`	This shows that all tests passed.
`1`	This shows that one or more tests failed.

Key Entry **Run Complete Self Test**

*WAI

Supported All Models

`*WAI`

The Wait-to-Continue (WAI) command causes the signal generator to wait until all pending commands are completed, before executing any other commands.

Memory Subsystem (:MEMory)

:CATalog:BINary

Supported All Models

`:MEMory:CATalog:BINary?`

This command outputs a list of binary files. 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. Each file listing parameter will be in the following form:

`"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **Binary**

:CATalog:BIT

Supported E8267D with Option 601or 602

`:MEMory:CATalog:BIT?`

This command outputs a list of bit files. 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. Each file listing parameter will be in the following form:

`"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **Bit**

:CATalog:DMOD

Supported E8267D with Option 601or 602

`:MEMory:CATalog:DMOD?`

This command outputs a list of arbitrary waveform digital modulation files. 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. Each file listing parameter will be in the following form: `"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **DMOD**

:CATalog:FIR

Supported E8267D with Option 601or 602

`:MEMory:CATalog:FIR?`

This command outputs a list of finite impulse response (FIR) filter files. 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. Each file listing parameter will be in the following form: `"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **FIR**

:CATalog:FSK

Supported E8267D with Option 601or 602

`:MEMory:CATalog:FSK?`

This command outputs a list of frequency shift keying (FSK) files. 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. Each file listing parameter will be in the following form: `"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **FSK**

:CATalog:IQ

Supported E8267D with Option 601or 602

`:MEMory:CATalog:IQ?`

This command outputs a list of IQ files. 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. Each file listing parameter will be in the following form: `"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **I/Q**

:CATalog:LIST

Supported All Models

`:MEMory:CATalog:LIST?`

This command outputs a list of List Sweep files. 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. Each file listing parameter will be in the following form: `"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **List**

:CATalog:MDMod

Supported E8267D with Option 601or 602

`:MEMory:CATalog:MDMod?`

This command outputs a list of arbitrary waveform multicarrier digital modulation (MDMod) files. 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. Each file listing parameter will be in the following form: `"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **MDMOD**

:CATalog:MTONe

Supported E8267D with Option 601or 602

`:MEMory:CATalog:MTONe?`

This command outputs a list of arbitrary waveform multitone files. 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. Each file listing parameter will be in the following form:

`"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **MTONE**

:CATalog:SEQ

Supported E8267D with Option 601or 602

`:MEMory:CATalog:SEQ?`

This command outputs a list of arbitrary waveform sequence files. 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. Each file listing parameter will be in the following form:

`"<file_name,file_type,file_size>"`

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **Seq**

:CATalog:SHAPe

Supported E8267D with Option 601or 602

`:MEMory:CATalog:SHAPe?`

This command outputs a list of burst shape files. 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. Each file listing parameter will be in the following form:

```
"<file_name,file_type,file_size>"
```

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **Shape**

:CATalog:STATe

Supported All Models

```
:MEMory:CATalog:STATe?
```

This command outputs a list of state files. 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. Each file listing parameter will be in the following form:

```
"<file_name,file_type,file_size>"
```

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **State**

:CATalog:UFLT

Supported All Models

```
:MEMory:CATalog:UFLT?
```

This command outputs a list of user-flatness correction files. 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. Each file listing parameter will be in the following form:

```
"<file_name,file_type,file_size>"
```

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Key Entry **User Flatness**

:CATalog[:ALL]

Supported All Models

`:MEMory:CATalog[:ALL]?`

This command outputs a list of all files in the memory subsystem, but does not include files stored in the Option 601or 602 baseband generator memory. The return data is in the following form: <mem_used>,<mem_free>{,"<file_listing>"}

The signal generator returns the two memory usage parameters and as many file listings as there are files in the memory subsystem. Each file listing parameter is in the following form: "<file_name,file_type,file_size>"

See Table 2-1 on page 66 for file types, and "File Name Variables" on page 12 for syntax.

Key Entry **All**

:COPY[:NAME]

Supported All Models

:MEMory:COPY[:NAME] "<src_name>","<dest_name>"

This command copies the data from one file into another file. The file can use the same name if the specified directory is different. For example, if the file resides in non-volatile waveform memory (NVWFM) it can be copied, using the same name, to the signal generator's volatile memory (WFM1).

"<src_name>" This variable names a file residing in memory that will be copied. For information on the file name syntax, see "File Name Variables" on page 12.

"<dest_name>" This variable names the file that is a copy of the "<src_name>" file.

Example

```
:MEM:COPY "/USER/IQ/4QAM","/USER/IQ/test_QAM"
```

The preceding example copies the 4QAM file in the signal generator's `/USER/IQ` directory to a file named `test_QAM` and saves it in the same directory.

Key Entry **Copy File**

:DATA

Supported All Models

```
:MEMory:DATA "<file_name>",<data_block>
:MEMory:DATA? "<file_name>"
```

This command loads waveform data into signal generator memory using the `<data_block>` parameter and saves the data to a file designated by the `"<file_name>"` variable. The query returns the file contents of the file as a datablock.

The waveform file must be located in volatile waveform memory (WFM1) before it can be played by the signal generator's Dual ARB player. For downloads directly into volatile waveform memory use the path `"WFM1:<file_name>"`. For downloads to non-volatile waveform memory, use the path `"NVWFM:<file_name>"`.

Refer to "File Name Variables" on page 12 for information on the file name syntax.

`"<file_name>"` This variable names the destination file, including the directory path. Refer to "ARB Waveform File Directories" on page 13 for information on directory paths and the file name syntax.

`<data_block>` This parameter represents the data and file length parameters. The data in the file is represented by the `<data_block>` variable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

NOTE ARB waveform files created using the :DATA command cannot be retrieved or uploaded. Attempting to do so will cause the signal generator to display the message: `ERROR:221, Access denied`. To download ARB data to files for later retrieval, use the :DATA:UNPRotected" command on page 59.

Example

```
:MEM:DATA "NVWFM:IQ_Data",#210Qaz37pY9oL
```

The preceding example downloads 10 bytes of data to a file, `IQ_Data`., in the signal generator's non-volatile memory. The table shown below describes the command parameters.

• `"NVWFM:IQ_Data"`	`IQ_Data` is the data filename. The directory path is specified along with the filename
• `#210Qaz37pY9oL`	Data block
`#`	This character indicates the beginning of the data block
`2`	Number of digits in the byte count
`10`	Byte count

• `"NVWFM:IQ_Data"`	`IQ_Data` is the data filename. The directory path is specified along with the filename
`Qaz37pY9oL`	10 bytes of data

:DATA:APPend

Supported All

`:MEMory:DATA:APPend "<file_name>",<data_block>`

This commands appends data to an existing file stored in signal generator memory.

`"<file_name>"` This variable names the destination file and directory path. Refer to "File Name Variables" on page 12 for information on the file name syntax.

`<data_block>` This parameter represents the data and file length parameters. The data in the file is represented by the `<data_block>` variable. The file length parameters are used by the signal generator for allocating memory.

Refer to the *PSG Programming Guide* for more information on downloading and using files.

Example

`:MEM:DATA:APPend "NVWFM:IQ_Data",#14Y9oL`

The preceding example downloads and appends the data, `Y9oL,` to an existing file named `IQ_Data` stored in the signal generator's non-volatile memory (NVWFM).

• `"NVWFM:IQ_Data"`	`IQ_Data` is the filename to append data to. The directory path is specified along with the filename.
• `#14Y9oL`	Data block
`#`	This character indicates the beginning of the data block
`1`	Number of digits in the byte count
`4`	Byte count
`Y9oL`	4 bytes of data

:DATA:BIT

Supported E8267D with Option 601or 602

`:MEMory:DATA:BIT "<file_name>",<bit_count>,<data_block>`
`:MEMory:DATA:BIT? "<file_name>"`

This command loads bit data into signal generator memory using the `<bit_count>` and `<data_block>` parameters and saves the data to a file designated by the `"<file_name>"` variable.

The query returns the bit count, file length information, and the data.

`"<file_name>"`	This variable names the destination file and the directory path. Refer to "File Name Variables" on page 12 for information on the file name syntax.
`<bit_count>`	This number represents the number of bits in the data block.
`<data_block>`	This parameter represents the data and file length parameters. The data in the file is represented by the `<data_block>` variable. The file length parameters are used by the signal generator for allocating memory.

Refer to the *PSG Programming Guide* for more information on downloading and using files.

Example

```
:MEM:DATA:BIT "/USER/BIT/Test_Data",16,#12Qz
```

The preceding example downloads bit data to the file, `Test_Data`. The table below describes the command parameters.

• `"/USER/BIT/Test_Data"`	`Test_Data` is the bit data filename. The directory path is specified along with the filename
• `16`	Number of bits in the data block
• `#12Qz`	Data block
`#`	This character indicates the beginning of the data block
`1`	Number of digits in the byte count
`2`	Byte count
`Qz`	16 bits of data (ascii representation of bit data)

:DATA:FIR

Supported E8267D with Option 601or 602

```
:MEMory:DATA:FIR "<file_name>",osr,coefficient{,coefficient}
:MEMory:DATA:FIR? "<file_name>"
```

This command loads oversample ratio (OSR) and user-defined finite impulse response (FIR) coefficient data into a file in the signal generator's non–volatile memory (NVWFM). The query returns the oversample ratio and coefficient data.

`"<file_name>"`	This variable is the directory path and file name of the destination file. Refer to "File Name Variables" on page 12 for information on the file name syntax.
`osr`	The OSR is the number of filter taps per symbol.
`coefficient`	This variable is the FIR coefficient. The maximum number of coefficients is 1024.

`{,coefficient}` This optional variable is used when you enter additional coefficients.

Refer to the *PSG Programming Guide* for more information on downloading and using files.

Example

```
:MEM:DATA:FIR "/USER/FIR/FIR_1",4,0,0,0,0,0,0.000001,0.000012,0.000132,
0.001101,0.006743,0.030588,0.103676,0.265790,0.523849,0.809508,1,1,
0.809508,0.523849,0.265790,0.103676,0.030588,0.006743,0.001101,0.000132,
0.000012,0.000001,0,0,0,0,0
```

The preceding example downloads FIR coefficient and oversampling ratio data to the signal generator's non–volatile memory in a file named FIR_1.

Range *osr:* 1–32

coefficient: –1000 to 1000

Key Entry **Oversample Ratio**

:DATA:FSK

Supported E8267D with Option 601or 602

```
:MEMory:DATA:FSK "<file_name>",<num_states>,<f0>,<f1>,...<f(n)>
[,<diff_state>,<num_diff_states>,<diff1>,...<diff(n)>]
:MEMory:DATA:FSK? "<file_name>"
```

This command loads custom frequency shift keying (FSK) data into a file in the signal generator's non–volatile memory (NVWFM).

The query returns data in the following form:

```
<num_states>,<f0>,<f1>,...<f(n)>,<diff_state>,<num_diff_states>,<diff1>,
...<diff(n)>
```

`"<file_name>"` This variable string identifies the name of the FSK file. The filename must be enclosed with quotation marks. Refer to "File Name Variables" on page 12 for information on the file name syntax.

`<num_states>` This variable identifies the number of frequency states.

`<f0>` This variable identifies the value of the first frequency state.

`<f1>,...<f(n)>` This variable identifies the value of the second and subsequent frequency states with a frequency resolution of 0.1Hz.

`<diff_state>` This variable enables or disables differential encoding.

`<num_diff_states>` This variable identifies the number of differential states.

`<diff0>` This variable identifies the value of the first differential state.

`<diff1>,...<diff(n)>` This variable identifies the value of the second and subsequent differential states.

Refer to the *PSG Programming Guide* for more information on downloading and using files.

Example

```
:MEM:DATA:FSK "/USER/FSK/4FSK",4,–2kHZ,–1kHZ,2kHZ,1kHZ,ON,2,1,0
```

The preceding example downloads a four–level FSK data to a file named `4FSK` that has four states (frequencies): –2kHZ, –1kHZ, 2kHZ, 1kHZ; differential encoding is toggled ON, and there are two differential states 1 and 0. The table shown below describes the command parameters.

- `"/USER/FSK/4FSK"` — `4FSK` is the FSK data filename. The directory path is specified along with the filename
- `4` — Number of states
- `-2kHz` — First frequency state
- `-1kHz` — Second frequency state
- `2kHz` — Third frequency state
- `1kHz` — Fourth frequency state
- `ON` — Differential encoding is on
- `2` — Number of differential states
- `1` — Value of the first differential state.
- `0` — Value of the second differential state.

Range *num_diff_states:* 0–256

num_states: 2–16

f0–f(n): –20MHZ to 20MHZ

diff0–diff(n): –128 to 127

:DATA:IQ

Supported E8267D with Option 601or 602

```
:MEMory:DATA:IQ "<file_name>",<offsetQ>,<num_states>,<i0>,<q0>,<i1>,
<q1>,...<i(n)>,<q(n)>[,<diff_state>,<num_diff_states>,<diff0>,<diff1>,..
<diff(n)>]
:MEMory:DATA:IQ? "<file_name>"
```

This command loads custom I/Q data into a file in the signal generator’s non–volatile waveform memory (NVWFM).

The query returns data in the following form:

`<offsetQ>,<num_states>,<i0>,<q0>,<i1>,<q1>,...<i(n)>,<q(n)>,<diff_state>,<num_diff_states>,<diff0>,<diff1>,...<diff(n)>`

`"<file_name>"` This variable string identifies the name of the I/Q file. The filename must be enclosed with quotation marks. Refer to "File Name Variables" on page 12 for information on the file name syntax.

`<offsetQ>` This variable enables (1) or disables (0) the Q output delay by 1/2 symbol from the I output.

`<num_states>` This is the number of symbols.

`<i0>...<i(n)>` This is the I value of the first and subsequent I symbols.

`<q0>...<q(n)>` This is the Q value of the first and subsequent Q symbols.

`<diff_state>` This variable enables and disables differential encoding.

`<num_diff_states>` This variable identifies the number of differential states.

`<diff0>` This variable identifies the value of the first differential state.

`<diff1,...diff(n)>` This variable identifies the value of the second and subsequent differential states.

Refer to the *PSG Programming Guide* for more information on downloading and using files.

Example

```
:MEM:DATA:IQ "/USER/IQ/Test_BPSK",1,2,1,0,0,0
```

The preceding example loads and stores a two-symbol I/Q file named `Test_BPSK` that has a Q offset. The table shown below describes the command parameters.

• `"/USER/IQ/Test_BPSK"`	`Test_Data` is the bit data filename. The directory path is specified along with the filename
• `1`	Q Offset. The Q output delay is enabled.
• `2`	Number of symbols
• `1`	Value of the first I symbol
• `0`	Value of the first Q symbol.
• `0`	Value of the second I symbol
• `0`	Value of the second Q symbol.

Range *num_states:* 2–256

i0–i(n): −1 to 1

q0–q(n): −1 to 1

num_diff_states: 0–256

diff0–diff(n): -128 to 127

:DATA:PRAM:FILE:BLOCk

Supported E8267D with Option 601or 602

`:MEMory:DATA:PRAM:FILE:BLOCk "<file_name>", <data_block>`

This command loads block-formatted data directly into pattern RAM volatile memory (WFM1). Pattern RAM memory describes how memory (WFM1) is used and is not a distinct piece of memory. A PRAM file is specified as an array of bytes. No directory path name is needed.

`"<file_name>"` This variable names the destination file. Refer to "File Name Variables" on page 12 for information on the file name syntax.

`<data_block>` This parameter represents the data and file length parameters. The data in the file is represented by the `<data_block>` variable. The file length parameters are used by the signal generator for allocating memory.

Pattern Ram files are binary files downloaded directly into waveform memory as an array of bytes. Each byte specifies a data bit (`LSB 0`), a burst bit (`BIT 2`), and an Event 1 output bit (`BIT 6`). Refer to the *PSG Programming Guide* for more information on downloading and using files.

Example

`:MEM:DATA:PRAM:FILE:BLOC "PRAM_Data",#14Yq8L`

The preceding example downloads PRAM data to a file named `PRAM_Data` into the signal generator's volatile memory.

- `"PRAM_Data"` — `PRAM_Data` is the data filename. PRAM files are saved to the signal generator's non-volatile memory (WFM1).
- `#14Yq8L` — Data block

`#`	This character indicates the beginning of the data block
`1`	Number of digits in the byte count
`4`	Byte count
`Yq8L`	4 bytes of data

:DATA:PRAM:FILE:LIST

Supported E8267D with Option 601or 602

`:MEMory:DATA:PRAM:FILE:LIST "<file_name>",<uint8>[,<uint8>,<...>]`

This command loads list-formatted data directly into pattern RAM volatile memory (WFM1). Pattern RAM memory describes how memory (WFM1) is used and is not a distinct piece of memory. A PRAM file is specified as an array of bytes.

NOTE This command should be preceded by a *WAI (Wait-to-Continue) command to ensure that all pending operations are completed, before loading the list.

`"<file_name>"` This variable names the destination file. Refer to "File Name Variables" on page 12 for information on the file name syntax.

`<uint8>` This variable is any of the valid 8–bit, unsigned integer values between 0 and 255.

`[,<uint8>,<...>]` This variable identifies the value of the second and subsequent 8–bit unsigned integer variables.

Pattern Ram files are binary files downloaded directly into waveform memory as an array of bytes. Each byte specifies a data bit (`LSB 0`), a burst bit (`BIT 2`), and an Event 1 output bit (`BIT 6`). Refer to the *PSG Programming Guide* for more information on downloading and using files.

Example

`:MEM:DATA:PRAM:LIST "Pram_Data", 85,21,21,20,20,100`

The preceding example downloads PRAM data, in list format, to a file named `Pram_Data` in the signal generator's volatile memory (WFM1).

- `"Pram_Data"` — `Pram_Data` is the data filename. PRAM files are saved to the signal generator's non–volatile memory (WFM1).
- `85` — The first 8–bit integer value
- `21,21,20,20,100` — Subsequent 8–bit integer values.

Range 0–255

:DATA:PRAM?

NOTE This query is no longer supported; however, it is still valid for backward compatibility. Refer to ":DATA:PRAM?" on page 377 for information on this command.

:DATA:PRAM:BLOCk

NOTE This command has been replaced by ":DATA:PRAM:FILE:BLOCk" on page 56. This command is no longer supported; however, it is still valid for backward compatibility. Refer to ":DATA:PRAM:BLOCk" on page 377 for information.

:DATA:PRAM:LIST

NOTE This command has been replaced by ":DATA:PRAM:FILE:LIST" on page 57. This command is no longer supported; however, it is still valid for backward compatibility. Refer to ":DATA:PRAM:LIST" on page 377 for information.

:DATA:SHAPe

Supported E8267D with Option 601or 602

```
:MEMory:DATA:SHAPe
"<file_name>",<rise_pnts>,<rp0>,<rp1>,...<fall_points>,<fp0>,
<fp1>,...<fp(n)>
:MEMory:DATA:SHAPe? "<file_name>"
```

This command loads a burst shape file into the signal generator's non–volatile memory (NVWFM).

`"<file_name>"` This variable names the destination file and directory path. Refer to "File Name Variables" on page 12 for information on the file name syntax.

`rise_pnts` This variable indicates the number of rise points used to describe the burst shape rising slope.

`rp0,...rp(n)` This variable defines each successive rise point, where 0 is no power and 1 is full power.

`fall_points` This variable indicates the number of fall points used to describe the burst shape falling slope.

`fp0,...fp(n)` This variable defines each successive fall point, where 1 is full power and 0 is no power.

Refer the *PSG Programming Guide* for more information on downloading and using files.

Example

```
:MEM:DATA:SHAP "/USER/SHAPE/Shape_File",6,0,0.2,0.4,0.6,0.8,1.0,2,0.5,0
```

The preceding example loads shape data to a file named `Shape_File` in the signal generator's non–volatile memory.

• `"/USER/SHAPE/Shape_File"`	`Shape_File` is the data filename. The directory path is specified along with the file name.
• `6`	Number of rise points describing the burst shape.
• `0,0.2,0.4,0.6,0.8,1.0`	Rise point values.
• `2`	Number of fall points describing the burst shape.
• `0.5,0`	Fall point values.

Range *num_rise_points:* 2–256

num_fall_points: 2–256

rp0–rp(n): 0.0–1.0

fp0–fp(n): 0.0–1.0

:DATA:UNPRotected

Supported E8267D with Option 601or 602

```
:MEMory:DATA:UNPRotected "<file_name>",<data_block>
```

This command allows you to download data and store it in a file on the signal generator with the ability to retrieve it. This command is intended for downloading waveform data; however, you can use it to download all types of data.

NOTE If you do not use the UNPRotected command when downloading a waveform file, you will not be able to retrieve or upload the file. Attempting to do so will cause the signal generator to display the message: `ERROR:221, Access denied`.

The UNPRotected command does not require the directory path in the "<file_name>" parameter if the destination directory is BINARY.

Waveform files created with Agilent's Signal Studio are encrypted. These files can be used in other signal generators (provided the other signal generator has the same options and licenses required by the file) only if the SECUREWAVE directory path is specified in both the download and upload command parameters. The securewave directory path is SNVWFM: for non-volatile waveform memory and SWFM1: for volatile waveform memory.

"<file_name>"	This variable names the destination file and directory path. Refer to "File Name Variables" on page 12 for information on the file name syntax.
<data_block>	This parameter represents the data and file length parameters. The data in the file is represented by the <data_block> variable.

Refer to the *PSG Programming Guide* for more information on downloading and using files.

Example

```
:MEM:DATA:UNPR "NVWFM:Data_File",#18Qz37pY9o
```

The preceding example downloads waveform data to a file named Data_File in the signal generator's non-volatile memory. The table shown below describes the command parameters.

• "NVWFM:Data_File"	Data_File is the data filename. The directory path is implied along with the filename.
• #18Qz37pY9o	Data block
#	This character indicates the beginning of the data block
1	Number of digits in the byte count
8	Byte count
Qz37pY9o	8 bytes of data

:DELete:ALL

Supported All Models

CAUTION Using this command deletes all user files including binary, list, state, and flatness correction files, and any saved setups which use the front panel table editor. However, this does not include files stored in the Option 601or 602 baseband generator memory. You cannot recover the files after executing this command.

```
:MEMory:DELete:ALL
```

This command clears the file system of all user files.

Key Entry **Delete All Files**

:DELete:BINary

Supported All Models

`:MEMory:DELete:BINary`

This command deletes all binary files.

Key Entry **Delete All Binary Files**

:DELete:BIT

Supported E8267D with Option 601or 602

`:MEMory:DELete:BIT`

This command deletes all bit files.

Key Entry **Delete All Bit Files**

:DELete:DMOD

Supported E8267D with Option 601or 602

`:MEMory:DELete:DMOD`

This command deletes all arbitrary waveform digital modulation (DMOD) files.

Key Entry **Delete All ARB DMOD Files**

:DELete:FIR

Supported E8267D with Option 601or 602

`:MEMory:DELete:FIR`

This command deletes all finite impulse response (FIR) filter files.

Key Entry **Delete All FIR Files**

:DELete:FSK

Supported E8267D with Option 601or 602

`:MEMory:DELete:FSK`

This command deletes all frequency shift keying (FSK) files.

Key Entry **Delete All FSK Files**

:DELete:IQ

Supported E8267D with Option 601or 602

`:MEMory:DELete:IQ`

This command deletes all I/Q files.

Key Entry **Delete All I/Q Files**

:DELete:LIST

Supported All Models

`:MEMory:DELete:LIST`

This command deletes all List files.

Key Entry **Delete All List Files**

:DELete:MDMod

Supported E8267D with Option 601or 602

`:MEMory:DELete:MDMod`

This command deletes all arbitrary waveform multicarrier digital modulation (MDMod) files.

Key Entry **Delete All ARB MDMOD Files**

:DELete:MTONe

Supported E8267D with Option 601or 602

`:MEMory:DELete:MTONe`

This command deletes all arbitrary waveform multitone files.

Key Entry **Delete All ARB MTONE Files**

:DELete:SEQ

Supported E8267D with Option 601or 602

`:MEMory:DELete:SEQ`

This command deletes all sequence files.

Key Entry **Delete All Sequence Files**

:DELete:SHAPe

Supported E8267D with Option 601or 602

`:MEMory:DELete:SHAPe`

This command deletes all burst shape files.

Key Entry **Delete All Shape Files**

:DELete:STATe

Supported All Models

`:MEMory:DELete:STATe`

This command deletes all state files.

Key Entry **Delete All State Files**

:DELete:UFLT

Supported All Models

`:MEMory:DELete:UFLT`

This command deletes all user-flatness correction files.

Key Entry **Delete All UFLT Files**

:DELete[:NAME]

Supported All Models

`:MEMory:DELete[:NAME] "<file_name>"`

This clears the user file system of `"<file_name>"`. When deleting an ARB waveform file, the associated marker and header files are also deleted.

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Example

```
:MEM:DEL "/USER/WAVEFORM/Test_Data"
```

The preceding example deletes the file named `Test_Data` from the signal generator's non–volatile memory.

Key Entry **Delete File**

:FREE[:ALL]

Supported All Models

```
:MEMory:FREE[:ALL]?
```

This command returns the number of bytes left in the user file system.

Key Entry **All**

:LOAD:LIST

Supported All Models

```
:MEMory:LOAD:LIST "<file_name>"
```

This command loads a List Sweep file.

Example

```
:MEM:LOAD:LIST "List_Data"
```

The preceding example loads the file `"List_Data"` into volatile waveform memory.

Key Entry **Load From Selected File**

:MOVE

Supported All Models

```
:MEMory:MOVE "<src_file>","<dest_file>"
```

This command renames the `src_file` to `dest_file` in the signal generator's memory catalog.

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Example

```
:MEM:MOV "NVWFM:Test_Data","NVWFM:New_Data"
```

The preceding example renames the file `Test_Data` to `New_Data` in the signal generator's non-volatile memory directory.

Key Entry **Rename File**

:STATe:COMMent

Supported All Models

```
:MEMory:STATe:COMMent <reg_num>,<seq_num>,"<comment>"
:MEMory:STATe:COMMent? <reg_num>,<seq_num>
```

This command lets you to add a descriptive comment to the saved instrument in the state register, <reg_num>,<seq_num>. Comments can be up to 55 characters long.

Example

```
:MEM:STAT:COMM 00,1, "ARB file using external reference"
```

The preceding example writes a descriptive comment to the state file saved in register 00, sequence 1.

Key Entry **Add Comment To Seq[n] Reg[nn]**

:STORe:LIST

Supported All Models

```
:MEMory:STORe:LIST "<file_name>"
```

This command stores the current list sweep data to a file.

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Example

```
:MEM:STOR:LIST "Test_Data"
```

The preceding example writes list sweep data to a file named Test_Data and stores the file in the signal generator's non–volatile memory, List directory.

Key Entry **Store To File**

Mass Memory Subsystem (:MMEMory)

:CATalog

Supported All Models

`:MMEMory:CATalog? "<msus>"`

This command outputs a list of the files from the specified file system. The variable `"<msus>"` (mass storage unit specifier) represents a file system. The file systems and types are shown in Table 2-1.

Table 2-1

File System	File Type
BIN - Binary file	BIN
BIT	BIT
DMOD - ARB digital modulation file	DMOD
FIR - finite impulse response filter file	FIR
FSK - frequency shift keying modulation file	FSK
I/Q - modulation file	IQ
LIST - sweep list file	LIST
MDMOD - ARB multicarrier digital modulation file	MDM
MTONE - ARB multitone file	MTON
NVMKR - non–volatile arbitrary waveform marker file	NVMKR
NVWFM - non–volatile arbitrary waveform file	NVWFM
SEQ - ARB sequence file	SEQ
SHAPE - burst shape file	SHAP
STATE	STAT
USERFLAT - user-flatness file	UFLT
WFM1 - waveform file	WFM1

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 specified file system. Each file listing will be in the following format:
"<file_name,file_type,file_size>"

Refer to "MSUS (Mass Storage Unit Specifier) Variable" on page 14 for information on the use of the "<msus>" variable.

Key Entry **Binary** **List** **State** **User Flatness** **FIR** **Shape** **Bit** **FSK**
I/Q **Seq** **DMOD** **MTONE** **MDMOD** **WFM1** **NVMKR** **NVWFM**

:COPY

Supported All Models

```
:MMEMory:COPY[:NAME] "<src_name>","<dest_name>"
```

This command copies the data from one file into another file. The file can use the same name if the specified directory is different. For example, if the file resides in non-volatile waveform memory (NVWFM) it can be copied, using the same name, to the signal generator's volatile memory (WFM1)

"<src_name>" This variable names a file residing in memory that will be copied. For information on the file name syntax, see "File Name Variables" on page 12

"<dest_name>" This variable names the file that is a copy of the "<src_name>" file.

Example

```
:MMEM:COPY "/USER/IQ/4QAM","/USER/IQ/test_QAM"
```

The preceding example copies the 4QAM file in the signal generator's /USER/IQ directory to a file named test_QAM and saves it in the same directory.

Key Entry **Copy File**

:DATA

Supported All Models

```
:MMEMory:DATA "<file_name>",<data_block>
:MMEMory:DATA? "<file_name>"
```

This command loads waveform data into signal generator memory using the `<data_block>` parameter and saves the data to a file designated by the `"<file_name>"` variable. The query returns the file contents of the file as a datablock.

The waveform file must be located in volatile waveform memory (WFM1) before it can be played by the signal generator's Dual ARB player. For downloads directly into volatile waveform memory use the path `"WFM1:<file_name>"`. For downloads to non-volatile waveform memory, use the path `"NVWFM:<file_name>"`.

Refer to "File Name Variables" on page 12 for information on the file name syntax.

`"<file_name>"` This variable names the destination file, including the directory path. Refer to "ARB Waveform File Directories" on page 13 for information on directory paths and the file name syntax.

`<data_block>` This parameter represents the data and file length parameters. The data in the file is represented by the `<data_block>` variable. The file length parameters are used by the signal generator for allocating memory.

Refer to the *PSG Programming Guide* for more information on downloading and using files.

NOTE Files created using the :DATA command cannot be retrieved or uploaded. Attempting to do so will cause the signal generator to display the message: `ERROR:221, Access denied`. To download data to files for later retrieval, use the :DATA:UNPRotected" command on page 59.

Example

```
:MMEM:DATA "NVWFM:IQ_Data",#210Qaz37pY9oL
```

The preceding example downloads 10 bytes of data to a file, `IQ_Data`., in the signal generator's non-volatile memory. The table shown below describes the command parameters.

• `"NVWFM:IQ_Data"`	`IQ_Data` is the data filename. The directory path is specified along with the filename
• `#210Qaz37pY9oL`	Data block
`#`	This character indicates the beginning of the data block
`2`	Number of digits in the byte count

• "NVWFM:IQ_Data"	IQ_Data is the data filename. The directory path is specified along with the filename
10	Byte count
Qaz37pY9oL	10 bytes of data

:DELete:NVWFm

Supported E8267D with Option 601or 602

`:MMEMory:DELete:NVWFm`

This command clears the memory file system of all non–volatile arbitrary waveform (NVWFM) files.

Key Entry **Delete All NVWFM Files**

:DELete:WFM

Supported E8267D with Option 601or 602

`:MMEMory:DELete:WFM`

This command clears the memory file system of all volatile arbitrary waveform (WFM1) files. It performs the same function as DELete:WFM1 command.

Key Entry **Delete All WFM1 Files**

:DELete[:NAME]

Supported All Models

`:MMEMory:DELete[:NAME] "<file_name>",["<msus>"]`

This command clears the memory file system of `"<file_name>"` with the option of specifying the file system `["<msus>"]` separately.

The variable `"<msus>"` (mass storage unit specifier) represents the file system. For a list of the file systems refer to Table 2-1 on page 66. Refer to “MSUS (Mass Storage Unit Specifier) Variable” on page 14 for information on the mass storage unit specifier.

If the optional variable `"<msus>"` is omitted, the file name needs to include the file system extension.

Refer to “File Name Variables” on page 12 for information on the file name syntax.

Example

```
:MMEM:DEL "/USER/BIN/Test_Data"
:MMEM:DEL "Test_Data",":BIN"
```

The preceding examples delete the file named `Test_Data` from the signal generator's USER/BIN directory. The first example uses the full file name path while the second example uses the `"<msus>"` specifier.

Key Entry **Delete File**

:HEADer:CLEar

Supported E8267D

```
:MMEMory:HEADer:CLEar "<file_name>"
```

This command deletes header file information for the waveform file `"<file_name>"`. This command does not require a personality modulation to be on. The header file contains signal generator settings and marker routings associated with the waveform file.

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Example

```
:MMEM:HEAD "/USER/WAVEFORM/Test_Data"
```

The preceding example deletes header file information for the `Test_Data` waveform file.

***RST** N/A

Key Entry **Clear Header**

:HEADer:DESCription

Supported E8267D

```
:MMEMory:HEADer:DESCription "<file_name>","<description>"
:MMEMory:HEADer:DESCription? "<file_name>"
```

This command inserts a description for the header file named. The header description is limited to 32 characters.

Refer to "File Name Variables" on page 12 for information on the file name syntax.

Example

```
:MMEM:HEAD:DESC "/USER/WAVEFORM/Test_Data","This is new header data"
```

The preceding example inserts a description into the `Test_Data` header file. In this example, the file is located in the signal generator's non–volatile waveform memory.

***RST** N/A

Key Entry **Edit Description**

:LOAD:LIST

Supported All Models

`:MMEMory:LOAD:LIST "<file_name>"`

This command loads list data from the List file `"<file_name>"`.

Refer to “File Name Variables” on page 12 for information on the file name syntax.

Example

`:MMEM:LOAD:LIST "Sweep_Data"`

The preceding example loads sweep configuration data from the `Sweep_Data` List file.

Key Entry **Load From Selected File**

:MOVE

Supported All Models

`:MMEMory:MOVE "<src_file>","<src_file_1>"`

This command renames the `src_file` to `src_file_1` in the memory catalog.

Refer to “File Name Variables” on page 12 for information on the file name syntax and using quotes for different programming languages.

Example

`:MMEM:MOV "NVWFM:Test_Data","NVWFM:New_Data"`

The preceding example renames the file `Test_Data` to `New_Data` in the signal generator’s non–volatile memory.

Key Entry **Rename File**

:STORe:LIST

Supported All Models

`:MMEMory:STORe:LIST "<file_name>"`

This command copies the current list sweep data to the `"<file_name>"` and saves it in the catalog of List files.

Refer to “File Name Variables” on page 12 for information on the file name syntax.

Example

```
:MMEM:STOR:LIST "Sweep_Data"
```

The preceding example stores the current list sweep data to the file `Sweep_Data` in the signal generator's catalog of List files.

Key Entry **Store To File**

Output Subsystem (:OUTPut)

:BLANking:AUTO

Supported All Models

```
[:SOURce]:OUTPut:BLANking:AUTO ON|OFF|1|0
[:SOURce]:OUTPut:BLANking:AUTO?
```

This command sets the state for automatic RF Output blanking. Blanking occurs when the RF output is momentarily turned off as the sweep transitions from one frequency segment (band) to another, allowing the signal to settle. Blanking also occurs during the retrace, so the signal can settle before the next sweep. In CW mode, blanking occurs whenever you change the frequency.

ON (1) This choice activates the automatic blanking function. The signal generator determines the blanking occurrences for optimum performance.

OFF (0) This choice turns off the automatic blanking function, which also sets the blanking state to off.

Example

```
:OUTP:BLAN:AUTO 0
```

The preceding example disables RF output blanking.

***RST** 1

Key Entry **Output Blanking Off On Auto**

:BLANking:[STATe]

Supported All Models

```
[:SOURce]:OUTPut:BLANking[:STATe] ON|OFF|1|0
[:SOURce]:OUTPut:BLANking[:STATe]?
```

This command sets the state for RF Output blanking. Blanking occurs when the RF output is momentarily turned off as the sweep transitions from one frequency segment (band) to another, allowing the signal to settle. Blanking also occurs during the retrace, so the signal can settle before the next sweep. In CW mode, blanking occurs whenever you change the frequency.

ON (1) This choice activates the blanking function. Blanking occurs on all frequency changes, including segment transitions and retrace

OFF (0) This choice turns off the blanking function.

Example

```
:OUTP:BLAN:ON
```

The preceding example enables RF output blanking.

Key Entry **Output Blanking Off On Auto**

:MODulation[:STATe]

Supported E8267D and E8257D with Option UNT

```
:OUTPut:MODulation[:STATe] ON|OFF|1|0
:OUTPut:MODulation[:STATe]?
```

This command enables or disables the modulation of the RF output with the currently active modulation type(s). Most modulation types can be simultaneously enabled except FM with ΦM.

An annunciator on the signal generator always displays to indicate whether modulation is on or off.

Example

```
:OUTP:MOD 0
```

The preceding example disables RF modulation.

***RST** 1

Key Entry **Mod On/Off**

[:STATe]

Supported All Models

```
:OUTPut[:STATe] ON|OFF|1|0
:OUTPut[:STATe]?
```

This command enables or disables the RF output. Although you can configure and engage various modulations, no signal is available at the RF OUTPUT connector until this command is executed.

An annunciator always displays on the signal generator to indicate whether the RF output is on or off.

Example

```
:OUTP ON
```

The preceding example turns on the signal generator’s RF output.

***RST** 0

Key Entry **RF On/Off**

Route Subsystem (:ROUTe:HARDware:DGENerator)

:INPut:BPOLarity

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:INPut:BPOLarity POSitive|NEGative
:ROUTe:HARDware:DGENerator:INPut:BPOLarity?
```

This command sets the signal generator up to respond to either a high (+5 vdc) or low (0 vdc) level TTL input signal at the BURST GATE IN connector. This command performs the same function as “:IPOLarity:BGATe” on page 77.

Example

```
:ROUT:HARD:DGEN:INP:BPOL NEG
```

The preceding example sets up the signal generator to respond to a LOW level TTL signal at the BURST GATE IN connector.

***RST** POS

Key Entry **Burst Gate In Polarity Neg Pos**

:INPut:CPOLarity

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:INPut:CPOLarity POSitive|NEGative
:ROUTe:HARDware:DGENerator:INPut:CPOLarity?
```

This command sets the signal generator up to respond to either a high (+5 vdc) or low (0 vdc) level TTL input signal at the DATA CLOCK input connector. This command performs the same function as “:IPOLarity:CLOCk” on page 77.

Example

```
:ROUT:HARD:DGEN:INP:CPOL POS
```

The preceding example sets up the signal generator to respond to a high level TTL signal at the DATA CLOCK input connector.

***RST** POS

Key Entry **Data Clock Polarity Neg Pos**

:INPut:DPOLarity

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:INPut:DPOLarity POSitive|NEGative
:ROUTe:HARDware:DGENerator:INPut:DPOLarity?
```

This command sets the signal generator up to respond to either a high (+5 vdc) or low (0 vdc) level TTL input signal at the DATA connector. This command performs the same function as ":IPOLarity:DATA" on page 78.

Example

```
:ROUT:HARD:DGEN:INP:DPOL POS
```

The preceding example sets up the signal generator to respond to a high level TTL signal at the DATA input connector.

***RST** POS

Key Entry **Data Polarity Neg Pos**

:INPut:SPOLarity

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:INPut:SPOLarity POSitive|NEGative
:ROUTe:HARDware:DGENerator:INPut:SPOLarity?
```

This command sets the signal generator up to respond to either a high (+5 vdc) or low (0 vdc) level TTL input signal at the SYMBOL SYNC input connector.

This command performs the same function as ":IPOLarity:SSYNc" on page 78.

Example

```
:ROUT:HARD:DGEN:INP:SPOL POS
```

The preceding example sets up the signal generator to respond to a high level TTL signal at the SYMBOL SYNC input connector.

***RST** POS

Key Entry **Symbol Sync Polarity Neg Pos**

:IPOLarity:BGATe

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:IPOLarity:BGATe POSitive|NEGative
:ROUTe:HARDware:DGENerator:IPOLarity:BGATe?
```

This command sets the signal generator up to respond to either a high (+5 vdc) or low (0 vdc) level TTL signal at the BURST GATE IN connector. This command performs the same function as “:INPut:BPOLarity” on page 75

Example

```
:ROUT:HARD:DGEN:IPOL:BGAT POS
```

The preceding example sets up the signal generator to respond to a high level TTL signal at the rear-panel BURST GATE IN connector.

***RST** POS

Key Entry **Burst Gate In Polarity Neg Pos**

:IPOLarity:CLOCk

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:IPOLarity:CLOCk POSitive|NEGative
:ROUTe:HARDware:DGENerator:IPOLarity:CLOCk?
```

This command sets the signal generator up to respond to either a high (+5 vdc) or low (0 vdc) level TTL input signal at the DATA CLOCK connector.

This command performs the same function as “:INPut:CPOLarity” on page 75.

Example

```
:ROUT:HARD:DGEN:IPOL:CLOC POS
```

The preceding example sets up the signal generator to respond to a high level TTL signal at the DATA CLOCK input connector.

***RST** POS

Key Entry **Data Clock Polarity Neg Pos**

:IPOLarity:DATA

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:IPOLarity:DATA POSitive|NEGative
:ROUTe:HARDware:DGENerator:IPOLarity:DATA?
```

This command sets the signal generator up to respond to either a high (+5 vdc) or low (0 vdc) level TTL input signal at the DATA connector. This command performs the same function as “:INPut:DPOLarity” on page 76

Example

```
:ROUT:HARD:DGEN:IPOL:DATA POS
```

The preceding example sets up the signal generator to respond to a high level TTL signal at the DATA input connector.

***RST** POS

Key Entry **Data Polarity Neg Pos**

:IPOLarity:SSYNc

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:IPOLarity:SSYNc POSitive|NEGative
:ROUTe:HARDware:DGENerator:IPOLarity:SSYNc?
```

This command sets the signal generator up to respond to either a high (+5 vdc) or low (0 vdc) level TTL input signal at the SYMBOL SYNC connector.

This command performs the same function as “:INPut:SPOLarity” on page 76.

Example

```
:ROUT:HARD:DGEN:IPOL:SSYN POS
```

The preceding example sets up the signal generator to respond to a high level TTL signal at the SYMBOL SYNC input connector.

***RST** POS

Key Entry **Symbol Sync Polarity Neg Pos**

:OPOLarity:CLOCk

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:OPOLarity:CLOCk POSitive|NEGative
:ROUTe:HARDware:DGENerator:OPOLarity:CLOCk?
```

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level TTL signal at the DATA CLK OUT pin on the rear panel AUXILIARY I/O connector.

This command performs the same function as ":OUTPut:CPOLarity" on page 80.

Example

```
:ROUT:HARD:DGEN:OPOL:CLOC POS
```

The preceding example sets up the signal generator to output a high level TTL signal at the DATA CLK OUT pin on the rear panel AUXILIARY I/O connector.

***RST** POS

Key Entry **Data Clock Out Neg Pos**

:OPOLarity:DATA

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:OPOLarity:DATA POSitive|NEGative
:ROUTe:HARDware:DGENerator:OPOLarity:DATA?
```

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level TTL signal at the DATA OUT pin on the rear panel AUXILIARY I/O connector.

This command performs the same function as ":OUTPut:DPOLarity" on page 81.

Example

```
:ROUT:HARD:DGEN:OPOL:DATA NEG
```

The preceding example sets up the signal generator to output a low level TTL signal at the DATA OUT pin on the rear panel AUXILIARY I/O connector.

***RST** POS

Key Entry **Data Out Polarity Neg Pos**

:OPOLarity:EVENt[1]|2|3|4

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:OPOLarity:EVENt[1]|2|3|4 POSitive|NEGative
:ROUTe:HARDware:DGENerator:OPOLarity:EVENt[1]|2|3|4?
```

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level TTL signal at the EVENT 1 or EVENT 2 connector.

This command performs the same function as ":OUTPut:EPOL[1]|2|3|4" on page 82.

Example

```
:ROUT:HARD:DGEN:OPOL:DATA NEG
```

The preceding example sets up the signal generator to output a low level TTL signal at the DATA OUT pin on the rear panel AUXILIARY I/O connector.

:OPOLarity:SSYNc

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:OPOLarity:SSYNc POSitive|NEGative
:ROUTe:HARDware:DGENerator:OPOLarity:SSYNc?
```

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level signal at the SYM SYNC OUT pin on the rear panel AUXILIARY I/O connector.

This command performs the same function as ":OUTPut:SPOLarity" on page 82.

Example

```
:ROUT:HARD:DGEN:OPOL:SSYN POS
```

The preceding example sets up the signal generator to output a high level TTL signal at the SYM SYNC OUT pin on the rear panel AUXILIARY I/O connector.

***RST** POS

Key Entry **Symbol Sync Out Polarity Neg Pos**

:OUTPut:CPOLarity

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:OUTPut:CPOLarity POSitive|NEGative
:ROUTe:HARDware:DGENerator:OUTPut:CPOLarity?
```

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level TTL signal at the DATA CLK OUT pin on the rear panel AUXILIARY I/O connector.

This command performs the same function as ":OPOLarity:CLOCk" on page 79.

Example

```
:ROUT:HARD:DGEN:OUTP:CPOL POS
```

The preceding example sets up the signal generator to output a high level TTL signal at the DATA CLOCK OUT pin on the rear panel AUXILIARY I/O connector.

***RST** POS

Key Entry **Data Clock Polarity Neg Pos**

:OUTPut:DCS[:STATe]

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:OUTPut:DCS[:STATe] ON|OFF|1|0
:ROUTe:HARDware:DGENerator:OUTPut:DCS[:STATe]?
```

This command is used to enable or disable the DATA OUT, DATA CLK OUT, and SYM SYNC OUT signals from the rear panel AUXILIARY I/O connector. Normally, these output signals should be enabled (On). However, disabling these outputs will decrease the spurs that are sometimes present when operating at high symbol rates.

Example

```
:ROUT:HARD:DGEN:OUTP:DCS 1
```

The preceding example sets up or enables the DATA OUT, DATA CLK OUT, and SYM SYNC OUT output signals from the rear panel AUXILIARY I/O connector.

***RST** 1

Key Entry **DATA/CLK/SYNC Rear Outputs Off On**

:OUTPut:DPOLarity

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:OUTPut:DPOLarity POSitive|NEGative
:ROUTe:HARDware:DGENerator:OUTPut:DPOLarity?
```

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level TTL signal at the DATA OUT connector.

This command performs the same function as ":OPOLarity:DATA" on page 79.

Example

`:ROUT:HARD:DGEN:OUTP:DPOL POS`

The preceding example sets up the signal generator to output a high level TTL signal at the DATA OUT connector.

***RST** POS

Key Entry **Data Out Polarity Neg Pos**

:OUTPut:EPOL[1]|2|3|4

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:OUTPut:EPOLarity[1]|2|3|4 POSitive|NEGative
:ROUTe:HARDware:DGENerator:OUTPut:EPOLarity[1]|2|3|4?
```

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level TTL signal at the EVENT1 or EVENT 2 connector.

This command performs the same function as ":OPOLarity:EVENt[1]|2|3|4" on page 80.

Example

`:ROUT:HARD:DGEN:OUTP:EPOL1` POS

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level TTL signal at the EVENT1 or EVENT 2 connector.

:OUTPut:SPOLarity

Supported E8267D with Option 601or 602

```
:ROUTe:HARDware:DGENerator:OUTPut:SPOLarity POSitive|NEGative
:ROUTe:HARDware:DGENerator:OUTPut:SPOLarity?
```

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level TTL signal at the SYMBOL SYNC connector.

Example

`:ROUT:HARD:DGEN:OUTP:SPOL` POS

This command sets the signal generator up to output either a high (+5 vdc) or low (0 vdc) level TTL signal at the EVENT1 or EVENT 2 connector.

***RST** POS

Key Entry **Symbol Sync Out Polarity Neg Pos**

Status Subsystem (:STATus)

:OPERation:BASeband:CONDition

Supported E8267D with Option 601or 602

`:STATus:OPERation:BASeband:CONDition?`

This query returns the decimal sum of the bits in the Baseband Operation Condition register. For example, if the baseband is busy (bit 0), the value 1 is returned.

The data in this register is continuously updated and reflects current signal generator conditions.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:OPERation:BASeband:ENABle

Supported E8267D with Option 601or 602

```
:STATus:OPERation:BASeband <val>
:STATus:OPERation:BASeband:ENABle?
```

This command enables bits in the Baseband Operation Event Enable register. Bits enabled and set in this register will set bit 10 in the Standard Operation Condition register.

The variable `<val>` is the sum of the decimal values of the bits you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

`:STAT:OPER:BAS:ENAB 3`

This command enables bit 0 (decimal 1, Baseband is Busy) and bit 1 (decimal 2, Baseband 1 Communicating) in the Baseband Operation Event Enable register.

Range 0–32767

:OPERation:BASeband:NTRansition

Supported E8267D with Option 601or 602

```
:STATus:OPERation:BASeband:NTRansition <val>
:STATus:OPERation:BASeband:NTRansition?
```

This command enables bits in the Baseband Operation Negative Transition Filter register. A negative transition (1 to 0) of corresponding bits in the Baseband Operation Condition register will pass through and be read by the Baseband Operation Event register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:OPER:BAS:NTR 3
```

This command enables bit 0 (decimal 1, Baseband 1 Busy) and bit 1 (decimal 2, Baseband 1 Communicating) in the Baseband Operation Negative Transition Filter register.

Range 0–32767

:OPERation:BASeband:PTRansition

Supported E8267D with Option 601or 602

```
:STATus:OPERation:BASeband:PTRansition <val>
:STATus:OPERation:BASeband:PTRansition?
```

This command enables bits in the Baseband Operation Positive Transition Filter register. A positive transition (0 to 1) of corresponding bits in the Baseband Operation Condition register will pass through and be read by the Baseband Operation Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:OPER:BAS:PTR 3
```

This command enables bit 0 (decimal 1, Baseband 1 Busy) and bit 1 (decimal 2, Baseband 1 Communicating) in the Baseband Operation Positive Transition Filter register.

Range 0–32767

:OPERation:BASeband[:EVENt]

Supported E8267D with Option 601or 602

`:STATus:OPERation:BASeband[:EVENt]?`

NOTE This is a destructive read. The data in the Baseband Operation Event register is latched until it is queried. Once queried, the data is cleared.

This query returns the decimal sum of the bits in the Baseband Operation Event register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:OPERation:CONDition

Supported All Models

`:STATus:OPERation:CONDition?`

This query returns the decimal sum of the bits in the Standard Operation Condition register. This register monitors signal generator functions such as I/Q calibrating, sweeping, and measuring. For example, if a sweep is in progress (bit 3), a decimal 8 is returned with this query.

The data in this register is continuously updated and reflects current conditions.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:OPERation:ENABle

Supported All Models

`:STATus:OPERation:ENABle <val>`
`:STATus:OPERation:ENABle?`

This command enables bits in the Standard Operation Event Enable register. Bits enabled and set in this register will set the Operation Status Summary bit (bit 7) in the Status Byte register. When bit 7 in the Status Byte register is set, you can read the Standard Operation Event register to determine the cause.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:OPER:ENAB 43
```

This command enables bit 0 (decimal 1, I/Q calibrating), bit 1 (decimal 2, Settling), bit 3 (decimal 8, Sweeping), and bit 5 (decimal 32, Waiting for Trigger) of the Standard Operation Event Enable register.

Range 0–32767

:OPERation:NTRansition

Supported All Models

```
:STATus:OPERation:NTRansition <val>
:STATus:OPERation:NTRansition?
```

This command enables bits in the Standard Operation Negative Transition Filter register. A negative transition (1 to 0) of corresponding bits in the Standard Operation Condition register will pass through and be read by the Standard Operation Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:OPER:NTR 3
```

This command enables bit 0 (decimal 1, I/Q Calibrating) and bit 1 (decimal 2, Settling) in the Standard Operation Negative Transition Filter register.

Range 0–32767

:OPERation:PTRansition

Supported All Models

```
:STATus:OPERation:PTRansition <val>
:STATus:OPERation:PTRansition?
```

This command enables bits in the Standard Operation Positive Transition Filter register. A positive transition (0 to 1) of corresponding bits in the Standard Operation Condition register will pass through and be read by the Standard Operation Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:OPER:PTR 3
```

This command enables bit 0 (decimal 1, I/Q Calibrating) and bit 1 (decimal 2, Settling) in the Standard Operation Positive Transition Filter register.

Range 0–32767

:OPERation[:EVENt]

Supported All Models

NOTE This is a destructive read. The data in the register is latched until it is queried. Once queried, the data is cleared.

```
:STATus:OPERation[:EVENt]?
```

This query returns the decimal sum of the bits in the Standard Operation Event register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:PRESet

Supported All Models

```
:STATus:PRESet
```

This command presets all positive and negative transition filters, enable registers, and error/event queue enable registers.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

:QUEStionable:CALibration:CONDition

Supported All Models

```
:STATus:QUEStionable:CALibration:CONDition?
```

This query returns the decimal sum of the bits in the Data Questionable Calibration Condition register. For example, if the DCFM or DCΦM zero calibration fails (bit 0), a value of 1 is returned.

The data in this register is continuously updated and reflects the current conditions.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:QUEStionable:CALibration:ENABle

Supported All Models

```
:STATus:QUEStionable:CALibration:ENABle <val>
:STATus:QUEStionable:CALibration:ENABle?
```

This command enables bits in the Data Questionable Calibration Event Enable register. Bits enabled and set in this register will set the Calibration Summary bit (bit 8) in the Data Questionable Condition register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:QUES:CAL:ENAB 1
```

This command enables bit 0 (decimal 1, DCFM/DCΦM Zero Failure) in the Data Questionable Calibration Event Enable register.

Range 0–32767

:QUEStionable:CALibration:NTRansition

Supported All Models

```
:STATus:QUEStionable:CALibration:NTRansition <val>
:STATus:QUEStionable:CALibration:NTRansition?
```

This command enables bits in the Data Questionable Calibration Negative Transition Filter register. A negative transition (1 to 0) of corresponding bits in the Data Questionable Calibration Condition register will pass through and be read by the Data Questionable Calibration Event register

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:OPER:NTR 3
```

This command enables bit 0 (decimal 1, DCFM/DCΦM Zero Failure) and bit 1 (decimal 2, I/Q Calibration Failure) in the Data Questionable Calibration Negative Transition Filter register.

Range 0–32767

:QUEStionable:CALibration:PTRansition

Supported All Models

```
:STATus:QUEStionable:CALibration:PTRansition <val>
:STATus:QUEStionable:CALibration:PTRansition?
```

This command enables bits in the Data Questionable Calibration Positive Transition Filter register. A positive transition (0 to 1) of corresponding bits in the Data Questionable Calibration Condition register will pass through and be read by the Data Questionable Calibration Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:OPER:PTR 3
```

This command enables bit 0 (decimal 1, DCFM/DCΦM Zero Failure) and bit 1 (decimal 2, I/Q Calibration Failure) in the Data Questionable Calibration Positive Transition Filter register.

Range 0–32767

:QUEStionable:CALibration[:EVENt]

Supported All Models

NOTE This is a destructive read. The data in the register is latched until it is queried. Once queried, the data is cleared.

```
:STATus:QUEStionable:CALibration[:EVENt]?
```

This command returns the decimal sum of the bits in the Data Questionable Calibration Event register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:QUEStionable:CONDition

Supported All Models

```
:STATus:QUEStionable:CONDition?
```

This query returns the decimal sum of the bits in the Data Questionable Condition register. For example, if the internal reference oscillator oven is cold (bit 4), a value of 16 is returned.

The data in this register is continuously updated and reflects current conditions.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:QUEStionable:ENABle

Supported All Models

```
:STATus:QUEStionable:ENABle <val>
:STATus:QUEStionable:ENABle?
```

This command enables bits in the Data Questionable Event Enable register. Bits enabled and set in this register will set the Data Questionable Summary bit (bit 3) in the Status Byte register. When bit 3 in the Status Byte register is set, you can read the Data Questionable Event register to determine the cause.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:QUES:ENAB 8
```

This command enables bit 3 (decimal 8, the Power Summary bit), in the Data Questionable Event Enable register.

Range 0–32767

:QUEStionable:FREQuency:CONDition

Supported All Models

```
:STATus:QUEStionable:FREQuency:CONDition?
```

This query returns the decimal sum of the bits in the Data Questionable Frequency Condition register. For example, if the 1 GHz internal reference clock is unlocked (bit 2), a value of 4 is returned.

The data in this register is continuously updated and reflects current conditions.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:QUEStionable:FREQuency:ENABle

Supported All Models

```
:STATus:QUEStionable:FREQuency:ENABle <val>
:STATus:QUEStionable:FREQuency:ENABle?
```

This command enables bits in the Data Questionable Frequency Event Enable register. Bits enabled and set in this register will set the Data Questionable Condition register bit 5.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:QUES:FREQ:ENAB 7
```

This command enables bit 0 (decimal 1, Synthesizer Unlocked), bit 1 (decimal 2, 10 MHz Reference Unlocked), and bit 2 (decimal 4, 1 GHz reference Unlocked) in the Data Questionable Frequency Event Enable register.

Range 0–32767

:QUEStionable:FREQuency:NTRansition

Supported All Models

```
:STATus:QUEStionable:FREQuency:NTRansition <val>
:STATus:QUEStionable:FREQuency:NTRansition?
```

This command enables bits in the Data Questionable Frequency Negative Transition Filter register. A negative transition (1 to 0) of corresponding bits in the Data Questionable Frequency Condition register will pass through and be read by the Data Questionable Frequency Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:QUES:FREQ:NTR 96
```

This command enables bit 5 (decimal 32, Sampler Loop Unlocked) and bit 6 (decimal 64, YO Loop Unlocked) in the Data Questionable Frequency Negative Transition Filter register.

Range 0–32767

:QUEStionable:FREQuency:PTRansition

Supported All Models

```
:STATus:QUEStionable:FREQuency:PTRansition <val>
:STATus:QUEStionable:FREQuency:PTRansition?
```

This command enables bits in the Data Questionable Frequency Positive Transition Filter register. A positive transition (0 to 1) of corresponding bits in the Data Questionable Frequency Condition register will pass through and be read by the Data Questionable Frequency Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

`:STAT:QUES:FREQ:PTR 8`

This command enables bit 3 (decimal 8, Baseband 1 Unlocked) in the Data Questionable Frequency Positive Transition Filter register.

Range 0–32767

:QUEStionable:FREQuency[:EVENt]

Supported All Models

CAUTION This is a destructive read. The data in the register is latched until it is queried. Once queried, the data is cleared.

`:STATus:QUEStionable:FREQuency[:EVENt]?`

This query returns the decimal sum of the bits in the Data Questionable Frequency Event register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:QUEStionable:MODulation:CONDition

Supported All Models

`:STATus:QUEStionable:MODulation:CONDition?`

This command returns the decimal sum of the bits in the Data Questionable Modulation Condition register. For example, if the modulation is uncalibrated (bit 4), a value of 16 is returned.

The data in this register is continuously updated and reflects current conditions.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:QUEStionable:MODulation:ENABle

Supported All Models

```
:STATus:QUEStionable:MODulation:ENABle <val>
:STATus:QUEStionable:MODulation:ENABle?
```

This command enables bits in the Data Questionable Modulation Event Enable register. Bits enabled and set in this register will set bit 7 in the Data Questionable Condition register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:QUES:MOD:ENAB 20
```

This command enables bit 2 (decimal 4, Modulation 1 Overmod) and bit 4 (decimal 16, Modulation Uncalibrated) in the Data Questionable Modulation Event Enable register.

Range 0–32767

:QUEStionable:MODulation:NTRansition

Supported All Models

```
:STATus:QUEStionable:MODulation:NTRansition <val>
:STATus:QUEStionable:MODulation:NTRansition?
```

This command enables bits in the Modulation Questionable Negative Transition Filter register. A negative transition (1 to 0) of corresponding bits in the Modulation Questionable Condition register will pass through and be read by the Modulation Questionable Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:QUES:MOD:NTR 3
```

This command enables bit 0 (decimal 1, Modulation 1 Undermod) and bit 1 (decimal 2, Modulation 1 Overmod) in the Data Questionable Modulation Negative Transition Filter register.

Range 0–32767

:QUEStionable:MODulation:PTRansition

Supported All Models

```
:STATus:QUEStionable:MODulation:PTRansition <val>
:STATus:QUEStionable:MODulation:PTRansition?
```

This command enables bits in the Data Questionable Modulation Positive Transition Filter register. A positive transition (0 to 1) of corresponding bits in the Data Questionable Modulation Condition register will pass through and be read by the Data Questionable Modulation Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:QUES:MOD:PTR 3
```

This command enables bit 0 (decimal 1, Modulation 1 Undermod) and bit 1 (decimal 2, Modulation 1 Overmod) in the Data Questionable Modulation Positive Transition Filter register.

Range 0–32767

:QUEStionable:MODulation[:EVENt]

Supported All Models

CAUTION This is a destructive read. The data in the register is latched until it is queried. Once queried, the data is cleared.

```
:STATus:QUEStionable:MODulation[:EVENt]?
```

This query returns the decimal sum of the bits in the Data Questionable Modulation Event register.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:QUEStionable:NTRansition

Supported All Models

```
:STATus:QUEStionable:NTRansition <val>
:STATus:QUEStionable:NTRansition?
```

This command enables bits in the Data Questionable Negative Transition Filter register. A negative transition (1 to 0) of corresponding bits in the Data Questionable Condition register will pass through and be read by the Data Questionable Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:QUES:MOD:NTR 3072
```

This command enables bit 10 (decimal 1024, Baseband is busy) and bit 11 (decimal 2048, Sweep Calculating) in the Data Questionable Negative Transition Filter register.

Range 0–32767

:QUEStionable:POWer:CONDition

Supported All Models

```
:STATus:QUEStionable:POWer:CONDition?
```

This query returns the decimal sum of the bits in the Data Questionable Power Condition register. For example, if the RF output signal is unleveled (bit 1), a value of 2 is returned.

The data in this register is continuously updated and reflects current conditions.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Range 0–32767

:QUEStionable:POWer:ENABle

Supported All Models

```
:STATus:QUEStionable:POWer:ENABle <val>
:STATus:QUEStionable:POWer:ENABle?
```

This command enables bits in the Data Questionable Power Event Enable register. Bits enabled and set in this register will set bit 3 in the Data Questionable Condition register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

Refer to the *PSG Programming Guide* for more information on programming the status registers.

Example

```
:STAT:QUES:POW:ENAB 1
```

This command enables bit 0 (decimal 1, Reverse Power Protection Tripped) in the Data Questionable Power Event Enable register.

Range 0–32767

:QUEStionable:POWer:NTRansition

Supported All Models

```
:STATus:QUEStionable:POWer:NTRansition <val>
:STATus:QUEStionable:POWer:NTRansition?
```

This command enables bits in the Data Questionable Power Negative Transition Filter register. A negative transition (1 to 0) of corresponding bits in the Data Questionable Power Condition register will pass through and be read by the Data Questionable Power Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

See the *PSG Programming Guide* for more information on programming the status register system.

Example

```
:STAT:QUES:POW:NTR 1
```

This command enables bit 0 (Reverse Power Protection Tripped) in the Data Questionable Power Negative Transition Filter register.

Range 0–32767

:QUEStionable:POWer:PTRansition

Supported All Models

```
:STATus:QUEStionable:POWer:PTRansition <val>
:STATus:QUEStionable:POWer:PTRansition?
```

This command enables bits in the Data Questionable Power Positive Transition Filter register. A positive transition (0 to 1) of corresponding bits in the Data Questionable Power Condition register will pass through and be read by the Data Questionable Power Event register.

The variable `<val>` is the sum of the decimal values of the bits that you want to enable.

See the *PSG Programming Guide* for more information on programming the status register system.

Example

```
:STAT:QUES:POW:PTR 1
```

This command enables bit 0 (decimal 1, Reverse Power Protection Tripped) in the Data Questionable Power Positive Transition Filter register.

Range 0–32767

:QUEStionable:POWer[:EVENt]

Supported All Models

CAUTION This is a destructive read. The data in the register is latched until it is queried. Once queried, the data is cleared.

```
:STATus:QUEStionable:POWer[:EVENt]?
```

This query returns the decimal sum of the bits in the Data Questionable Power Event register.

See the *PSG Programming Guide* for more information on programming the status register system.

Range 0–32767

:QUEStionable:PTRansition

Supported All Models

```
:STATus:QUEStionable:PTRansition <val>
:STATus:QUEStionable:PTRansition?
```

This command enables bits in the Data Questionable Positive Transition Filter register. A positive transition (0 to 1) of corresponding bits in the Data Questionable Condition register will pass through and be read by the Data Questionable Event register.

See the *PSG Programming Guide* for more information on programming the status register system.

Example

```
:STAT:QUES:PTR 8
```

This command enables bit 3 (decimal 8, Power Summary) in the Data Questionable Positive Transition Filter register.

Range 0–32767

:QUEStionable[:EVENt]

Supported All Models

CAUTION This is a destructive read. The data in the register is latched until it is queried. Once queried, the data is cleared.

```
:STATus:QUEStionable[:EVENt]?
```

This query returns the decimal sum of the bits in the Standard Operation Event register.

See the *PSG Programming Guide* for more information on programming the status register system.

Range 0–32767

System Subsystem (:SYSTem)

:ALTernate

Supported All Models with Option 007

```
:SYSTem:ALTernate <reg_num>
:SYSTem:ALTernate? [MAXimum|MINimum]
```

This command sets up the signal generator to use a sweep state stored in a state register to alternate with the current sweep. The alternate sweep state must be stored in state registers 1 through 9 in sequence 0. Alternate sweep must be selected and both sweeps must be ramp sweeps.

Example

```
:SYST:ALT 3
```

The preceding example alternates the current sweep with the sweep settings saved in state register number three.

Key Entry **Alternate Sweep Seq 0, Register 1–9**

:ALTernate:STATe

Supported All Models with Option 007

```
:SYSTem:ALTernate:STATe ON|OFF|1|0
:SYSTem:STATe?
```

This command enables or disables the alternate sweep state for the signal generator. With alternate state on, the signal generator uses the current sweep setup and alternates with a sweep saved in on of the state registers. Both sweeps must be ramp sweeps.

Example

```
:SYST:ALT:STAT OFF
```

The preceding example disables the alternate sweep mode.

Key Entry **Alternate Sweep Off On**

:CAPability

Supported All Models

```
:SYSTem:CAPability?
```

This query returns the signal generator’s capabilities and outputs the appropriate specifiers:
`(RFSOURCE WITH((AM|FM|PULM|PM|LFO)&(FSSWEEP|FLIST)&(PSSWEEP|PLIST)`
`&TRIGGER&REFERENCE))`

This is a list of the SCPI-defined basic functionality of the signal generator and the additional capabilities it has in parallel (a&b) and singularly (a|b).

:DATE

Supported All Models

```
:SYSTem:DATE <year>,<month>,<day>
:SYSTem:DATE?
```

This command sets the date as shown in the lower right area of the signal generator display.

`<year>` This variable requires a four digit integer.

The query returns the date in the following format: <+year>, <+month>, <+day>

Range *<month>*: 1–12 *<day>*: 1–31

Example

```
:SYST:DATE 2004,12,15
```

The preceding example sets the date.

Key Entry **Time / Date**

:ERRor[:NEXT]

Supported All Models

```
:SYSTem:ERRor[:NEXT]?
```

This query returns the most recent error message from the signal generator error queue. If there are no error messages, the query returns the following output:

```
+0,"No error"
```

When there is more than one error message, the query will need to be sent for each message.

The error messages are erased after being queried.

Key Entry **Error Info** **View Next Error Message**

:ERRor:SCPI[:SYNTax]

Supported All

```
:SYSTem:ERRor:SCPI[:SYNTax] ON|OFF|1|0
:SYSTem:ERRor:SCPI[:SYNTax]?
```

This command allows you to turn on verbose error messages that point out where the SCPI parser generated an error. Use the `ERRor[:NEXT]` command to read any reported errors.

Example

```
:SYST:ERR:SCPI ON
```

The preceding example enables the SCPI command error report function.

***RST** 1

:FILEsystem:SAFEmode

Supported All

```
:SYSTem:FILEsystem:SAFEmode ON|OFF|1|0
:SYSTem:FILEsystem:SAFEmode?
```

This command selects the safe mode for file handling. When safe mode is set to `OFF`, volatile waveform files can be edited and saved while the signal generator plays the file without signal interruption. However, it is possible with complex waveforms, for corruption of memory to occur which will be reported as an error on the front-panel display and require a reboot of the signal generator to resolve.

Example

```
:SYST:FILE:SAVE ON
```

The preceding example enables the safe mode setting and waveform files cannot be edited without signal disruption while the signal generator plays them.

***RST** On

:HELP:MODE

Supported All Models

```
:SYSTem:HELP:MODE SINGle|CONTinuous
:SYSTem:HELP:MODE?
```

This command sets the help function mode of the signal generator.

SINGle Help is provided only for the next key that you press.

CONTinuous	Help is provided for each key you press. In addition, the function of the key is executed.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:SYST:HELP:MODE CONT
```

The preceding example enables the Help system continuos mode.

Key Entry **Help Mode Single Cont**

:IDN

Supported All Models

```
:SYSTem:IDN "string"
```

This command modifies the identification string that the `*IDN?` query returns. Sending an empty string returns the query output of `*IDN?` to its factory-shipped setting. The maximum string length is 72 characters.

Modification of the *IDN? query output enables the signal generator to identify itself as another signal generator when used as a replacement.

The display diagnostic information, shown by pressing the **Diagnostic Info** softkey, is not affected by this command.

:LANGuage

Supported All Models

```
:SYSTem:LANGuage "SCPI"|"8340"|"8360"|"83712"|"83732"|"83752"|"8757"
:SYSTem:LANGuage?
```

This command sets the remote language for the signal generator.

SCPI	This choice provides compatibility for SCPI commands.
8340	This choice provides compatibility for 8340B and 8341B microwave sources, which are supported by using the GPIB interface.
8360	This choice provides compatibility for 8360 series swept signal generators, which are supported only through a GPIB interface.
83712	This choice provides compatibility for 83711B and 83712B synthesized CW generators, which are supported only through a GPIB interface.
83732	This choice provides compatibility for 83731B and 83732B synthesized signal

generators, which are supported only through a GPIB interface.

83752	This choice provides compatibility for 83751B and 83752B synthesized sweepers, which are supported only through a GPIB interface.
8757	This choice provides compatibility for a system, comprising a PSG signal generator and a 8757D scalar network analyzer. It is supported only through a GPIB interface.
8662	This choice provides compatibility for the Agilent 8662A Synthesized Signal Generator. The 8662A is controlled only through a GPIB interface.
8663	This choice provides compatibility for the Agilent 8663A Synthesized Signal Generator. The 8663A is controlled only through a GPIB interface.

The setting enabled by this command is not affected by a power-on, preset, or `*RST` command.

For more information on supported SCPI commands and programming codes, refer to Chapter 7, "SCPI Command Compatibility," on page 375.

Example

```
:SYST:LANG "8757"
```

The preceding example enables the 8757 Network Analyzer language as the language used to control the signal generator.

Key Entry	**SCPI**	**8360 Series**	**83711B,83712B**	**8757D System**
	83731B,83732B	**8340B,8341B**	**83751B,83752B**	**8662A,8663A**

:OEMHead:FREQuency:STARt

Supported All

```
:SYSTem:OEMHead:FREQuency:STARt <val>
:SYSTem:OEMHead:FREQuency:STARt?
```

This command sets the start frequency or minimum band frequency for an external source module. The pre-defined start or minimum band frequency for the selected WR (waveguide rectangular) is overwritten with this command. For more information on pre-defined frequency bands, refer to ":OEMHead:FREQuency:BAND WR15|WR12|WR10|WR8|WR6|WR5|WR3" on page 103.

Example

```
:SYST:OEMH:FREQ:STAR 90GHZ
```

The preceding example sets the start frequency for the OEM module to 90 GHz.

***RST** 5.0000000000000E+10

Key Entry **Min Band Freq**

:OEMHead:FREQuency:STOP

Supported All

```
:SYSTem:OEMHead:FREQuency:STOP <val>
:SYSTem:OEMHead:FREQuency:STOP?
```

This command sets the stop frequency or maximum band frequency for an external source module. The pre-defined stop or maximum band frequency for the selected WR (waveguide rectangular) is overwritten with this command. For more information on pre-defined frequency bands, refer to ":OEMHead:FREQuency:BAND WR15|WR12|WR10|WR8|WR6|WR5|WR3" on page 103.

Example

```
:SYST:OEMH:FREQ:STOP 70GHZ
```

The preceding example sets the stop frequency for the OEM module to 70 GHz.

***RST** 7.0000000000000E+10

Key Entry **Max Band Freq**

:OEMHead:SELect

Supported All

```
:SYSTem:OEMHead:SELect ON|OFF|NONE|REAR|FRONT
:SYSTem:OEMHead:SELect?
```

This command selects an external millimeter-wave source module. The ON, REAR, and FRONT parameters select the OEM source module while the OFF and NONE parameters deselect the OEM source module. The `MMOD` and `MULT` annunciators, in the signal generator's frequency display will appear when a OEM millimeter-wave source module is selected.

Example

```
:SYST:OEMH:SEL ON
```

The preceding example turns on the OEM source module.

***RST** Off

Key Entry **OEM Source Module Off On**

:OEMHead:FREQuency:BAND WR15|WR12|WR10|WR8|WR6|WR5|WR3

Supported All

```
:SYSTem:OEMHead:FREQuency:BAND WR3
:SYSTem:OEMHead:FREQuency:BAND?
```

This command allows you to select a pre-defined waveguide rectangular (WR) frequency band. The WR selection is determined by the external millimeter-wave source module frequency range. Selection of a WR frequency band sets the minimum and maximum frequency bands, for the external mm-wave source module, to pre-defined values shown in the table below. These pre-defined frequency bands are common to commercially available mixers and multipliers. Different start, stop, and multiplier values can be selected from the menu displayed under the **OEM Source Module Config** softkey.

Table 2-2

Waveguide Band	**PSG Start Frequency**	**PSG Stop Frequency**	**Multiplier**
WR15 50–75GHz	12.5000000000 GHz	18.7500000000 GHz	4.000 x
WR12 60–90GHz	10.0000000000 GHz	15.0000000000 GHz	6.000 x
WR10 75–110GHz	12.5000000000 GHz	18.4000000000 GHz	6.000 x
WR8 90–140GHz	11.2200000000 GHz	17.5000000000 GHz	8.000 x
WR6 110–170GHz	9.1000000000 GHz	14.2000000000 GHz	12.000 x
WR5 140–220GHz	11.6000000000 GHz	18.4000000000 GHz	12.000 x
WR3 220–325GHz	12.2000000000 GHz	18.1000000000 GHz	18.000 x

Example

```
:SYST:OEMH:FREQ:BAND WR12
```

The preceding example selects the 60-90 GHz WR frequency band.

***RST** WR15

Key Entry **WR15 50-75GHz**

:OEMHead:FREQuency:MULTiplier

Supported All

```
:SYSTem:OEMHead:FREQuency:MULTiplier <val>
:SYSTem:OEMHead:FREQuency:MULTiplier?
```

This command allows you to select a multiplier for an external millimeter-wave source module. The multiplier factor allows the signal generator’s frequency display to show the source module’s frequency. The selection is valid only when the OEM source module is selected.

The signal generator’s actual RF frequency is not changed by the multiplier. For example, if the signal generator’s RF frequency is 20 GHz and a 4.000 x multiplier is selected, the signal generator will display 80 GHz.

The displayed frequency on the signal generator is affected if the frequency reference and frequency offset settings. The relationship is described as follows: Displayed Frequency = (Actual Freq – Freq Reference)* Frequency Multiplier + Freq Offset. Refer to the “:FREQuency:OFFSet” on page 130 and “:FREQuency:REFerence” on page 131 command descriptions for more information.

Example

```
:SYST:OEMH:FREQ:MULT 4
```

The preceding example selects a 4x multiplier so that the signal generator display shows the frequency at the output of the mm-wave source module.

***RST** 4.00000000E+000

Key Entry **Freq Multiplier**

:PON:TYPE

Supported All Models

```
:SYSTem:PON:TYPE PRESet|LAST
:SYSTem:PON:TYPE?
```

This command sets the defined conditions for the signal generator at power on.

PRESet This choice sets the conditions to factory- or user-defined as determined by the choice for the preset type. Refer to “:PRESet:TYPE” on page 108 for selecting the type of preset.

LAST This choice retains the settings at the time the signal generator was last powered down.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

NOTE When LAST is selected, no signal generator interaction can occur for at least 3 seconds prior to cycling the power for the current settings to be saved.

Example

```
:SYST:PON:TYPE PRES
```

The preceding example sets the preset state for the signal generator to factory settings.

Key Entry **Power On Last Preset**

:PRESet

Supported All Models

`SYSTem:PRESet`

This command returns the signal generator to a set of defined conditions. It is equivalent to pressing the front panel **Preset** hardkey.

The defined conditions are either factory– or user–defined. Refer to ":PRESet:TYPE" on page 108 for selecting the type of defined conditions.

Key Entry **Preset**

:PRESet:ALL

Supported All Models

`:SYSTem:PRESet:ALL`

This command sets all states of the signal generator back to their factory default settings, including states that are not normally affected by a signal generator power-on, preset, or `*RST` command.

:PRESet:LANGuage

Supported All Models

```
:SYSTem:PRESet:LANGuage "SCPI"|"8340"|"8360"|"83712"|"83732"|"83752"|
"8757"
:SYSTem:PRESet:LANGuage?
```

This command sets the remote language that is available when the signal generator is preset.

SCPI	This choice provides compatibility for SCPI commands.
8340	This choice provides compatibility for 8340B and 8341B microwave sources, which are supported by using the GPIB interface.
8360	This choice provides compatibility for 8360 series swept signal generators, which are supported only through a GPIB interface.
83712	This choice provides compatibility for 83711B and 83712B synthesized CW generators, which are supported only through a GPIB interface.
83732	This choice provides compatibility for 83731B and 83732B synthesized signal generators, which are supported only through a GPIB interface.
83752	This choice provides compatibility for 83751B and 83752B synthesized sweepers, which are supported only through a GPIB interface.

8757	This choice provides compatibility for a system, comprising a PSG signal generator and a 8757D scalar network analyzer. It is supported only through a GPIB interface.
8662	This choice provides compatibility for 8662A series synthesized waveform generators, which are supported only through a GPIB interface.
8663	This choice provides compatibility for 8663A series synthesized waveform generators, which are supported only through a GPIB interface.

Example

```
:SYST:PRES:LANG "8340"
```

The preceding example selects 8340 signal generator language as the language used by the signal generator following an instrument preset.

***RST** "SCPI"

Key Entry	**SCPI**	**8360 Series**	**83711B,83712B**	**8757D System**
	83731B,83732B	**8340B,8341B**	**83751B,83752B**	

:PRESet:PERSistent

Supported All Models

```
:SYSTem:PRESet:PERSistent
```

This command sets the states that are not affected by a signal generator power-on, preset, or `*RST` command to their factory default settings.

Key Entry **Restore Sys Defaults**

:PRESet:PN9

Supported All Models

```
:SYSTem:PRESet:PN9 NORMal|QUICk
:SYSTem:PRESet:PN9?
```

This command sets the preset length of the PN9 sequence for personalities that require software PRBS generation.

NORMal	This choice produces a maximal length PN9 sequence.
QUICk	This choice produces a truncated (216 bits) PN9 sequence.

Example

```
:SYST:PRES:PN9 NORMAL
```

The preceding example selects a maximum length PN9 sequence.

***RST** NORM

Key Entry **PN9 Mode Preset Normal Quick**

:PRESet:TYPE

Supported All Models

```
:SYSTem:PRESet:TYPE NORMal|USER
:SYSTem:PRESet:TYPE?
```

This command toggles the preset state between factory- and user-defined conditions. Refer to “:PRESet[:USER]:SAVE” for saving the USER choice preset settings. The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:SYST:PRES:TYPE USER
```

The preceding example selects a user defined conditions for the signal generator preset state.

Key Entry **Preset Normal User**

:PRESet[:USER]:SAVE

Supported All Models

```
:SYSTem:PRESet[:USER]:SAVE
```

This command saves your user-defined preset conditions to a state file.

Only one user-defined preset file can be saved. Subsequent saved user–defined preset files will overwrite the previously saved file.

Key Entry **Save User Preset**

:SECurity:DISPlay

Supported All Models

```
:SYSTem:SECurity:DISPlay ON|OFF|1|0
:SYSTem:SECurity:DISPlay?
```

This command enables or disables the secure display mode.

On(1)	This selection turns the signal generator display back on, showing the current settings. Cycling the signal generator power also restores the display, however the current settings may change depending on the power-on configuration choice. See “:PON:TYPE” on page 105 for information on the power-on choices available.
OFF(0)	This selection blanks the signal generator’s display, hiding the settings and disabling the front panel keys. While in this mode, the display shows `*** SECURE DISPLAY ACTIVATED ***`.

For more information about security functions, refer to the *PSG User’s Guide*.

Example

```
:SYST:SEC:DISP OFF
```

The preceding example enables the secure display mode.

***RST** 1

Range N/A

Key Entry **Activate Security Display**

:SECurity:ERASeall

Supported All Models

```
:SYSTem:SECurity:ERASall
```

This command removes all user files, flatness correction files, and baseband generator files. In addition, all table editor files are returned to their original factory values.

This command differs from the :DELete:ALL command, which does not reset table editors to factory values. For more information about security functions, refer to the *PSG User’s Guide*.

Key Entry **Erase All**

:SECurity:LEVel

Supported All Models

```
:SYSTem:SECurity:LEVel NONE|ERASe|OVERwrite|SANitize
:SYSTem:SECurity:LEVel?
```

This command selects the security level operation for the signal generator.

NONE	This selection causes the signal generator to reset to factory default settings.
ERASe	This selection removes all user files, table editor files, flatness correction files, and baseband generator files.

OVERwrite — This selection removes all user files, table editor files, flatness correction files, and baseband generator files. The memory is then overwritten with random data.

SRAM	All addressable locations will be overwritten with random characters.
Hard Disk	All addressable locations will be overwritten with random characters.
Flash Memory	The flash blocks will be erased.

SANitize — This selection removes all user files, table editor files, flatness correction files, and baseband generator files using the same techniques as the OVERwrite selection for SRAM and flash memory. For the hard disk, the signal generator overwrites all addressable locations with a single character, its complement, and then with a random character.

Once you select the security level, you must execute the command from ":SECurity:LEVel:STATe" to arm the security level.

NOTE Once you select a security level and arm it, you cannot change the level.

For other cleaning and security operation descriptions, see ":SECurity:ERASeall" on page 109, ":SECurity:OVERwrite" on page 111, and ":SECurity:SANitize" on page 111. For more information about security functions, refer to the *PSG User's Guide*.

Example

```
:SYST:SEC:LEV NONE
```

The preceding example sets the secure mode so it resets the signal generator to factory settings after completing the security operation.

Key Entry **None** **Erase** **Overwrite** **Sanitize**

:SECurity:LEVel:STATe

Supported All Models

CAUTION Ensure that you select the security level prior to executing this command with the ON (1) selection. Once you enable the state, you cannot reduce the security level.

```
:SYSTem:SECurity:LEVel:STATe ON|OFF|1|0
:SYSTem:SECurity:LEVel:STATe?
```

This command arms and executes the current security level parameter.

On (1)	This selection arms and prevents any changes to the current security level. Refer to “:SECurity:LEVel” on page 109 for setting the security level.
OFF (0)	This selection performs the actions required for the current security level setting. Cycling the signal generator power also performs the same function.

For more information about security functions, refer to the *PSG User’s Guide*.

Example

```
:SYST:SEC:LEV:STAT ON
```

The preceding example arms the secure mode selected with the SYSTem:SECurity:LEVel command.

Key Entry **Enter Secure Mode**

:SECurity:OVERwrite

Supported All Models

```
:SYSTem:SECurity:OVERwrite
```

This command removes all user files, table editor files values, flatness correction files, and baseband generator files. The memory is then overwritten with random data as described below. For more information about security functions, refer to the *PSG User’s Guide*.

SRAM	All addressable locations will be overwritten with random characters.
HARD DISK	All addressable locations will be overwritten with random characters.
FLASH MEMORY	The flash blocks will be erased.

Key Entry **Erase and Overwrite All**

:SECurity:SANitize

Supported All Models

```
:SYSTem:SECurity:SANitize
```

This command removes all user files, table editor files values, flatness correction files, and baseband generator files. The memory is then overwritten with a sequence of data as described below. For more information about security functions, refer to the *PSG User’s Guide*.

SRAM	All addressable locations will be overwritten with random characters.
HARD DISK	All addressable locations will be overwritten with a single character and then a random character.
FLASH MEMORY	The flash blocks will be erased.

Key Entry **Erase and Sanitize All**

:SSAVer:DELay

Supported All Models

```
:SYSTem:SSAVer:DELay <val>
:SYSTem:SSAVer:DELay?
```

This command sets the amount of time before the display light or display light and text is switched off. The time delay represents the time during which there is no signal generator front panel input. The variable `<val>` is a positive integer number, in hours. The setting enabled by this command is not affected by power-on, preset, or `*RST`. See “:SSAVer:MODE” on page 112 for selecting the screen saver mode.

Example

```
:SYST:SSAV:DEL 2
```

The preceding example sets two hours delay time for the screen saver mode.

Range 1–12

Key Entry **Screen Saver Delay:**

:SSAVer:MODE

Supported All Models

```
:SYSTem:SSAVer:MODE LIGHt|TEXT
:SYSTem:SSAVer:MODE?
```

This command toggles the screen saver mode between light only or light and text.

LIGHt Enables only the light to turn off during the screen saver operation while leaving the text visible on the darkened screen.

TEXT Enables both the display light and text to turn off during screen saver operation.

The setting is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:SYST:SSAV:MODE TEXT
```

The preceding example sets the screen saver mode.

Key Entry **Screen Saver Mode**

:SSAVer:STATe

Supported All Models

```
:SYSTem:SSAVer:STATe ON|OFF|1|0
:SYSTem:SSAVer:STATe?
```

This command enables or disables the display screen saver. The setting enabled by this command is not affected by a signal generator power-on, preset, or *RST command.

Example

```
:SYST:SSAV:STAT 1
```

The preceding example enables the screen saver mode.

Key Entry **Screen Saver Off On**

:TIME

Supported All Models

```
:SYSTem:TIME <hour>,<minute>,<second>
:SYSTem:TIME?
```

This command sets the time displayed in the lower right area of the signal generator’s display.

Range *<hour>*: 0–23 *<minute>*: 0–59 *<second>*: 0–59

Example

```
:SYST:TIME 9,30,45
```

The preceding example sets the signal generator time to 09:30:45.

Key Entry **Time / Date**

:VERSion

Supported All Models

```
:SYSTem:VERSion?
```

This command returns the SCPI version number with which the signal generator complies.

Trigger Subsystem

:ABORt

Supported All Models

```
:ABORt
```

This command causes the List or Step sweep in progress to abort. If `INIT:CONT[:ALL]` is set to ON, the sweep will immediately re-initiate. The pending operation flag affecting `*OPC`, *OPC?, and `*WAI` will undergo a transition once the sweep has been reset.

:INITiate:CONTinuous[:ALL]

Supported All Models

```
:INITiate:CONTinuous[:ALL] ON|OFF|1|0
:INITiate:CONTinuous[:ALL]?
```

This command selects either a continuous or single List or Step sweep. Execution of this command does not affect a sweep in progress.

ON (1) Selects continuous sweep where, after the completion of the previous sweep, the sweep restarts automatically, or waits for a trigger.

OFF (0) This choice selects a single sweep. Refer to ":INITiate[:IMMediate][:ALL]" on page 115 for single sweep triggering information.

Example

```
:INIT:CONT ON
```

The preceding example enables the continuos mode for the sweep type.

***RST** 0

Key Entry **Sweep Repeat Single Cont**

:INITiate[:IMMediate][:ALL]

Supported All Models

`:INITiate[:IMMediate][:ALL]`

This command either sets or sets and starts a single List or Step sweep, depending on the trigger type. The command performs the following:

- arms a single sweep when BUS, EXTernal, or KEY is the trigger source selection
- arms and starts a single sweep when IMMediate is the trigger source selection

This command is ignored if a sweep is in progress. See ":INITiate:CONTinuous[:ALL]" on page 114 for setting continuous or single sweep. See ":TRIGger[:SEQuence]:SOURce" on page 116 to select the trigger source.

In some atypical cases, the `:INIT` command could be ignored if it immediately follows an `*OPC?` command. If the `:INIT` command is ignored, then use a 10ms sleep function before sending the command.

Key Entry **Single Sweep**

:TRIGger:OUTPut:POLarity

Supported All Models

```
:TRIGger:OUTPut:POLarity POSitive|NEGative
:TRIGger:OUTPut:POLarity?
```

Sets the TTL signal level present at the TRIGGER OUT connector to either high (5 vdc) or low (0 vdc). The trigger out is asserted after the frequency and/or power is set while the sweep is waiting for its step trigger. In addition, the swept-sine sends a pulse to the TRIGGER OUT at the beginning of each sweep.

Example

`:TRIG:OUTP:POL NEG`

The preceding example enables the continuos mode as the sweep type.

***RST** POS

Key Entry **Trigger Out Polarity Neg Pos**

:TRIGger[:SEQuence]:SLOPe

Supported All Models

```
:TRIGger[:SEQuence]:SLOPe POSitive|NEGative
:TRIGger[:SEQuence]:SLOPe?
```

This command sets the polarity of the ramp or sawtooth waveform slope present at the TRIGGER IN connector that will trigger a List or Step sweep.

Example

```
:TRIG:SLOP POS
```

The preceding example sets a positive ramp slope.

***RST** POS

Key Entry **Trigger In Polarity Neg Pos**

:TRIGger[:SEQuence]:SOURce

Supported All Models

```
:TRIGger[:SEQuence]:SOURce BUS|IMMediate|EXTernal|KEY
:TRIGger[:SEQuence]:SOURce?
```

This command sets the sweep trigger source for a List or Step sweep.

BUS	This choice enables GPIB triggering using the `*TRG` or `GET` command or LAN and RS-232 triggering using the `*TRG` command.
IMMediate	This choice enables immediate triggering of the sweep event.
EXTernal	This choice enables the triggering of a sweep event by an externally applied signal at the TRIGGER IN connector.
KEY	This choice enables front-panel triggering by pressing the **Trigger** hardkey.

The wait for the BUS, EXTernal, or KEY trigger can be bypassed by sending the `:TRIGger[:SEQuence][:IMMediate]` command.

Example

```
:TRIG:SOUR BUS
```

The preceding example sets the sweep trigger source to BUS.

***RST** IMM

Key Entry **Bus** **Free Run** **Ext** **Trigger Key**

:TRIGger[:SEQuence][:IMMediate]

Supported All Models

`:TRIGger[:SEQuence][:IMMediate]`

This event command causes an armed List or Step sweep to immediately start without the selected trigger occurring.

In some atypical cases, the `:TRIG` command could be ignored if it immediately follows an `*OPC?` command. If the `:TRIG` command is ignored, then use a 10ms sleep function before sending the command.

Unit Subsystem (:UNIT)

:POWer

Supported All Models

```
:UNIT:POWer DBM|DBUV|DBUVEMF|V|VEMF|DB
:UNIT:POWer?
```

This command terminates an amplitude value in the selected unit of measure.

If the amplitude reference state is set to on, the query returns units expressed in dB. Setting any other unit will cause a setting conflict error stating that the amplitude reference state must be set to off. Refer to, ":REFerence:STATe" on page 162 for more information.

All power values in this chapter are shown with DBM as the unit of measure. If a different unit of measure is selected, replace DBM with the newly selected unit whenever it is indicated for the value.

Example

```
:UNIT:POW DBM
```

The preceding example selects dBm as the unit of amplitude measurement.

***RST** DBM

Key Entry **dBm** **dBuV** **dBuVemf** **mV** **uV** **mVemf** **uVemf**

3 Basic Function Commands

In the following sections, this chapter provides SCPI descriptions for subsystems dedicated to signal generator operations common to all PSG models:

Correction Subsystem ([:SOURce]:CORRection)

:FLATness:LOAD

Supported All Models

```
[:SOURce]:CORRection:FLATness:LOAD "<file_name>"
```

This command loads a user-flatness correction file designated by the file name `"<file_name>"` variable. The file will be loaded from the signal generator's `USERFLAT` directory. The directory path does not need to be specified in the command. Refer to the *PSG Programming Guide* for more information on flatness corrections.

For information on file name syntax, refer to "File Name Variables" on page 12.

Example

```
:CORR:FLAT:LOAD "Flatness_Data"
```

The preceding example loads a user flatness file named `Flatness_Data` from the signal generator's user flatness directory.

Key Entry **Load From Selected File**

:FLATness:PAIR

Supported All Models

```
[:SOURce]:CORRection:FLATness:PAIR <freq>,<corr>
```

This command adds or edits a frequency and amplitude correction pair. The maximum number of pairs or points that can be entered is 1601. Refer to the *PSG Programming Guide* for more information on flatness corrections.

The `<corr>` variable is the power correction in dB.

Power and frequency ranges for different signal generator models and options are listed on page 164.

Example

```
:CORR:FLAT:PAIR 10MHZ,.1
```

The preceding example enters a frequency of 10 megahertz and a power of 0.1dB into the user flatness table.

***RST** *Option 520*: +2.0000000000000E+10

Option 532: +3.2000000000000E+10

Option 540: +4.0000000000000E+10

Option 544: +4.4000000000000E+10

Option 550: +5.0000000000000E+10

Option 567: +6.7000000000000E+10

Range *Option 520*: 250kHZ–20GHZ

Option 532: 250kHZ–32GHZ

Option 540: 250kHZ–40GHZ

Option 544: 250kHZ–44GHZ

Option 550: 250kH–50GHZ

Option 567: 250kHZ–70GHZ[a]

a. 67-70 GHz performance not specified

Key Entry **Configure Cal Array**

:FLATness:POINts

Supported All Models

`[:SOURce]:CORRection:FLATness:POINts?`

This query returns the number of points in the user-flatness correction file.

:FLATness:PRESet

Supported All Models

CAUTION Once this command is executed, correction data is overwritten; If needed, save the current correction data (see ":FLATness:STORe" on page 122).

`[:SOURce]:CORRection:FLATness:PRESet`

This command presets the user-flatness correction to a factory-defined setting that consists of one frequency point and one amplitude point with no corrections.

Key Entry **Preset List**

:FLATness:STORe

Supported All Models

```
[:SOURce]:CORRection:FLATness:STORe "<file_name>"
```

This command stores the current user-flatness correction data to a file named by the `"<file_name>"` variable. All user-flatness files are stored in the signal generator's `USERFLAT` directory. The directory path does not need to be specified in the command.

For information on file name syntax, refer to "File Name Variables" on page 12.

Example

```
:CORR:FLAT:STOR "New_Flat_data"
```

The preceding example stores the current user-flatness table entries in a file named `"New_Flat_data"`.

Key Entry **Store To File**

[:STATe]

Supported All Models

```
[:SOURce]:CORRection[:STATe] ON|OFF|1|0
[:SOURce]:CORRection[:STATe]?
```

This command toggles the application of user-flatness corrections to the current signal generator power output.

Example

```
:CORR OFF
```

The preceding example turns off correction data.

***RST** 0

Key Entry **Flatness Off On**

Frequency Subsystem ([:SOURce])

:FREQuency:CENTer

Supported All Models with Option 007

```
[:SOURce]:FREQuency:CENTer <num>[<freq_suffix>]|UP|DOWN
[:SOURce]:FREQuency:CENTer? [MAXimum|MINimum]
```

This command sets the center frequency for a ramp sweep. The center frequency symmetrically divides the selected frequency span and is coupled to the start and stop frequency settings. The frequency range and reset values are dependent on the signal generator model and option number.

The query returns the start and stop ramp frequencies if the optional `MAXimum` or `MINimum` are used.

***RST** *Option 520*: +2.0000000000000E+10

Option 532: +3.20000000000000E+10

Option 540: +4.0000000000000E+10

Option 544: +4.4000000000000E+10

Option 550: +5.0000000000000E+10

Option 567: +7.0000000000000E+10

Range *Option 520*: 250kHZ–20GHZ

Option 532: 250kHZ–32GHZ

Option 540: 250kHZ–40GHZ

Option 544: 250kHZ–44GHZ

Option 550: 250kH–50GHZ

Option 567: 250kHZ–70GHZ[a]

a. 67-70 GHz performance not specified

Example

```
:FREQ:CENT 15GHZ
```

The preceding example sets the center frequency for a ramp sweep to 15 GHz.

Key Entry **Freq Center**

:FREQuency:CHANnels:BAND

Supported All Models

```
[:SOURce]:FREQuency:CHANnels:BAND
NBASe|NMOBile|BPGSm|MPGSm|BEGSm|MEGSm|
BRGSm|MRGSm|BDCS|MDCS|BPCS|MPCS|B450|GM450|B480|M480|B850|M850|B8|M8
|B15|M15|B390|B420|B460|B915|M380|M410|M450|M870|PHS|DECT
[:SOURce]:FREQuency:CHANnels:BAND?
```

This command sets the frequency of the signal generator by specifying a frequency channel band. The frequency channel state must be enabled for this command to work. See ":FREQuency:CHANnels[:STATe]" on page 126.

NBASe	This choice selects Standard Base as the frequency band for NADC.
NMOBile	This choice selects Standard Mobile as the frequency band for NADC.
BPGSm	This choice selects P-Gsm 900 Base as the frequency band for GSM.
MPGSm	This choice selects P-Gsm 900 Mobile as the frequency band for GSM.
BEGSm	This choice selects E-Gsm 900 Base as the frequency band for GSM.
MEGSm	This choice selects E-Gsm 900 Mobile as the frequency band for GSM.
BRGSm	This choice selects R-Gsm 900 Base as the frequency band for GSM.
MRGSm	This choice selects R-Gsm 900 Mobile as the frequency band for GSM.
BDCS	This choice selects DCS 1800 Base as the frequency band for GSM.
MDCS	This choice selects DCS 1800 Mobile as the frequency band for GSM.
BPCS	This choice selects PCS 1900 Base as the frequency band for GSM.
MPCS	This choice selects PCS 1900 Mobile as the frequency band for GSM.
B450	This choice selects Gsm 450 Base as the frequency band for GSM.
GM450	This choice selects Gsm 450 Mobile as the frequency band for GSM.
B480	This choice selects Gsm 480 Base as the frequency band for GSM.
M480	This choice selects Gsm 480 Mobile as the frequency band for GSM.
B850	This choice selects Gsm 850 Base as the frequency band for GSM.
M850	This choice selects Gsm 850 Mobile as the frequency band for GSM.
B8	This choice selects 800MHz Base as the frequency band for PDC.
M8	This choice selects 800MHz Mobile as the frequency band for PDC.
B15	This choice selects 1500MHz Base as the frequency band for PDC.
M15	This choice selects 1500MHz Mobile as the frequency band for PDC.
B390	This choice selects Base 390-400 as the frequency band for TETRA.
B420	This choice selects Base 420-430 as the frequency band for TETRA.
B460	This choice selects Base 460-470 as the frequency band for TETRA.
B915	This choice selects Base 915-921 as the frequency band for TETRA.
M380	This choice selects Mobile 380-390 as the frequency band for TETRA.
M410	This choice selects Mobile 410-420 as the frequency band for TETRA.

M450	This choice selects Mobile 450-460 as the frequency band for TETRA.
M870	This choice selects Mobile 870-876 as the frequency band for TETRA.
PHS	This choice selects Standard PHS as the frequency band.
DECT	This choice selects Standard DECT as the frequency band.

Example

```
:FREQ:CHAN:BAND DECT
```

The preceding example sets the frequency band to standard DECT.

***RST** BPGS

Key Entry **P-GSM Base** **E-GSM Base** **R-GSM Base** **DCS Base**
PCS Base **GSM 450 Base** **GSM 480 Base** **GSM 850 Base**
NADC Base **800MHZ Base** **1500MHZ Base**
Tetra Base 390/400 **Tetra Base 420/430** **Tetra Base 460/470**
Tetra Base 915/921 **PHS Standard** **DECT Standard**
P-GSM Mobile **E-GSM Mobile** **R-GSM Mobile** **DCS Mobile**
PCS Mobile **GSM 450 Mobile** **GSM 480 Mobile** **GSM 850 Mobile**
NADC Mobile **800MHZ Mobile** **1500MHZ Mobile**
Tetra Mobile 380/390 **Tetra Mobile 410/420** **Tetra Mobile 450/460**
Tetra Mobile 870/876

:FREQuency:CHANnels:NUMBer

Supported All Models

```
[:SOURce]:FREQuency:CHANnels:NUMBer <number>
[:SOURce]:FREQuency:CHANnels:NUMBer?
```

This command sets the frequency of the signal generator by specifying a channel number of a given frequency band.

The channel band and channel state must be enabled for this command to work. Refer to ":FREQuency:CHANnels[:STATe]" on page 126.

Example

```
:FREQ:CHAN:NUMB 24
```

The preceding example sets the channel number to 24 for the current band.

***RST**	+1	
Range	P-GSM Base/Mobile:	1–24
	E-GSM and R-GSM Base/Mobile:	1–1023
	DCS Base/Mobile:	512–885
	PCS Base/Mobile:	512–900
	GSM-450 Base/Mobile:	259–293
	GSM-480 Base/Mobile:	306–340
	GSM-850 Base/Mobile:	128–251
	NADC Base/Mobile:	1–1023
	800MHZ Base/Mobile:	0–640
	1500MHZ Base/Mobile:	0–960
	TETRA 380/390 Mobile:	3600–4000
	TETRA 390/4000 Base:	3600–4000
	TETRA 410/420 Mobile:	800–1200
	TETRA 420/430 Base:	800–1200
	TETRA 460/470: 2400 through 2800	2400–2800
	TETRA 870/876 Mobile:	600–640
	TETRA 915/921 Base:	600–940
	PHS Standard:	1–255
	DECT Standard:	0–9

Key Entry **Channel Number**

:FREQuency:CHANNels[:STATe]

Supported All Models

```
[:SOURce]:FREQuency:CHANnels[:STATe] ON|OFF|1|0
[:SOURce]:FREQuency:CHANnels[:STATe]?
```

This command enables or disables the frequency channel and band selection. The signal generator frequency will be set to the channel frequency when the state is on. To set frequency channel bands refer to ":FREQuency:CHANnels:BAND" on page 123.

Example

```
:FREQ:CHAN ON
```

The preceding example turns on the frequency channel.

***RST** 0

Key Entry **Freq Channels Off On**

:FREQuency:FIXed

Supported All Models

```
[:SOURce]:FREQuency:FIXed <val><units>
[:SOURce]:FREQuency:FIXed?
```

This command sets the signal generator output frequency. To set the frequency mode, see ":FREQuency:MODE" on page 128. For a listing of signal generator frequency and power specifications, refer to "[:LEVel][:IMMediate][:AMPLitude]" on page 164.

Example

```
:FREQ:FIX 10GHZ
```

The preceding example sets the signal generator frequency to 10 GHz.

***RST** *Option 520*: +2.0000000000000E+10

Option 532: +3.20000000000000E+10

Option 540: +4.0000000000000E+10

Option 544: +4.4000000000000E+10

Option 550: +5.0000000000000E+10

Option 567: +6.7000000000000E+10

Range *Option 520*: 250kHZ–20GHZ

Option 532: 250kHZ–32GHZ

Option 540: 250kHZ–40GHZ

Option 544: 250kHZ–44GHZ

Option 550: 250kH–50GHZ

Option 567: 250kHZ–70GHZ[a]

a. 67-70 GHz performance not specified

Key Entry **Freq CW**

:FREQuency:MANual

Supported All Models with Option 007

```
[:SOURce]:FREQuency:MANual <val><unit>
[:SOURce]:FREQuency:MANual?
```

This command sets the RF output frequency when performing a ramp sweep in manual mode. The frequency value selected must fall within the range of the current start and stop frequency settings.

Entering a value with this command has no effect unless manual sweep mode is on. Refer to ":SWEep:MODE" on page 147 for setting the mode.

The variable `<val>` is a numeric value. The `<units>` variable can be expressed in HZ, KHZ, MHZ, or GHZ.

Example

```
:FREQ:MAN 10GHZ
```

The preceding example sets the signal generator manual ramp sweep frequency to 10 GHz.

***RST**	*Option 520*: +2.0000000000000E+10
	Option 532: +3.20000000000000E+10
	Option 540: +4.0000000000000E+10
	Option 544: +4.4000000000000E+10
	Option 550: +5.0000000000000E+10
	Option 567: +6.7000000000000E+10
Range	*Option 520*: 250kHZ–20GHZ
	Option 532: 250kHZ–32GHZ
	Option 540: 250kHZ–40GHZ
	Option 544: 250kHZ–44GHZ
	Option 550: 250kH–50GHZ
	Option 567: 250kHZ–70GHZ[a]

a. 67-70 GHz performance not specified

Key Entry **Manual Freq**

:FREQuency:MODE

Supported All Models

```
[:SOURce]:FREQuency:MODE FIXed|CW|SWEep|LIST
```

`[:SOURce]:FREQuency:MODE?`

This command sets the frequency mode of the signal generator.

FIXed and CW	These choices are synonymous. Any currently running frequency sweeps are turned off, and the current CW frequency settings are used to control the output frequency. To set the frequency in the CW frequency mode, see ":FREQuency[:CW]" on page 135. To set the frequency in the fixed frequency mode, see ":FREQuency:FIXed" on page 127.
SWEep	The effects of this choice are determined by the sweep generation type selected (refer to ":SWEep:GENeration" on page 147). In analog sweep generation, the ramp sweep frequency settings (start, stop, center, and span) control the output frequency. In step sweep generation, the current step sweep frequency settings control the output frequency. In both cases, this selection also activates the sweep. This choice is available with Option 007 only.
LIST	This choice selects the swept frequency mode. If sweep triggering is set to immediate along with continuous sweep mode, executing the command starts the LIST or STEP frequency sweep.

NOTE To perform a frequency and amplitude sweep, you must also select LIST or SWEep as the power mode (see ":MODE" on page 160).

Example

`:FREQ:MODE LIST`

The preceding example selects a list frequency sweep.

***RST** CW

Key Entry **Freq CW** **Sweep Type** **Freq** **Off** **Freq & Ampl**

:FREQuency:MULTiplier

Supported All Models

```
[:SOURce]:FREQuency:MULTiplier <val>
[:SOURce]:FREQuency:MULTiplier?
```

This command sets the multiplier for the signal generator carrier frequency. For any multiplier other than one, the `MULT` indicator is shown in the frequency area of the display. The multiplier value is used to multiply the signal generator's displayed frequency. The true frequency remains constant. For example, if the signal generator frequency is 20 GHz and a multiplier of 3 is selected, the displayed frequency will be 60 GHz. This feature is useful when working with mixers and multipliers.

Example

```
:FREQ:MULT 2
```

The preceding example sets the carrier multiplier to 2.

***RST**	+1.00000000E+000
Key Entry	**Freq Multiplier**

:FREQuency:OFFSet

Supported	All Models

```
[:SOURce]:FREQuency:OFFSet <val><units>
[:SOURce]:FREQuency:OFFSet?
```

This command sets the frequency offset. When an offset has been entered, the `OFFS` indicator appears in the frequency area of the signal generator's front-panel display and the frequency reading will include the offset value.

When any non-zero value is entered, the frequency offset state turns on; entering zero turns it off. To set the offset state independent of entering offset values see ":FREQuency:OFFSet:STATe" .

Example

```
:FREQ:OFFS 10GHZ
```

The preceding example sets the frequency offset to 10 GHz.

***RST**	+0.0000000000000E+00
Range	–200GHZ to 200GHZ
Key Entry	**Freq Offset**

:FREQuency:OFFSet:STATe

Supported	All Models

```
[:SOURce]:FREQuency:OFFSet:STATe ON|OFF|1|0
[:SOURce]:FREQuency:OFFSet:STATe?
```

This command enables or disables the offset frequency.

Entering OFF (0) will set the frequency offset to 0 Hz.

Example

```
:FREQ:OFFS:STAT 0
```

The preceding example disables the frequency offset and sets the offset to 0 hertz.

***RST** 0

Key Entry **Freq Offset**

:FREQuency:REFerence

Supported All Models

```
[:SOURce]:FREQuency:REFerence <val><units>
[:SOURce]:FREQuency:REFerence?
```

This command sets the output reference frequency for the signal generator. Once the reference frequency is set, any change to the signal generator's CW frequency will be displayed referenced to 0 hertz. For example, if the signal generator's CW frequency is set to 100 megahertz and the frequency reference is set (the frequency reference state will automatically turn on). The frequency display will read 0 Hz. If you change the signal generator's CW frequency to 1 megahertz, the frequency display will read 1 megahertz. However, the true frequency is 101 megahertz. This can be verified by turning the frequency reference state off. The signal generator frequency display will read 101 megahertz. Refer to :FREQuency:REFerence:STATe" for more information.

Example

```
:FREQ:REF 100MHZ
```

The preceding example sets the output reference frequency to 100 megahertz.

***RST** +0.0000000000000E+00

Key Entry **Freq Ref Set**

:FREQuency:REFerence:SET

Supported All Models

```
[:SOURce]:FREQuency:REFerence:Set
```

This command sets the current CW output frequency, along with any offset, as a 0 hertz reference value.

***RST** +0.0000000000000E+00

Key Entry **Freq Ref Set**

:FREQuency:REFerence:STATe

Supported All Models

```
[:SOURce]:FREQuency:REFerence:STATe ON|OFF|1|0
[:SOURce]:FREQuency:REFerence:STATe?
```

This command enables or disables the frequency reference mode. When the frequency reference mode is on, changes in the signal generator’s CW frequency are displayed relative to the 0 hertz frequency reference. When the state is off, the front-panel display indicates the true signal generator frequency.

Example

```
:FREQ:REF:STAT OFF
```

The preceding example turns off the reference frequency mode.

***RST** 0

Key Entry **Freq Ref Off On**

:FREQuency:SPAN

Supported All Models with Option 007

```
[:SOURce]:FREQuency:SPAN <num>[<freq_suffix>]|UP|DOWN
[:SOURce]:FREQuency:SPAN? [MAXimum|MINimum]
```

This command sets the length of the frequency range for a ramp sweep. Span setting is symmetrically divided by the selected center frequency and is coupled to the start and stop frequency settings. The span range is dependent on the signal generator model and option number.

Example

```
:FREQ:SPAN 100MHZ
```

The preceding example sets the frequency span to 100 megahertz.

***RST** +0.0000000000000E+00

Key Entry **Freq Span**

:FREQuency:STARt

Supported All Models

```
[:SOURce]:FREQuency:STARt <val><units>
[:SOURce]:FREQuency:STARt?
```

This command sets the frequency start point for a step sweep or ramp sweep (Option 007). In a ramp sweep setup, the selected value must be less than or equal to the value selected for the frequency stop point. In ramp sweep, this setting is coupled with the span and center frequency settings.

Refer to "[:LEVel][:IMMediate][:AMPLitude]" on page 164 for frequency and power specifications for different signal generator options and model numbers.

Example

```
:FREQ:STAR 1GHZ
```

The preceding example sets the start frequency for a sweep to 1 GHz.

***RST**	*Option 520*: +2.0000000000000E+10
	Option 532: +3.20000000000000E+10
	Option 540: +4.0000000000000E+10
	Option 544: +4.4000000000000E+10
	Option 550: +5.0000000000000E+10
	Option 567: +6.7000000000000E+10
Range	*Option 520*: 250kHZ–20GHZ
	Option 532: 250kHZ–32GHZ
	Option 540: 250kHZ–40GHZ
	Option 544: 250kHZ–44GHZ
	Option 550: 250kH–50GHZ
	Option 567: 250kHZ–70GHZ[a]

a. 67-70 GHz performance not specified

Key Entry	**Freq Start**

:FREQuency:STOP

Supported	All Models

```
[:SOURce]:FREQuency:STOP <val><units>
[:SOURce]:FREQuency:STOP?
```

This command sets the stop frequency for a step sweep or ramp sweep (Option 007). In a ramp sweep setup, the selected value must be greater than or equal to the value selected for the frequency start point. In ramp sweep, this setting is coupled with the span and center frequency settings.

Refer to "[:LEVel][:IMMediate][:AMPLitude]" on page 164 for frequency and power specifications for different signal generator options and model numbers.

Example

```
:FREQ:STOP 10GHZ
```

The preceding example sets the stop frequency for a sweep to 10 GHz.

***RST**	*Option 520*: +2.0000000000000E+10
	Option 532: +3.20000000000000E+10
	Option 540: +4.0000000000000E+10
	Option 544: +4.4000000000000E+10
	Option 550: +5.0000000000000E+10
	Option 567: +6.7000000000000E+10
Range	*Option 520*: 250kHZ–20GHZ
	Option 532: 250kHZ–32GHZ
	Option 540: 250kHZ–40GHZ
	Option 544: 250kHZ–44GHZ
	Option 550: 250kH–50GHZ
	Option 567: 250kHZ–70GHZ[a]

a. 67-70 GHz performance not specified

Key Entry **Freq Stop**

:FREQuency:SYNThesis

Supported All Models except Option UNR

```
[:SOURce]:FREQuency:SYNThesis 1|2
[:SOURce]:FREQuency:SYNThesis?
```

This command sets the phase-lock loop (PLL) bandwidth to optimize phase noise for offsets above and below 10 kHz.

`1`	This choice will select mode 1 which optimizes phase noise at offsets below 10 kHz.
`2`	This choice will select mode 2 which optimizes phase noise at offsets above 10 kHz.

Example

```
:FREQ:SYNT 2
```

The preceding example sets PLL bandwidth to mode 2.

***RST** +1

Key Entry **Mode 1 Optimize <10kHZ Offset** **Mode 2 Optimize >10kHZ Offset**

:FREQuency[:CW]

Supported All Models

```
[:SOURce]:FREQuency[:CW] <val><unit>
[:SOURce]:FREQuency[:CW]?
```

This command sets the signal generator output frequency for the CW frequency mode.

To set the frequency mode to CW, refer to “:FREQuency:MODE” on page 128.

Example

```
:FREQ 12GHZ
```

The preceding example sets signal generator’s output frequency to 12 GHz.

***RST** *Option 520*: +2.0000000000000E+10

Option 532: +3.20000000000000E+10

Option 540: +4.0000000000000E+10

Option 544: +4.4000000000000E+10

Option 550: +5.0000000000000E+10

Option 567: +6.7000000000000E+10

Range *Option 520*: 250kHZ–20GHZ

Option 532: 250kHZ–32GHZ

Option 540: 250kHZ–40GHZ

Option 544: 250kHZ–44GHZ

Option 550: 250kH–50GHZ

Option 567: 250kHZ–70GHZ[a]

a. 67-70 GHz performance not specified

Key Entry **Frequency**

:PHASe:REFerence

Supported All Models

```
[:SOURce]:PHASe:REFerence
```

This command sets the output phase reference to zero. Subsequent phase adjustments are set relative to the new reference.

Key Entry **Phase Ref Set**

:PHASe[:ADJust]

Supported All Models

```
[:SOURce]:PHASe[:ADJust] <val><unit>
[:SOURce]:PHASe[:ADJust]?
```

This command adjusts the phase of the modulating signal. The query returns values in radians.

Example

```
:PHAS 30DEG
```

The preceding example sets the phase of the modulating signal to 30 degrees relative to the previous phase setting.

***RST** +0.00000000E+000

Range *Radians*: –3.14 to 3.14RAD *Degrees*: –180 to 179DEG

Key Entry **Adjust Phase**

:ROSCillator:BANDwidth:DEFaults

Supported All Models with Option UNR

```
[:SOURce]:ROSCillator:BANDwidth:DEFaults
```

This command resets the bandwidth of the reference oscillator to the factory-defined default state. The default value for the internal reference bandwidth is 125 Hz. The default value for the external reference bandwidth is 25 Hz.

Key Entry **Restore Factory Defaults**

:ROSCillator:BANDwidth:EXTernal

Supported All Models with Option UNR

```
[:SOURce]:ROSCillator:BANDwidth:EXTernal 25HZ|55HZ|125HZ|300HZ|650HZ
[:SOURce]:ROSCillator:BANDwidth:EXTernal?
```

This command sets the bandwidth of the external reference oscillator.

Example

```
:ROSC:BAND:EXT 300HZ
```

The preceding example sets the bandwidth of the external oscillator to 300 hertz.

Key Entry **External Ref Bandwidth**

:ROSCillator:BANDwidth:INTernal

Supported All Models with Option UNR

```
[:SOURce]:ROSCillator:BANDwidth:INTernal 25HZ|55HZ|125HZ|300HZ|650HZ
[:SOURce]:ROSCillator:BANDwidth:INTernal?
```

This command sets the bandwidth of the internal reference oscillator.

Example

```
:ROSC:BAND:INT 125HZ
```

The preceding example sets the bandwidth of the internal oscillator to 125 hertz.

Key Entry **Internal Ref Bandwidth**

:ROSCillator:SOURce

Supported All Models

```
[:SOURce]:ROSCillator:SOURce?
```

This command queries the reference oscillator source: INT (internal) or EXT (external).

:ROSCillator:SOURce:AUTO

Supported All Models without Option UNR

```
[:SOURce]:ROSCillator:SOURce:AUTO ON|OFF|1|0
[:SOURce]:ROSCillator:SOURce:AUTO?
```

This command enables or disables the ability of the signal generator to automatically select between the internal and an external reference oscillator.

ON (1) This choice enables the signal generator to detect when a valid reference signal is present at the 10 MHz IN connector and automatically switches from internal to external frequency reference.

OFF (0) This choice selects the internal reference oscillator and disables the switching capability between the internal and an external frequency reference.

Example

```
:ROSC:SOUR:AUTO 0
```

The preceding example turns off the automatic selection of internal or external reference oscillators.

***RST** 1

Key Entry **Ref Oscillator Source Auto Off On**

List / Sweep Subsystem ([:SOURce])

A complete sweep setup requires commands from other subsystems. Table 3-1 shows the function and location of these commands.

Table 3-1 Location of Commands from the other Subsystems

Sweep Type	Function	Command Location	Key Entry under Sweep / List key
List and Step	Start/stop frequency sweep	":FREQuency:MODE" (page 128)	**Freq Off**
	Start/stop amplitude sweep	":MODE" (page 160)	**Ampl Off**
	Start/stop frequency and amplitude sweep[a]	":MODE" (page 160) ":FREQuency:MODE" (page 128)	**Freq & Ampl Off**
	Set up & control sweep triggering[b]	"Trigger Sweep Subsystem ([:SOURce])" (page 167)	See the "Trigger Sweep Subsystem ([:SOURce])"
Step	Start frequency sweep	":FREQuency:STARt" (page 132)	**Freq Start**
	Stop frequency sweep	":FREQuency:STOP" (page 133)	**Freq Stop**
	Start amplitude sweep	":STARt" (page 163)	**Ampl Start**
	Stop amplitude sweep	":STOP" (page 163)	**Ampl Stop**

a. Execute both commands to start or stop a frequency and amplitude sweep.
b. For point to point triggering, see ":LIST:TRIGger:SOURce" on page 143.

:LIST:DIRection

Supported All Models

```
[:SOURce]:LIST:DIRection UP|DOWN
[:SOURce]:LIST:DIRection?
```

This command sets the direction of a list or step sweep.

UP This choice enables a sweep in an ascending order:

- first to last point for a list sweep
- start to stop for a step sweep

DOWN This choice reverses the direction of the sweep.

Example

`:LIST:DIR UP`

The preceding example selects an ascending sweep direction.

***RST** UP

Key Entry **Sweep Direction Down Up**

:LIST:DWELl

Supported All Models

```
[:SOURce]:LIST:DWELl <val>{,<val>}
[:SOURce]:LIST:DWELl?
```

This command sets the dwell time for points in the current list sweep.

The variable `<val>` is expressed in units of seconds with a 0.001 resolution. If only one point is specified, that value is used for all points in the list. Otherwise, there must be a dwell point for each frequency and amplitude point in the list.

NOTE	The dwell time `<val>` does not begin until the signal generator frequency and/or amplitude change has settled.

Dwell time is used when IMMediate is the trigger source. Refer to ":LIST:TRIGger:SOURce" on page 143 for the trigger setting.

The dwell time is the amount of time the sweep pauses after setting the frequency and/or power for the current point.

The setting enabled by this command is not affected by a signal generator power cycle, preset, or `*RST` command.

Example

`:LIST:DWEL .1,.2,.1,.2,.3`

The preceding example sets the dwell time for a list of five points.

Range 0.001–60

:LIST:DWELl:POINts

Supported All Models

```
[:SOURce]:LIST:DWELl:POINts?
```

This command queries the signal generator for the number of dwell points in the list sweep file.

:LIST:DWELl:TYPE

Supported All Models

```
[:SOURce]:LIST:DWELl:TYPE LIST|STEP
[:SOURce]:LIST:DWELl:TYPE?
```

This command toggles the dwell time for the list sweep points between the values defined in the list sweep and the value for the step sweep.

LIST This choice selects the dwell times from the list sweep. Refer to ":LIST:DWELl" on page 139 for setting the list dwell points.

STEP This choice selects the dwell time from the step sweep. Refer to ":SWEep:DWELl" on page 146 for setting the step dwell.

Example

```
:LIST:DWEL:TYPE STEP
```

The preceding example selects the dwell time from step sweep values.

***RST** LIST

Key Entry **Dwell Type List Step**

:LIST:FREQuency

Supported All Models

```
[:SOURce]:LIST:FREQuency <val>{,<val>}
[:SOURce]:LIST:FREQuency?
```

This command sets the frequency values for the current list sweep points. The maximum number of points is 1601. The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

The variable `<val>` is expressed in hertz.

For signal generator frequency and power specifications, refer to "[:LEVel][:IMMediate][:AMPLitude]" on page 164.

Example

```
:LIST:FREQ 10GHZ,12GHZ,14GHZ,16GHZ
```

The preceding example sets the frequency value for a four point sweep.

***RST**	*Option 520*: +2.0000000000000E+10
	Option 532: +3.20000000000000E+10
	Option 540: +4.0000000000000E+10
	Option 544: +4.4000000000000E+10
	Option 550: +5.0000000000000E+10
	Option 567: +6.7000000000000E+10
Range	*Option 520*: 250kHZ–20GHZ
	Option 532: 250kHZ–32GHZ
	Option 540: 250kHZ–40GHZ
	Option 544: 250kHZ–44GHZ
	Option 550: 250kH–50GHZ
	Option 567: 250kHZ–70GHZ[a]

a. 67–70 GHz performance not specified

:LIST:FREQuency:POINts

Supported All Models

```
[:SOURce]:LIST:FREQuency:POINts?
```

This command queries the current list sweep file for the number of frequency points.

:LIST:MANual

Supported All Models

```
[:SOURce]:LIST:MANual <val> |UP|DOWN
[:SOURce]:LIST:MANual?
```

This command selects a list point or step sweep point as the current sweep point controlling the frequency and power output. If list or step mode is controlling frequency or power, or both, the indexed point in the respective list(s) is used.

The MANual mode must be selected and sweep enabled for this command to have an effect.

For information on setting the proper mode, see ":LIST:MODE" .

If the point selected is beyond the length of the longest enabled list, the point sets to the maximum possible point, and an error is generated.

Example

```
:LIST:MAN UP
```

The preceding example selects the next positive–direction, sequential point in the list.

Range List Sweep: 1– 1601
Step Sweep: 1– 65535

Key Entry **Manual Point**

:LIST:MODE

Supported All Models

```
[:SOURce]:LIST:MODE AUTO|MANual
[:SOURce]:LIST:MODE?
```

This command sets the operating mode for the current list or step sweep.

AUTO This choice enables the selected sweep type to perform a sweep of all points.

MANual This choice enables you to select an individual sweep point to control the RF output parameters. For more about selecting a sweep point, see ":LIST:MANual" on page 141.

Example

```
:LIST:MODE AUTO
```

The preceding example sets the mode to automatic.

***RST** AUTO

Key Entry **Manual Mode Off On**

:LIST:POWer

Supported All Models

```
[:SOURce]:LIST:POWer <val>{,<val>}
[:SOURce]:LIST:POWer?
```

This command sets the amplitude for the current list sweep points.

The setting enabled by this command is not affected by signal generator power-on, preset, or `*RST`.

During an amplitude sweep operation, signal generators with Option 1E1 protect the step attenuator by automatically switching to attenuator hold mode (OFF). The attenuator locks at its current setting and the amplitude sweep range is limited to 40 dB. The maximum number of points is 1601.

Example

```
:LIST:POW .1,.2,.1,.3,.1,-.1
```

The preceding example sets the power level for a six point sweep list.

Range See “[:LEVel][:IMMediate][:AMPLitude]” on page 164.

:LIST:POWer:POINts

Supported All Models

```
[:SOURce]:LIST:POWer:POINts?
```

This command queries the number of power points in the current list sweep file.

:LIST:RETRace

Supported All Models

```
[:SOURce]:LIST:RETRace ON|OFF|1|0
[:SOURce]:LIST:RETRace?
```

Upon completion of a single sweep operation, this command either resets the sweep to the first sweep point, or leaves it at the last sweep point. The command is valid for the list, step, or ramp (Option 007) single-sweep modes.

ON (1) The sweep resets to the first sweep point.

OFF (0) The sweep stays at the last sweep point.

Example

```
:LIST:RETR 1
```

The preceding example sets the retrace on. The sweep will reset to the first point after completing a sweep.

***RST** 1

Key Entry **Sweep Retrace Off On**

:LIST:TRIGger:SOURce

Supported All Models

```
[:SOURce]:LIST:TRIGger:SOURce BUS|IMMediate|EXTernal|KEY
[:SOURce]:LIST:TRIGger:SOURce?
```

This command sets the trigger source for a list or step sweep event.

To set the sweep trigger, see ":TRIGger[:SEQuence]:SOURce" on page 116.

BUS	This choice enables GPIB triggering using the `*TRG` or `GET` command, or LAN and RS-232 triggering using the `*TRG` command.
IMMediate	This choice enables immediate triggering of the sweep event.
EXTernal	This choice enables the triggering of a sweep event by an externally applied signal at the TRIGGER IN connector.
KEY	This choice enables triggering by pressing the front-panel **Trigger** hardkey.

Example

```
:LIST:TRIG:SOUR BUS
```

The preceding example sets the trigger source to the instrument BUS.

***RST** IMM

Key Entry **Bus** **Free Run** **Ext** **Trigger Key**

:LIST:TYPE

Supported All Models

```
[:SOURce]:LIST:TYPE LIST|STEP
[:SOURce]:LIST:TYPE?
```

This command selects the sweep type.

LIST	This type of sweep has arbitrary frequencies and amplitudes.
STEP	This type of sweep has equally spaced frequencies and amplitudes.

Example

```
:LIST:TYPE LIST
```

The preceding example selects list as the sweep type.

***RST** STEP

Key Entry **Sweep Type List Step**

:LIST:TYPE:LIST:INITialize:FSTep

Supported All Models

CAUTION When you execute this command, the current list sweep data is overwritten. If needed, save the current data. For information on storing list sweep files, see":STORe:LIST" on page 65.

[:SOURce]:LIST:TYPE:LIST:INITialize:FSTep

This command replaces the loaded list sweep data with the settings from the current step sweep data points. You can have only one sweep list at a time.

The maximum number of list sweep points is 1,601. When copying the step sweep settings over to a list sweep, ensure that the number of points in the step sweep do not exceed the maximum list sweep points.

Key Entry **Load List From Step Sweep**

:LIST:TYPE:LIST:INITialize:PRESet

Supported All Models

CAUTION When you execute this command, the current list sweep data is overwritten. If needed, save the current data. For information on storing list sweep files, see ":STORe:LIST" on page 65.

[:SOURce]:LIST:TYPE:LIST:INITialize:PRESet

This command replaces the current list sweep data with a factory-defined file consisting of one point at a frequency, amplitude, and dwell time.

Key Entry **Preset List**

:SWEep:CONTrol:STATe

Supported All Models with Option 007

[:SOURce]:SWEep:CONTrol:STATe ON|OFF|1|0

[:SOURce]:SWEep:CONTrol:STATe?

This command sets the sweep control state for a PSG in a dual-PSG ramp sweep setup. When the sweep control is on, you can designate whether the PSG is operating as the master or the slave. For information on setting master and slave designations, see ":SWEep:CONTrol:TYPE" on page 146.

The dual-PSG ramp sweep setup uses a serial cable to connect the two signal generators. This connection enables one PSG to function as the master so that sweep, bandcross, and retrace times are synchronized between the two. Each PSG can have a different sweep range, but they must have identical sweep time settings.

Example

```
:SWE:CONT:STAT 1
```

The preceding example sets the sweep control state to on.

***RST** 0

Key Entry **Sweep Control**

:SWEep:CONTrol:TYPE

Supported All Models with Option 007

```
[:SOURce]:SWEep:CONTrol:TYPE MASTer|SLAVe
[:SOURce]:SWEep:CONTrol:TYPE?
```

In a dual-PSG ramp sweep setup, this command designates whether the PSG is performing as the master or the slave. The master/slave setup requires two signal generators from the same instrument family. Refer to the *PSG User's Guide* for more information.

MASTer This choice enables the PSG to provide the triggering.

SLAVe This choice causes the PSG to submit to the triggering parameters provided by the master PSG. You must set the slave PSG triggering to continuous ":INITiate:CONTinuous[:ALL]" on page 114.

Example

```
:SWE:CONT:TYPE MAST
```

The preceding example sets the PSG as the master sweep control instrument.

***RST** 0

Key Entry **Master** or **Slave**

:SWEep:DWELl

Supported All Models

```
[:SOURce]:SWEep:DWELl <val>
[:SOURce]:SWEep:DWELl?
```

This command enables you to set the dwell time for a step sweep.

The variable `<val>` is expressed in seconds with a 0.001 resolution.

NOTE The dwell time `<val>` does not begin until the signal generator has settled for the current frequency and/or amplitude change.

Dwell time is used when the trigger source is set to `IMMediate.`
For the trigger setting, refer to ":LIST:TRIGger:SOURce" on page 143.

The dwell time is the amount of time the sweep pauses after setting the frequency or power, or both, for the current point.

Example

```
:SWE:DWEL .1
```

The preceding example sets the dwell time for a step sweep to 100 milliseconds.

***RST**	+2.00000000E–003
Range	0.001–60S
Key Entry	**Step Dwell**

:SWEep:GENeration

Supported	All Models with Option 007

```
[:SOURce]:SWEep:GENeration ANALog|STEPped
[:SOURce]:SWEep:GENeration?
```

This command sets the sweep type to analog or stepped.

ANALog	This choice selects a ramp sweep.
STEPped	This choice selects a step sweep.

Example

```
:SWE:GEN STEP
```

The preceding example selects a step sweep.

***RST**	STEP
Key Entry	**Sweep Type**

:SWEep:MODE

Supported	All Models with Option 007

```
[:SOURce]:SWEep:MODE AUTO|MANual
[:SOURce]:SWEep:MODE?
```

This command sets the current ramp sweep operating mode.

AUTO	This choice enables the signal generator to automatically sweep through the selected frequency range.
MANual	This choice enables you to select a single frequency value within the current sweep range to control the RF output. For information on selecting the frequency value, see ":FREQuency:MANual" on page 128.

Example

```
:SWE:MODE AUTO
```

The preceding example sets the signal generator to automatically complete a sweep.

***RST** AUTO

Key Entry **Manual Mode Off On**

:SWEep:POINts

Supported All Models

```
[:SOURce]:SWEep:POINts <val>
[:SOURce]:SWEep:POINts?
```

This command enables you to define the number of points in a step sweep.

Example

```
:SWE:POIN 2001
```

The preceding example sets the number of step sweep points to 2001.

***RST** 2
Range 2–65535
Key Entry **# Points**

:SWEep:TIME

Supported All Models with Option 007

```
[:SOURce]:SWEep:TIME <val><units>
[:SOURce]:SWEep:TIME?
```

This command enables you to set the sweep time for a ramp sweep in seconds. If this command is executed while the signal generator is in automatic sweep time mode, the manual sweep time mode is activated and the new sweep time value is applied. The sweep time cannot be set to a value faster than what the automatic mode provides.

The sweep time is the duration of the sweep from the start frequency to the stop frequency. It does not include the bandcross time that occurs during a sweep or the retrace time that occurs between sweep repetitions.

Example

```
:SWE:TIME .250
```

The preceding example sets the ramp sweep time to 250 milliseconds.

***RST**	1.00000000E–002
Range	10mS–99S
Key Entry	**Sweep Time**

:SWEep:TIME:AUTO

Supported	All Models with Option 007

```
[:SOURce]:SWEep:TIME:AUTO ON|OFF|0|1
[:SOURce]:SWEep:TIME:AUTO?
```

This command enables you to set the sweep time mode for a ramp sweep.

The sweep time is the duration of the sweep from the start frequency to the stop frequency. It does not include the bandcross time that occurs during a sweep or the retrace time that occurs between sweep repetitions.

ON (1)	This choice enables the signal generator to automatically calculate and set the fastest allowable sweep time.
OFF (0)	This choice enables you to select the sweep time. The sweep time cannot be set to a value faster than what the automatic mode provides. To set the sweep time refer to“:SWEep:TIME” on page 148.

Example

```
:SWE:TIME:AUTO 0
```

The preceding example sets the ramp sweep time to manual allowing you to select a sweep time.

***RST**	1
Key Entry	**Sweep Time Manual Auto**

Marker Subsystem–Option 007 ([:SOURce])

:MARKer:AMPLitude[:STATe]

Supported All Models with Option 007

```
[:SOURce]:MARKer:AMPLitude[:STATe] ON|OFF|1|0
[:SOURce]:MARKer:AMPLitude[:STATe]?
```

This command sets the amplitude marker state for the currently activated markers. When the state is switched on, the RF output signal exhibits a spike with a magnitude relative to the power level at each marker's set frequency. (To set the magnitude of the spike, refer to ":MARKer:AMPLitude:VALue" on page 150.) The width of the amplitude spike is a nominal eight buckets, based on 1601 buckets per sweep.

Example

```
:MARK:AMPL ON
```

The preceding example enables amplitude markers.

***RST** 0

Key Entry **Amplitude Markers Off On**

:MARKer:AMPLitude:VALue

Supported All Models with Option 007

```
[:SOURce]:MARKer:AMPLitude:VALue <num>[DB]
[:SOURce]:MARKer:AMPLitude:VALue?
```

This command sets the relative power for the amplitude spikes at each marker's set frequency when the amplitude marker mode is activated. (To activate the amplitude markers, refer to ":MARKer:AMPLitude[:STATe]" on page 150.)

Example

```
:MARK:AMPL:VAL 4DB
```

The preceding example sets the relative marker power to 4 dB for all markers.

***RST** 2DB

Range –10DB to +10DB

Key Entry **Marker Value**

:MARKer:AOFF

Supported All Models with Option 007

```
[:SOURce]:MARKer:AOFF
```

This command turns off all active markers.

Key Entry **Turn Off Markers**

:MARKer:DELTa?

Supported All Models with Option 007

```
[:SOURce]:MARKer:DELTa? <num>,<num>
```

This query returns the frequency difference between two amplitude markers. The variables `<num>` are used to designate the marker numbers.

Example

```
:MARK:DELT? 1,2
```

The preceding example returns the frequency difference between amplitude markers 1 and 2.

Range 0–9

:MARKer[0,1,2,3,4,5,6,7,8,9]:FREQuency

Supported All Models with Option 007

```
[:SOURce]:MARKer[0,1,2,3,4,5,6,7,8,9]:FREQuency <val><unit>
[:SOURce]:MARKer[0,1,2,3,4,5,6,7,8,9]:FREQuency? MAXimum|MINimum
```

This command sets the frequency for a specific marker. If the marker designator [n] is not specified, marker 0 is the default. The frequency value must be within the current start, stop, frequency sweep range. Using the `MAXimum` or `MINimum` parameters in the query will return the frequency boundary values for the markers.

If the marker frequency mode is set to delta when the query is sent, the returned value is not absolute, but is relative to the reference marker. (See ":MARKer:MODe" on page 152 for more information.)

Example

```
:MARK2:FREQ 10GHZ
```

The preceding example places amplitude marker 2 at 10 GHz.

***RST** +5.25000000E+008

Range Equivalent to current sweep range

Key Entry **Marker Freq**

:MARKer:MODe

Supported All Models with Option 007

```
[:SOURce]:MARKer:MODE FREQuency|DELTa
[:SOURce]:MARKer:MODE?
```

This command sets the frequency mode for all markers.

FREQuency The frequency values for the markers are absolute.

DELTa The frequency values for the markers are relative to the designated reference marker. The reference marker must be designated before this mode is selected. (See :MARKer:REFerence” to select a reference marker.)

Example

```
:MARK:MODE DELT
```

The preceding example sets the marker mode to delta.

***RST** FREQuency

Key Entry **Marker Delta Off On**

:MARKer:REFerence

Supported All Models with Option 007

```
[:SOURce]:MARKer:REFerence <marker>
[:SOURce]:MARKer:REFerence?
```

This command designates the reference marker when using markers in delta mode. The variable <marker> designates the marker number.

Example

```
:MARK:REF 6
```

The preceding example sets marker 6 as the reference marker.

***RST** 0

Range 0–9

Key Entry **Delta Ref Set**

:MARKer[0,1,2,3,4,5,6,7,8,9][:STATe]

Supported All Models with Option 007

```
[:SOURce]:MARKer[0,1,2,3,4,5,6,7,8,9][:STATe] ON|OFF|1|0
[:SOURce]:MARKer[0,1,2,3,4,5,6,7,8,9][:STATe]?
```

This command turns a marker on or off. Marker 0 is the default if the marker designator [n] is not specified.

Example

```
:MARK6 ON
```

The preceding example turns marker 6 on.

***RST** 0
Key Entry **Marker On Off**

Power Subsystem ([:SOURce]:POWer)

:ALC:BANDwidth|BWIDth

Supported All Models

```
[:SOURce]:POWer:ALC:BANDwidth|BWIDth <num>[<freq_suffix>]
[:SOURce]:POWer:ALC:BANDwidth|BWIDth?
```

This command sets the bandwidth of the automatic leveling control (ALC) loop. You can select bandwidths of 100 Hz, 1 kHz, 10 kHz, or 100kHz. If you do not specify one of these exact bandwidths, your entry rounds to the nearest acceptable value. The bandwidth choices for this command are not effective if an internal I/Q source is being used. Refer to the *PSG User's Guide* for information on ALC and bandwidth considerations.

Example

```
:POW:ALC:BWID 1KHZ
```

The preceding example sets the ALC bandwidth to 1 kHz.

***RST** 100.0

Key Entry **ALC BW**

:ALC:BANDwidth|BWIDth:AUTO

Supported All Models

```
[:SOURce]:POWer:ALC:BANDwidth|BWIDth:AUTO ON|OFF|1|0
[:SOURce]:POWer:ALC:BANDwidth|BWIDth:AUTO?
```

This command sets the state of the automatic leveling control (ALC) automatic bandwidth function. When this state is turned on, the signal generator automatically selects the optimum bandwidth for the ALC.

Example

```
:POW:ALC:BWID:AUTO 0
```

The preceding example disables the automatic bandwidth optimizing function.

***RST** 1

Key Entry **ALC BW**

:ALC:LEVel

Supported E8257D with Option 1E1 and E8267D

```
[:SOURce]:POWer:ALC:LEVel <value>DB
[:SOURce]:POWer:ALC:LEVel?
```

This command sets the automatic leveling control (ALC) level when the attenuator hold is active.

Use this command when the automatic attenuation mode is set to OFF (0). Refer to ":ATTenuation:AUTO" on page 160 for choosing the attenuator mode.

Example

```
:POW:ALC:LEV 10DB
```

The preceding example sets the ALC to 10 dB.

***RST** +1.00000000E+000

Range -20 to 25

Key Entry **Set ALC Level**

:ALC:SEARch

Supported All Models

```
[:SOURce]:POWer:ALC:SEARch ON|OFF|1|0|ONCE
[:SOURce]:POWer:ALC:SEARch?
```

This command enables or disables the internal power search calibration. A power search is recommended for pulse-modulated signals with pulse widths less than one microsecond. Refer to the *PSG User's Guide* for more information on ALC and the power search function.

ON (1) This choice executes the power search automatically with each change in RF frequency or power.

OFF (0) This choice disables the automatic power search routine.

ONCE This choice executes a single power search of the current RF output signal.

Use this command when the automatic leveling control (ALC) state is set to OFF (0). Refer to ":ALC[:STATe]" on page 159 for setting the ALC state.

If ON was previously selected, executing ONCE will cause OFF to be the current selection after the power search is completed.

Example

```
:POW:ALC:SEAR ONCE
```

The preceding example starts a single power search of the RF output signal.

***RST** 0

Key Entry **Power Search Manual Auto** **Do Power Search**

:ALC:SEARch:REFerence

Supported All Models

```
[:SOURce]:POWer:ALC:SEARch:REFerence FIXed|MODulated
[:SOURce]:POWer:ALC:SEARch:REFerence?
```

This command sets either fixed or modulated modes for power search.

FIXed This choice uses a 0.5 volt reference.

MODulated This choice uses the RMS value of the current I/Q modulation as measured during the power search.

Example

```
:POW:ALC:SEAR:REF FIX
```

The preceding example selects a fixed voltage as the reference for a power search.

***RST** MOD

Key Entry **Power Search Reference Fixed Mod**

:ALC:SEARch:SPAN:START

Supported All Models

```
[:SOURce]:POWer:ALC:SEARch:SPAN:START <val><units>
[:SOURce]:POWer:ALC:SEARch:SPAN:START?
```

This command sets the start frequency for a power search over a user-defined range. The start frequency has no default value. The start frequency value will be set before powering off the instrument.

Example

```
:POW:ALC:SEAR:SPAN:START 12GHZ
```

The preceding example selects 12 GHz as the start frequency for a power search.

Key Entry **Start Frequency**

:ALC:SEARch:SPAN:STOP

Supported All Models

```
[:SOURce]:POWer:ALC:SEARch:SPAN:STOP <val><units>
[:SOURce]:POWer:ALC:SEARch:SPAN:STOP?
```

This command sets the stop frequency for a power search over a user-defined range. The stop frequency has no default value. The stop frequency value will be set before powering off the instrument

Example

```
:POW:ALC:SEAR:SPAN:STOP 20GHZ
```

The preceding example selects 20 GHz as the stop frequency for a power search.

Key Entry **Stop Frequency**

:ALC:SEARch:SPAN:TYPE FULL|USER

Supported All Models

```
[:SOURce]:POWer:ALC:SEARch:SPAN:TYPE FULL|USER
[:SOURce]:POWer:ALC:SEARch:SPAN:TYPE?
```

This command enables you to select the frequency range for a power search. You can specify the range (`USER`) or you can select the full range (`FULL`) of the signal generator.

Example

```
:POW:ALC:SEAR:SPAN:TYPE USER
```

The preceding example selects a user-defined frequency range for the power search.

Key Entry **Span Type User Full**

:ALC:SEARch:SPAN[:STATe] ON|OFF|1|0

Supported All Models

```
[:SOURce]:POWer:ALC:SEARch:SPAN[:STATe] ON|OFF|1|0
[:SOURce]:POWer:ALC:SEARch:SPAN[:STATe]?
```

This command enables (1) or disables (0) the span mode, allowing you to perform power searches over a selected range of frequencies. The power search corrections are then stored and used whenever the signal generator is tuned within the selected range.

Example

`:POW:ALC:SEAR:SPAN ON`

The preceding example enables the span mode.

:ALC:SOURce

Supported All Models

```
[:SOURce]:POWer:ALC:SOURce INTernal|DIODe|MMHead
[:SOURce]:POWer:ALC:SOURce?
```

This command enables you to select an automatic level control (ALC) source. You can select the internal ALC source, an external detector source, or a millimeter-wave source module. Refer to the *PSG User's Guide* for more information on ALC leveling, bandwidth, and the power search function.

Example

`:POW:ALC:SOUR MMH`

The preceding example selects an Agilent 8355x series external millimeter head as the source (the unit must be connected to the signal generator).

***RST** INT

Key Entry **Leveling Mode**

:ALC:SOURce:EXTernal:COUPling

Supported All Models

```
[:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling <value>DB
[:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling?
```

This command sets the external detector coupling factor. Use this command when DIODe is the selected ALC source (":ALC:SOURce" on page 158). (0 to 32 coupling value).

Example

`:POW:ALC:SOUR:EXT:COUP 20DB`

The preceding example sets the external coupling factor to 20 dB.

***RST** +1.60000000E+001

Range –200DB to 200DB.

Key Entry **Ext Detector Coupling Factor**

:ALC[:STATe]

Supported All Models

```
[:SOURce]:POWer:ALC[:STATe] ON|OFF|1|0
[:SOURce]:POWer:ALC[:STATe]?
```

This command enables or disables the automatic leveling control (ALC) circuit. The purpose of the ALC circuit is to hold output power at a desired level by adjusting the signal generator power circuits for power drift. Power will drift over time and with changes in temperature. Refer to the *PSG User's Guide* for more information on the ALC.

Example

```
:POW:ALC ON
```

The preceding example sets the ALC on.

***RST** 1

Key Entry **ALC Off On**

:ATTenuation

Supported E8257D with Option 1E1 and E8267D

```
[:SOURce]:POWer:ATTenuation <val><unit>
[:SOURce]:POWer:ATTenuation?
```

This command sets the attenuation level when the attenuator hold is active. For the E8267D, the attenuation is set in increments of 5 dB. For the E8257D with Option 1E1, the progression is 0, 5, 15, 25 and continues in 5 dB increments.

The output power is the ALC level minus the attenuator setting.

Use this command when the automatic attenuation mode is set to OFF (0). Refer to ":ATTenuation:AUTO" on page 160 for choosing the attenuator mode.

Example

```
:POW:ATT 10DB
```

The preceding example sets the attenuator to 10 dB.

***RST** +115

Range 0 to 115 dB

Key Entry **Set Atten**

:ATTenuation:AUTO

Supported E8257D with Option 1E1 and E8267D

```
[:SOURce]:POWer:ATTenuation:AUTO ON|OFF|1|0
[:SOURce]:POWer:ATTenuation:AUTO?
```

This command sets the state of the attenuator hold function.

ON (1) This choice enables the attenuator to operate normally.

OFF (0) This choice holds the attenuator at its current setting or at a selected value that will not change during power adjustments.

OFF (0) eliminates the power discontinuity normally associated with the attenuator switching during power adjustments. During an amplitude sweep operation, signal generators with Option 1E1 protect the step attenuator by automatically switching to attenuator hold mode (ON). The attenuator is locked at its current setting and the amplitude sweep range is limited to 40 dB.

Example

```
:POW:ATT:AUTO OFF
```

The preceding example turns off the attenuator hold function.

***RST** 1

Key Entry **Atten Hold Off On**

:MODE

Supported All Models

```
[:SOURce]:POWer:MODE FIXed|SWEep|LIST
[:SOURce]:POWer:MODE?
```

This command starts or stops an amplitude sweep and sets the power mode of the signal generator.

FIXed This choice stops a power sweep and allows the signal generator to operate at a fixed power level. Refer to “[:LEVel][:IMMediate][:AMPLitude]” on page 164 for more information on running power sweeps and setting CW amplitude settings that control the output power.

SWEep The effects of this choice are determined by the sweep generation type selected (refer to “:SWEep:GENeration” on page 147). If you are using analog sweep generation, the current ramp sweep amplitude settings (start and stop) control the output power. If you are using step sweep generation, the current step sweep amplitude settings control the output power. In both cases, this selection also activates the sweep. This choice is available with Option 007 only.

LIST This choice selects the swept power mode. If sweep triggering is set to immediate along with continuous sweep mode, executing the command starts the LIST or STEP frequency sweep.

NOTE To perform a frequency and amplitude sweep, you must also select LIST or SWEep as the frequency mode (see ":FREQuency:MODE" on page 128).

Example

```
:POW:MODE LIST
```

The preceding example sets list as the amplitude sweep mode.

***RST** FIX

Key Entry **Sweep Type** **Ampl** **Off** **Freq & Ampl**

:PROTection:STATe

Supported E8257D with Option 1E1 and E8267D

```
[:SOURce]:POWer:PROTection[:STATe] ON|OFF|1|0
[:SOURce]:POWer:PROTection[:STATe]?
```

This command enables or disables the power search protection function. The power search protection function sets the attenuator to its maximum level whenever a power search is initiated. This can be used to protect devices that are sensitive to high average power or high power changes. The trade off on using the power protection function is decreased attenuator life, as the attenuator will switch to its maximum setting during a power search.

NOTE Continual or excessive use of the power search protection function can decrease attenuator life.

ON (1) Causes the attenuator to switch to and hold its maximum level setting during a power search.

OFF (0) Sets the attenuator normal mode. The attenuator is not used during power search.

Example

```
:POW:PROT ON
```

The preceding example enables the power inhibit function.

***RST** 0

Key Entry **RF During Power Search Normal Minimum**

:REFerence

Supported All Models

```
[:SOURce]:POWer:REFerence <val><unit>
[:SOURce]:POWer:REFerence?
```

This command sets the power level for the signal generator RF output reference. The RF output power is referenced to the value entered in this command.

Example

```
:POW:REF 50DBM
```

The preceding example sets the RF output power reference to 50 dBm.

***RST**	+0.00000000E+000
Range	-400 to 300 dBm
Key Entry	**Ampl Ref Set**

:REFerence:STATe

Supported All Models

```
[:SOURce]:POWer:REFerence:STATe ON|OFF|1|0
[:SOURce]:POWer:REFerence:STATe?
```

This command enables or disables the RF output reference.

ON (1)	Sets the power reference state ON. dB is the unit displayed for commands (“:ANNotation:AMPLitude:UNIT” on page 34 and “:POWer” on page 118).
OFF (0)	Sets the power reference state OFF.

Once the reference state is ON, all subsequent output power settings are set relative to the reference value. Amplitude offsets can be used with the amplitude reference mode.

Example

```
:POW:REF:STAT 1
```

The preceding example sets the reference state on.

***RST**	0
Key Entry	**Ampl Ref Off On**

:STARt

Supported All Models

```
[:SOURce]:POWer:STARt <val><unit>
[:SOURce]:POWer:STARt?
```

This command sets the amplitude of the first point in a step or ramp sweep (Option 007).

During an amplitude sweep operation, signal generators with Option 1E1 protect the step attenuator by automatically switching to attenuator hold (ON) mode. The attenuator is locked at its current setting and the amplitude sweep range is limited to 40 dB.

Example

```
:POW:STAR -30DBM
```

The preceding example sets the amplitude of the first point in the sweep to –30 dBm.

***RST** Depends on model and option number

Range Refer to "[:LEVel][:IMMediate][:AMPLitude]" on page 164 for the output power ranges.

Key Entry **Ampl Start**

:STOP

Supported All Models

```
[:SOURce]:POWer:STOP <val><unit>
[:SOURce]:POWer:STOP?
```

This command sets the amplitude of the last point in a step or ramp sweep (Option 007).

During an amplitude sweep, signal generators with Option 1E1 protect the step attenuator by switching to attenuator hold (ON) mode. The attenuator is locked at its current setting and the amplitude sweep range is limited to 40 dB.

Example

```
:POW:STOP -10DBM
```

The preceding example sets the amplitude of the last point in the sweep to –10 dBm.

***RST** Depends on model and option number.

Range See "[:LEVel][:IMMediate][:AMPLitude]" on page 164 for the available power ranges.

Key Entry **Ampl Stop**

[:LEVel][:IMMediate]:OFFSet

Supported All Models

```
[:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet <val><unit>
[:SOURce]:POWer[:LEVel][:IMMediate]:OFFSet?
```

This command sets the power offset value as a dB power offset to the actual RF output. This simulates a power level at a test point beyond the RF OUTPUT connector without changing the actual RF output power. The offset value only affects the displayed amplitude setting.

You can enter an amplitude offset anytime in either normal operation or amplitude reference mode.

Example

```
:POW:OFFS 10DB
```

The preceding example sets the amplitude offset to 10 dB.

***RST** +0.00000000E+000

Range –200dB to 200dB

Key Entry **Ampl Offset**

[:LEVel][:IMMediate][:AMPLitude]

Supported All Models

```
[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] <val><unit>
[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]?
```

This command sets the RF output power.

The ranges for this command are specified values from the data sheet.

Example

```
:POW 0DBM
```

The preceding example sets the signal generator output power level to 0 dBm.

***RST** Depends on model and option number

Range

E8257D

Option	Standard	*with Option UNW*	*with Option 1EH*	*with Option UNW & 1EH*
Option 520				
250 kHz–3.2 GHz	–20 to 13DBM	–20 to 13DBM	–20 to 13DBM	–20 to 10DBM
>3.2 GHz–20 GHz	–20 to 13DBM	–20 to 13DBM	–20 to 13DBM	–20 to 13DBM

Option	Standard	*with Option UNW*	*with Option 1EH*	*with Option UNW & 1EH*
Option 540				
250 kHz–40 GHz	–20 to 9DBM	–20 to 9DBM	–20 to 9DBM	–20 to 9DBM
Option 550 and 567				
250 kHz–70 GHz	–20 to 5DBM	–20 to 5DBM	–20 to 5DBM	–20 to 5DBM

E8257D with Option 1EA

Option	*Option 1EA only*	*with Option UNW*	*with Option 1EH*	*with Option UNW & 1EH*
Option 520				
250 kHz–3.2 GHz	–20 to 16DBM	–20 to 13DBM	–20 to 13DBM	–20 to 10DBM
>3.2 GHz–20 GHz	–20 to 13DBM	–20 to 13DBM	–20 to 13DBM	–20 to 20DBM
Option 540				
250 kHz–3.2 GHz	–20 to 15DBM	–20 to 12DBM	–20 to 12DBM	–20 to 9DBM
>3.2 GHz–20 GHz	–20 to 18DBM	–20 to 18DBM	–20 to 18DBM	–20 to18DBM
>20 GHz–40 GHz	–20 to 14DBM	–20 to 14DBM	–20 to 14DBM	–20 to 14DBM
Option 550 and 567				
250 kHz–3.2 GHz	–20 to 14DBM	–20 to 11DBM	–20 to 11DBM	–20 to 8DBM
>3.2 GHz–20 GHz	–20 to 16DBM	–20 to 16DBM	–20 to 16DBM	–20 to16DBM
>20 GHz–65 GHz	–20 to 13DBM	–20 to 13DBM	–20 to 13DBM	–20 to13DBM
>65 GHz–67 GHz	–20 to 12DBM	–20 to 12DBM	–20 to 12DBM	–20 to12DBM
>67 GHz–70 GHz[a]	–20 to 11DBM	–20 to 11DBM	–20 to 11DBM	–20 to 11DBM

a. 67-70 GHz performance not specified

E8257D with Option 1E1

Option	*Option 1E1 only*	*with Option UNW*	*with Option 1EH*	*with Option UNW & 1EH*
Option 520				
250 kHz–3.2 GHz	–135 to 11DBM	–135 to 11DBM	–135 to 11DBM	–135 to 9DBM
>3.2GHz–20 GHz	–135 to 11DBM	–135 to 11DBM	–135 to 11DBM	–135 to 11DBM
Option 540				
250 kHz–40 GHz	–135 to 7DBM	–135 to 7DBM	–135 to 7DBM	–135 to 7DBM
Option 550 and 567				
250 kHz–67 GHz	–110 to 3DBM	–110 to 3DBM	–110 to 3DBM	–110 to 3DBM
>67 GHz–70 GHz[a]	–110 to 3DBM	–110 to 3DBM	–110 to 3DBM	–110 to 3DBM

a. 67-70 GHz performance not specified

E8257D with Option 1EA and 1E1

Option	*Option 1EA &1E1 only*	*with Option UNW*	*with Option 1EH*	*with Option UNW & 1EH*
Option 520				
250 kHz–3.2 GHz	–135 to 15DBM	–135 to 12DBM	–135 to 12DBM	–135 to 9DBM
>3.2 GHz–20 GHz	–135 to 18DBM	–135 to 18DBM	–135 to 18DBM	–135 to 18DBM
Option 540				
250 kHz–3.2 GHz	–135 to 14DBM	–135 to 11DBM	–135 to 11DBM	–135 to 8DBM
>3.2 GHz–20 GHz	–135 to 16DBM	–135 to 16DBM	–135 to 16DBM	–135 to 16DBM
>20 GHz–40 GHz	–135 to 12DBM	–135 to 12DBM	–135 to 12DBM	–135 to 12DBM

Option	Option 1EA &1E1 only	with Option UNW	with Option 1EH	with Option UNW & 1EH
Option 550 and 567				
250 kHz–3.2 GHz	–110 to 13DBM	–110 to 10DBM	–110 to 10DBM	–110 to 7DBM
>3.2 GHz–20 GHz	–110 to 14DBM	–110 to 14DBM	–110 to 14DBM	–110 to 14DBM
>20 GHz–65 GHz	–110 to 9DBM	–110 to 9DBM	–110 to 9DBM	–110 to 9DBM
>65 GHz–67 GHz	–110 to 8DBM	–110 to 8DBM	–110 to 8DBM	–110 to 8DBM
>67 GHz–70 GHz[a]	–110 to 6DBM	–110 to 6DBM	–110 to 6DBM	–110 to 6DBM

a. 67-70 GHz performance not specified

E8267D

Option	Standard	with Option UNW	with Option 1EH	with Option UNW & 1EH
Option 520				
250 kHz–3.2 GHz	–130 to 13DBM	–130 to 10DBM	–130 to 10DBM	–130 to 7DBM
>3.2GHz–20 GHz	–130 to 18DBM	–130 to 18DBM	–135 to 18DBM	–135 to 18DBM
Option 532 and 544				
250 kHz–3.2 GHz	–130 to 12DBM	–130 to 9DBM	–135 to 9DBM	–135 to 6DBM
>3.2 GHz–32 GHz	–130 to 14DBM	–130 to 14DBM	–135 to 14DBM	–135 to 14DBM
>32 GHz–40 GHz	–130 to 12DBM	–130 to 12DBM	–130 to 12DBM	–135 to 12DBM
>40 GHz–44 GHz	–130 to 10DBM	–130 to 10DBM	–130 to 10DBM	–135 to 10DBM

Key Entry **Amplitude**

Trigger Sweep Subsystem ([:SOURce])

:TSWeep

Supported All Models

`[:SOURce]:TSWeep`

This command aborts the current sweep, then either arms or arms and starts a single list, step, or ramp sweep (Option 007), depending on the trigger type.

The command performs the following:

- arms a single sweep when BUS, EXTernal, or KEY is the trigger source selection
- arms and starts a single sweep when IMMediate is the trigger source selection

Key Entry **Single Sweep**

4 Analog Commands

This chapter provides SCPI descriptions for subsystems dedicated to E8257D PSG Analog and E8267D PSG Vector signal generators. The following is a list of the subsystems:

Amplitude Subsystem ([:SOURce])

:AM[1]|2...

Supported	E8257D and E8267D

`[:SOURce]:AM[1]|2...`

This prefix enables the selection of the AM path and is part of most SCPI commands associated with this subsystem. The two paths are equivalent to the **AM Path 1 2** softkey.

AM1	**AM Path 1 2** with 1 selected
AM2	**AM Path 1 2** with 2 selected

When just AM is shown in a command, the command defaults to path 1.

Each path is set up separately. When a SCPI command uses AM1, only path one is affected. Consequently, when AM2 is selected, only path two is set up. However, the depth of the signals for the two paths can be coupled.

The two AM paths can be on at the same time provided the following conditions have been met:

- dual–sine or swept–sine is not one of the selections for the waveform type
- Each path uses a different source (Internal 1, Internal 2, Ext1, or Ext2)

:AM:INTernal:FREQuency:STEP[:INCRement]

Supported E8257D and E8267D

`[:SOURce]:AM:INTernal:FREQuency:STEP[:INCRement]<num>`|MAXimum|MINimum|DEFault
`[:SOURce]:AM:INTernal:FREQuency:STEP[:INCRement]?`

This command sets the step value for the AM internal frequency.

The step value set by this command is used with the UP and DOWN choices for the :AM[1]|2:INTernal[1]|2:FREQuency command described on page 174.

The step value set with this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

`:AM:INT:FREQ:STEP 1E3`

The preceding example sets the step size to 1000 hertz.

Range 0.5–1E6

Key Entry **Incr Set**

:AM:MODE

Supported All with Option UNT

`[:SOURce]:AM:MODE DEEP|NORMal`
`[:SOURce]:AM:MODE?`

This command sets the mode for amplitude modulation.

DEEP This choice enables amplitude modulation depth with a greater dynamic range than normal mode which utilizes the ALC. DEEP has no specified parameters and emulates the amplitude modulation NORMal mode with the ALC disabled.

NORMal This choice maintains the amplitude modulation standard behavior and has specified parameters as outlined in the data sheet.

The ALC is disabled when the carrier amplitude is less than –10 dBm and DEEP is the AM mode.

DEEP is limited to repetitive AM and will not work with a dc modulation signal.

Example

`:AM:MODE NORM`

The preceding example selects the normal mode for amplitude modulation.

***RST** NORM

Key Entry **AM Mode Normal Deep**

:AM:WIDeband:SENSitivity

Supported E8267D and Option UNT

```
[:SOURce]:AM:WIDeband:SENSitivity <val>
[:SOURce]:AM:WIDeband:SENSitivity?
```

This command sets the sensitivity level of the wideband AM signal in units of dB/volt. Sensitivity is .5V = 100% and is linear with .25V = 50%. Wideband AM uses input from the front panel I INPUT.

Example

```
:AM:WID:SENS 20
```

The preceding example sets the sensitivity level to 20%.

***RST** +2.00000000E+001

Range 0 – 40DB

Key Entry **AM Depth**

:AM:WIDeband:STATe

Supported E8267D with Option UNT

```
[:SOURce]:AM:WIDeband:STATe ON|OFF|1|0
[:SOURce]:AM:WIDeband:STATe?
```

This command enables or disables wideband amplitude modulation. The RF carrier is modulated when the signal generator's modulation state is ON, see ":MODulation[:STATe]" on page 74 for more information. The signal generator's I input is used to drive wideband AM modulation.

Whenever wideband amplitude modulation is enabled, the AM annunciator appears on the signal generator's front panel display. Wideband amplitude modulation can be simultaneously enabled with AM paths 1 and 2. Refer to ":AM[1]|2..." on page 170 for more information.

Example

```
:AM:WID:STAT 0
```

The preceding example turns off wideband amplitude modulation.

***RST** 0

Key Entry **AM Off On**

:AM[1]|2:EXTernal[1]|2:COUPling

Supported All

```
[:SOURce]:AM[1]|2:EXTernal[1]|2:COUPling AC|DC
[:SOURce]:AM[1]|2:EXTernal[1]|2:COUPling?
```

This command sets the coupling type for the selected external input. The command does not change the active source or switch the modulation on or off. The modulating signal may be the sum of several signals, with either internal or external sources.

AC This choice will pass only ac signal components.

DC This choice will pass both ac and dc signal components.

Example

```
:AM1:EXT1:COUP AC
```

The preceding example sets the AM path 1, external 1 source coupling to AC.

***RST** DC

Key Entry **Ext Coupling DC AC**

:AM[1]|2:EXTernal[1]|2:IMPedance

Supported All

```
[:SOURce]:AM[1]|2:EXTernal[1]|2:IMPedance <50|600>
[:SOURce]:AM[1]|2:EXTernal[1]|2:IMPedance?
```

This commands sets the impedance for the external input.

Example

```
:AM1:EXT1:IMP 600
```

The preceding example sets the AM path 1, external 1 source impedance to 600 ohms.

***RST** +5.00000000E+001

Key Entry **Ext Impedance 50 Ohm 600 Ohm**

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

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:INTernal[1]|2:FREQuency <val><units>|UP|DOWN
[:SOURce]:AM[1]|2:INTernal[1]|2:FREQuency?
```

This command sets the internal AM rate using the variable `<val><units>`. The command, used with the `UP|DOWN` parameters, will change the frequency rate by a user-defined step value. Refer to the :PULM:INTernal[1]:FREQuency:STEP command on page 174 for setting the value associated with the UP and DOWN choices.

The command changes:

- the frequency rate of the first tone of a dual–sine waveform
- the start frequency for a swept–sine waveform
- the AM frequency rate for all other waveforms

Refer to ":AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 176 for the waveform selection.

Example

```
:AM1:INT2:FREQ UP
```

The preceding example increases the modulation rate for AM path 1, AM internal source 2 by the step value set with the :AM:INTernal:FREQuency:STEP[:INCRement] command described on page 171.

***RST**	+4.00000000E+002	
Range	*Dual-Sine & Sine*: 0.5HZ–1MHZ	*Swept-Sine*: 1HZ–1MHZ
	All Other Waveforms: 0.5HZ–100kHZ	
Key Entry	**AM Tone 1 Rate** **AM Start Rate**	**AM Rate**

:AM[1]|2:INTernal[1]:FREQuency:ALTernate

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:INTernal[1]:FREQuency:ALTernate <val><units>
[:SOURce]:AM[1]|2:INTernal[1]:FREQuency:ALTernate?
```

This command sets the frequency for the alternate signal. The alternate signal frequency is the second tone of a dual–sine or the stop frequency of a swept–sine waveform.

Refer to ":AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 176 for the waveform selection.

Example

```
:AM2:INT1:FREQ:ALT 500KHZ
```

The preceding example sets the alternate frequency (AM path 2, AM internal source 1) for AM tone 2 to 500 kHz.

***RST**	+4.00000000E+002	
Range	*Dual-Sine*: 0.5HZ–1MHZ	*Swept-Sine*: 1HZ–1MHZ
Key Entry	**AM Tone 2 Rate**	**AM Stop Rate**

:AM[1]|2:INTernal[1]:FREQuency:ALTernate:AMPLitude:PERCent

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:INTernal[1]:FREQuency:ALTernate:AMPLitude:
PERCent <val>
[:SOURce]:AM[1]|2:INTernal[1]:FREQuency:ALTernate:AMPLitude:PERCent?
```

This command sets the amplitude of the second tone for a dual–sine waveform as a percentage of the total amplitude. For example, if the second tone makes up 30% of the total amplitude, then the first tone is 70% of the total amplitude.

Refer to “:AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe” on page 176 for the waveform selection.

Example

```
:AM2:INT1:FREQ:ALT:AMPL:PERC 50
```

The preceding example sets the amplitude (AM path 2, AM internal source 1) for AM tone 2 to 50% of the total amplitude.

***RST**	+5.00000000E+001
Range	0–100PCT
Key Entry	**AM Tone 2 Ampl Percent Of Peak**

:AM[1]|2:INTernal[1]|2:FUNCtion:NOISe

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:INTernal[1]|2:FUNCtion:NOISe GAUSsian|UNIForm
[:SOURce]:AM[1]|2:INTernal[1]|2:FUNCtion:NOISe?
```

This command selects a gaussian or uniform noise modulation for the selected waveform.

Refer to “:AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe” on page 176 for the waveform selection.

Example

```
:AM2:INT1:FUNC:NOIS GAUS
```

The preceding example selects the gaussian noise waveform for AM modulation on AM path 2, internal source 1.

***RST** UNIF

Key Entry **Gaussian** **Uniform**

:AM[1]|2:INTernal[1]|2:FUNCtion:RAMP

Supported All with Option UNT and Option 007

```
[:SOURce]:AM[1]|2:INTernal[1]|2:FUNCtion:RAMP POSitive|NEGative
[:SOURce]:AM[1]|2:INTernal[1]|2:FUNCtion:RAMP?
```

This command selects a positive or negative slope for the modulating ramp waveform.

Refer to ":AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" for the waveform selection.

Example

```
:AM2:INT1:FUNC:RAMP NEG
```

The preceding example sets the slope of the ramp modulation for AM path 2, internal source 1, to negative.

***RST** POS

Key Entry **Positive** **Negative**

:AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe SINE|TRIangle|SQUare|
RAMP|NOISe|DUALsine|SWEPtsine
[:SOURce]:AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe?
```

This command sets the AM waveform type. The INTernal2 source selection does not support the dual–sine or Sweep–Sine waveform choices.

Example

```
:AM1:INT1:FUNC:SHAP DUAL
```

The preceding example sets the AM waveform type for AM path 1, internal source 1, to dual sine.

***RST** SINE

Key Entry **Sine** **Triangle** **Square** **Ramp** **Noise** **Dual-Sine** **Swept-Sine**

:AM[1]|2:INTernal[1]:SWEep:RATE

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:INTernal[1]:SWEep:RATE <val><units>
[:SOURce]:AM[1]|2:INTernal[1]:SWEep:RATE?
```

This command sets the sweep rate for the AM swept–sine waveform.

Refer to ":AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 176 for the waveform selection. The sweep rate function is only available for internal source 1.

Example

```
:AM2:INT1:SWE:RATE 1KHZ
```

The preceding example sets the sweep rate for AM path 1, internal source 1to 1 kHz.

***RST** +4.00000000E+002

Range 0.5HZ–100kHZ

Key Entry **AM Sweep Rate**

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

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:INTernal[1]:SWEep:TRIGger BUS|IMMediate|EXTernal|KEY
[:SOURce]:AM[1]|2:INTernal[1]:SWEep:TRIGger?
```

This command sets the trigger source for the AM swept–sine waveform.

BUS	This choice enables GPIB triggering using the `*TRG` or `GET` command or LAN triggering using the `*TRG` command.
IMMediate	This choice enables immediate triggering of the sweep event.
EXTernal	This choice enables the triggering of a sweep event by an externally applied signal at the TRIGGER IN connector.
KEY	This choice enables triggering through front panel interaction by pressing the **Trigger** hardkey.

Refer to ":AM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 176 for the waveform selection.

Example

```
:AM1:INT1:SWE:TRIG EXT
```

The preceding example sets an external trigger source for the swept–sine waveform on AM path 1.

***RST** IMM

Key Entry **Bus** **Free Run** **Ext** **Trigger Key**

:AM[1]|2:SOURce

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:SOURce INT[1]|INT2|EXT[1]|EXT2
[:SOURce]:AM[1]|2:SOURce?
```

This command selects the source for amplitude modulation.

INT This choice selects internal source 1 or 2 to provide an ac–coupled signal.

EXT This choice selects the EXT 1 INPUT or the EXT 2 INPUT connector to provide an externally applied signal that can be ac- or dc-coupled. The externally applied, ac-coupled input signal is tested for a voltage level and an annunciator, on the signal generator’s front-panel display, will indicate a high or low condition if that voltage is > ±3% of 1 V_p.

Example

```
:AM2:SOUR INT1
```

The preceding example selects internal source 1 as the source for AM path 2.

***RST** INT

Key Entry **Internal 1** **Internal 2** **Ext1** **Ext2**

:AM[1]|2:STATe

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:STATe ON|OFF|1|0
[:SOURce]:AM[1]|2:STATe?
```

This command enables or disables amplitude modulation for the selected path.

The RF carrier is modulated when you have set the signal generator’s modulation state to ON, see “:MODulation[:STATe]” on page 74 for more information.

Whenever amplitude modulation is enabled, the `AM` annunciator appears on the signal generator’s front-panel display.

The two paths for amplitude modulation can be simultaneously enabled. Refer to “:AM[1]|2...” on page 170 for more information.

Example

```
:AM1:STAT ON
```

The preceding example turns on AM modulation for AM path 1.

***RST** 0

Key Entry **AM Off On**

:AM[1]|2:TYPE

Supported All with Option UNT

```
[:SOURce]:AM[1]|2:TYPE LINear|EXPonential
[:SOURce]:AM[1]|2:TYPE?
```

This command sets the AM type to linear or exponential AM.

LINear This choice selects linear AM type with depth values in units of percent/volt.

EXPonential This choice selects exponential AM type with depth values in units of dB/volt.

Example

```
:AM2:TYPE EXP
```

The preceding example selects exponential type depth values for AM path 2.

***RST** LIN

Key Entry **AM Type LIN EXP**

:AM[1]|2[:DEPTh]:EXPonential

Supported All with Option UNT

```
[:SOURce]:AM[1]|2[:DEPTh]:EXPonential <val>
[:SOURce]:AM[1]|2[:DEPTh]:EXPonential?
```

This commands sets the AM depth in dB/volt units. EXPonential must be the current AM type for this command to have any affect. Refer to “:AM[1]|2:TYPE” for setting the AM type.

Example

```
:AM2:EXP 20
```

The preceding example sets the exponential depth to 20 dB for AM path 2.

***RST** +4.00000000E+001

Range 0.00–40.00DB

Key Entry **AM Depth**

:AM[1]|2[:DEPTh][:LINear]

Supported All with Option UNT

```
[:SOURce]:AM[1]|2[:DEPTh][:LINear] <val>|UP|DOWN
[:SOURce]:AM[1]|2[:DEPTh][:LINear]?
```

This command sets the AM depth in percent/volt units.The command, used with the UP|DOWN parameters, will change the depth by a user-defined step value. Refer to the :AM[:DEPTh]:STEP[:INCRement] command on page 181 for setting the value associated with the UP and DOWN choices.

LINear must be the current AM type for this command to have any affect. Refer to ":AM[1]|2:TYPE" on page 179 for setting the AM measurement type. When the depth values are coupled, a change made to one path is applied to both. For AM depth value coupling, refer to the command ":AM[1]|2[:DEPTh][:LINear]:TRACk" on page 180.

Example

```
:AM2 20
```

The preceding example sets the AM path 2 linear depth to 20%.

***RST**	+1.00000000E-001
Range	0.0–100PCT
Key Entry	**AM Depth**

:AM[1]|2[:DEPTh][:LINear]:TRACk

Supported All with Option UNT

```
[:SOURce]:AM[1]|2[:DEPTh][:LINear]:TRACk ON|OFF|1|0
[:SOURce]:AM[1]|2[:DEPTh][:LINear]:TRACk?
```

This command enables or disables AM depth value coupling between AM paths 1 and 2. When the depth values are coupled, a change made to one path is applied to both. LINear must be the AM type for this command to have any affect. Refer to ":AM[1]|2:TYPE" on page 179 for setting the AM type.

ON (1)	This choice will link the depth value of AM[1] with AM2; AM2 will assume the AM[1] depth value. For example, if AM[1] depth is set to 15% and AM2 is set to 11%, enabling the depth tracking will cause the AM2 depth value to change to 15%. This applies regardless of the path (AM[1] or AM2) selected in this command
OFF (0)	This choice disables coupling and both paths will have independent depth values.

Example

```
:AM1:TRAC ON
```

The preceding example enables AM depth coupling between AM path 1 and AM path 2.

***RST** 0

Key Entry **AM Depth Couple Off On**

:AM[:DEPTh]:STEP[:INCRement]

Supported All with Option UNT

`[:SOURce]:AM[:DEPTh]:STEP[:INCRement] <val>`|MAXimum|MINimum|DEFault
`[:SOURce]:AM[:DEPTh]:STEP[:INCRement]?`

This command sets the linear depth step value in percent/volt units.

The step value set by this command is used with the UP and DOWN choices for the :AM[1]|2[:DEPTh][:LINear] command on page 180.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

`:AM:STEP 10`

The preceding example sets the step value for AM depth to 10%.

Range 0.1–100

Key Entry **Incr Set**

Frequency Modulation Subsystem ([:SOURce])

:FM[1]|2...

Supported	E8257D and E8267D

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

This prefix enables the selection of the FM path and is associated with all SCPI commands in this subsystem. The two paths are equivalent to the **FM Path 1 2** softkey.

FM1	**FM Path 1 2** with 1 selected
FM2	**FM Path 1 2** with 2 selected

When just FM is shown in a command, this means the command applies to path one only.

Each path is set up separately. When a SCPI command uses FM1, only path one is affected. Consequently, when FM2 is selected, only path two is set up. However, the deviation of the signals for the two paths can be coupled.

Deviation coupling links the deviation value of FM1 to FM2. Changing the deviation value for one path changes it for the other. These two paths can be on at the same time provided the following conditions have been met:

- dual–sine or swept–sine is not the selection for the waveform type
- each path uses a different source (Internal 1, Internal 2, Ext1, or Ext2)
- FM2 must be set to a deviation less than FM1

:FM:INTernal:FREQuency:STEP[:INCRement]

Supported All with Option UNT

```
[:SOURce]:FM:INTernal:FREQuency:STEP[:INCRement]<num>|MAXimum|MINimum|DE
Fault
[:SOURce]:FM:INTernal:FREQuency:STEP[:INCRement]?
```

This command sets the step value for the internal frequency modulation.

The step value set by this command is used with the UP and DOWN choices for the command :FM[1]|2:INTernal[1]|2:FREQuency command on page 186.

The setting enabled by this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:FM:INT:FREQ:STEP 1E5
```

The preceding example sets the step value to .1 MHz.

Range 0.5–1E6

:FM[1]|2:EXTernal[1]|2:COUPLing

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:EXTernal[1]|2:COUPling AC|DC
[:SOURce]:FM[1]|2:EXTernal[1]|2:COUPling?
```

This command sets the coupling type for the selected external input. The command does not change the active source or switch modulation on or off. The modulating signal may be the sum of several signals, from either internal or external sources.

AC This choice will pass only ac signal components.

DC This choice will pass both ac and dc signal components.

Example

```
:FM1:EXT1:COUP AC
```

The preceding example sets the coupling for FM path 1, external source 1 to AC.

***RST** DC

Key Entry **Ext Coupling DC AC**

:FM[1]|2:EXTernal[1]|2:IMPedance

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:EXTernal[1]|2:IMPedance <50|600>
[:SOURce]:FM[1]|2:EXTernal[1]|2:IMPedance?
```

This command sets the impedance for the external input.

Example

```
:FM1:EXT2:IMP 600
```

The preceding example sets the FM path 1, external 1 source impedance to 600 ohms.

***RST** +5.00000000E+001

Key Entry **Ext Impedance 50 Ohm 600 Ohm**

:FM[1]|2:INTernal[1]:FREQuency:ALTernate

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:INTernal[1]:FREQuency:ALTernate <val><units>
[:SOURce]:FM[1]|2:INTernal[1]:FREQuency:ALTernate?
```

This command sets the internal FM rate of the alternate signal. The alternate signal frequency is the second tone of a dual–sine or the stop frequency of a swept–sine waveform.

Refer to ":FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 187 for the waveform selection.

Example

```
:FM1:INT:FREQ:ALT 20KHZ
```

The preceding example sets the FM tone 2 rate for FM path 1, FM source 1, to 20 kHz.

***RST** +4.00000000E+002

Range *dual–sine:* 0.5HZ–100kHZ *swept–sine:* 0.5HZ–100kHZ

Key Entry **FM Tone 2 Rate** **FM Stop Rate**

:FM[1]|2:INTernal[1]:FREQuency:ALTernate:AMPLitude:PERCent

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:INTernal[1]:FREQuency:ALTernate:AMPLitude:
PERCent <val><units>
[:SOURce]:FM[1]|2:INTernal[1]:FREQuency:ALTernate:AMPLitude:PERCent?
```

This command sets the amplitude of the second tone for a dual–sine waveform as a percentage of the total amplitude. For example, if the second tone makes up 30% of the total amplitude, then the first tone is 70% of the total amplitude. Refer to ":FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 187 for the waveform selection.

Example

```
:FM1:INT:FREQ:ALT:AMPL:PERC 20
```

The preceding example sets the amplitude for FM tone 2, FM path 1, FM internal source 1 to 20% of the total amplitude.

***RST** +5.00000000E+001

Range 0–100PCT

Key Entry **FM Tone 2 Ampl Percent Of Peak**

:FM[1]|2:INTernal[1]:SWEep:RATE

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:INTernal[1]:SWEep:RATE <val><units>
[:SOURce]:FM[1]|2:INTernal[1]:SWEep:RATE?
```

This command sets the sweep rate for the swept–sine waveform. The minimum resolution is 0.5 hertz. Refer to ":FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 187 for the waveform selection.

Example

```
:FM1:INT:SWE:RATE 20KHZ
```

The preceding example sets the sweep rate for the swept–sine waveform to 20 kilohertz.

***RST** +4.00000000E+002

Range 0.5HZ–100kHZ

Key Entry **FM Sweep Rate**

:FM[1]|2:INTernal[1]:SWEep:TRIGger

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:INTernal[1]:SWEep:TRIGger BUS|IMMediate|EXTernal|KEY
[:SOURce]:FM[1]|2:INTernal[1]:SWEep:TRIGger?
```

This command sets the trigger source for the FM swept–sine waveform. Refer to ":FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 187 for the waveform selection.

BUS	This choice enables GPIB triggering using the `*TRG` or `GET` command or LAN triggering using the `*TRG` command.
IMMediate	This choice enables immediate triggering of the sweep event. This choice is equivalent to pressing the **Free Run** softkey.
EXTernal	This choice enables the triggering of a sweep event by an externally applied signal at the TRIGGER IN connector.
KEY	Enables triggering through front panel interaction (the **Trigger** hardkey).
***RST**	IMM

Example

```
:FM1:INT:SWE:TRIG BUS
```

The preceding example selects the bus as the trigger source for FM path 1.

Key Entry **Bus** **Free Run** **Ext** **Trigger Key**

:FM[1]|2:INTernal[1]|2:FREQuency

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:INTernal[1]|2:FREQuency <val><units>|UP|DOWN
[:SOURce]:FM[1]|2:INTernal[1]|2:FREQuency?
```

This command sets the internal FM rate using the `<val><units>` variable, or changes the FM rate by a user-defined up/down step value. Refer to the :FM:INTernal:FREQuency:STEP[:INCRement] command on page 183 for setting the value associated with the `UP` and `DOWN` choices.

The command changes:

- the FM rate of the first tone of a dual–sine waveform
- the starting FM rate for a swept–sine waveform
- the FM rate for all other waveforms

Refer to ":FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 187 for the waveform selection.

Example

```
:FM2:INT:FREQ 40KHZ
```

The preceding example sets the modulation rate for FM path 2 to 40 kHz.

***RST**	+4.00000000E+002	
Range	*Dual-Sine & Sine*: 0.5HZ–1MHZ	*Swept-Sine*: 1HZ–1MHZ
	All Other Waveforms: 0.5HZ–100kHZ	

Key Entry **FM Tone 1 Rate** **FM Start Rate** **FM Rate**

:FM[1]|2:INTernal[1]|2:FUNCtion:NOISe

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:INTernal[1]|2:FUNCtion:NOISe GAUSsian|UNIForm
[:SOURce]:FM[1]|2:INTernal[1]|2:FUNCtion:NOISe?
```

This command selects a gaussian or uniform noise type as the modulation. Refer to ":FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" on page 187 for the waveform selection.

Example

```
:FM2:INT2:FUNC:NOIS UNIF
```

The preceding example selects a uniform noise waveform as the modulation for FM path 2 and FM source 2.

***RST** UNIF

Key Entry **Gaussian** **Uniform**

:FM[1]|2:INTernal[1]|2:FUNCtion:RAMP

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:INTernal[1]|2:FUNCtion:RAMP POSitive|NEGative
[:SOURce]:FM[1]|2:INTernal[1]|2:FUNCtion:RAMP?
```

This command selects a positive or negative ramp as the internal modulating waveform. Refer to ":FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe" for the waveform selection.

Example

```
:FM2:INT2:FUNC:RAMP POS
```

The preceding example selects a positive sloped ramp as the internal modulating waveform.

***RST** POS

Key Entry **Positive** **Negative**

:FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe SINE|TRIangle|SQUare|
RAMP|NOISe|DUALsine|SWEPtsine
[:SOURce]:FM[1]|2:INTernal[1]|2:FUNCtion:SHAPe?
```

This command selects the FM waveform type. The INTernal2 source selection does not support the dual–sine or Sweep–Sine waveform types.

Example

```
:FM2:INT1:FUNC:SHAP SQU
```

The preceding example selects a square wave as the internal modulating waveform.

***RST** SINE

Key Entry **Sine** **Triangle** **Square** **Ramp** **Noise** **Dual-Sine** **Swept-Sine**

:FM[1]|2:SOURce

Supported All with Option UNT

```
[:SOURce]:FM[1]|2:SOURce INT[1]|INT2|EXT1|EXT2
[:SOURce]:FM[1]|2:SOURce?
```

This command selects the FM source.

INT This choice selects internal source 1 or 2 to provide an ac-coupled signal.

EXT This choice selects the EXT 1 INPUT or the EXT 2 INPUT connector to provide an externally applied signal that can be ac- or dc-coupled. The externally applied, ac-coupled input signal is tested for a voltage level and an annunciator, on the signal generator's front-panel display, will indicate a high or low condition if that voltage is > ±3% of 1 V_p.

Example

```
:FM2:SOUR INT2
```

The preceding example selects internal source 2 as the FM source for FM path 2.

***RST** INT

Key Entry **Internal 1** **Internal 2** **Ext1** **Ext2**

:FM[1]|2:STATe

Supported All with Option UNT

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

This command enables or disables the selected FM path.

The RF carrier is modulated when you set the signal generator's modulation state to ON, see

":MODulation[:STATe]" on page 74 for more information.

Whenever frequency modulation is enabled, the `FM` annunciator appears on the signal generator's front-panel display.

The two paths for frequency modulation can be simultaneously enabled. Refer to ":FM[1]|2..." on page 182 for more information.

Example

```
:FM2:STAT ON
```

The preceding example enables FM path 2.

***RST** 0

Key Entry **FM Off On**

:FM[1]|2[:DEViation]

Supported All with Option UNT

```
[:SOURce]:FM[1]|2[:DEViation] <val><units>
[:SOURce]:FM[1]|2[:DEViation]?
```

This command sets the FM deviation for the selected FM path.

If deviation tracking is ON, a change to the deviation value on one path will apply to both. Refer to ":FM[1]|2[:DEViation]:TRACk" on page 190 for more information on setting the deviation tracking.

Example

```
:FM2 1MHZ
```

The preceding example sets the frequency deviation to 1 megahertz.

***RST** +1.00000000E+003

Range

Frequency	*Deviation*
250KHZ–250MHZ	0–2MHZ
> 250–500MHZ	0–1MHZ
> 0.5–1GHZ	0–2MHZ
> 1–2GHZ	0–4MHZ
> 2–3.2GHZ	0–8MHZ
> 3.2–10GHZ	0–16MHZ
> 10–20GHZ	0–32MHZ

> 20–28.5GHZ[a]	0–48MHZ
> 20–40GHZ	0–64MHZ
> 28.5–44GHZ[a]	0–80MHZ
> 40–67GHZ	0–128MHZ

a. E8267D Only

Key Entry **FM DEV**

:FM[1]|2[:DEViation]:TRACk

Supported All with Option UNT

```
[:SOURce]:FM[1]|2[:DEViation]:TRACk ON|OFF|1|0
[:SOURce]:FM[1]|2[:DEViation]:TRACk?
```

This command enables or disables deviation coupling between FM paths 1 and 2.

ON (1) This choice will link the deviation value of FM1 with FM2; FM2 will assume the FM1 deviation value. For example, if FM1 deviation is set to 500 Hz and FM2 is set to 2 kHz, enabling the deviation tracking will cause the FM2 deviation value to change to 500 Hz. This applies regardless of the path (FM1 or FM2) selected.

OFF (0) This choice disables the coupling and both paths will have independent deviation values.

This command uses exact match tracking, not offset tracking.

Example

```
:FM2:TRAC 0
```

The preceding example disables deviation coupling.

***RST** 0

Key Entry **FM Dev Couple Off On**

Low Frequency Output Subsystem ([:SOURce]:LFOutput)

:LFOutput:AMPLitude

Supported All with Option UNT

```
[:SOURce]:LFOutput:AMPLitude <val><units>
[:SOURce]:LFOutput:AMPLitude?
```

This command sets the amplitude of the signal at the LF OUTPUT connector.

Example

```
:LFO:AMPL 2.1VP
```

The preceding example sets the peak amplitude to 2.1 volts.

***RST** 0.00

Range 0.000VP–3.5VP

Key Entry **LF Out Amplitude**

:LFOutput:FUNCtion[1]|2:FREQuency

Supported All with Option UNT

```
[:SOURce]:LFOutput:FUNCtion[1]|2:FREQuency <val><units>
[:SOURce]:LFOutput:FUNCtion[1]|2:FREQuency?
```

This command sets the frequency of function generator 1 or 2. The command sets:

- the frequency of the first tone of a dual–sine waveform
- the start frequency for a swept–sine waveform
- the frequency for all other waveform types

Refer to ":LFOutput:FUNCtion[1]|2:SHAPe" on page 193 for selecting the waveform type.

Example

```
:LFO:FUNC1:FREQ .1MHZ
```

The preceding example sets the frequency for function generator 1 to 100 kHz.

***RST** +4.00000000E+002

Range *Sine and Dual-Sine*: 0.5HZ–1MHZ

Range	*Swept-Sine*: 1HZ–1MHZ		
	All Other Waveforms: 0.5HZ–100KHZ		
Key Entry	**LF Out Tone 1 Freq**	**LF Out Start Freq**	**LF Out Freq**

:LFOutput:FUNCtion[1]:FREQuency:ALTernate

Supported All with Option UNT

```
[:SOURce]:LFOutput:FUNCtion[1]:FREQuency:ALTernate <val><units>
[:SOURce]:LFOutput:FUNCtion[1]:FREQuency:ALTernate?
```

This command sets the frequency for the alternate LF output signal. The alternate frequency is the second tone of a dual–sine or the stop frequency of a swept–sine waveform.

Refer to ":LFOutput:FUNCtion[1]|2:SHAPe" on page 193 for more infomation on selecting the waveform type.

Example

```
:LFO:FUNC1:FREQ:ALT 20KHZ
```

The preceding example sets the alternate frequency to 20 kHz.

***RST**	+4.00000000E+002	
Range	*Dual-Sine*: 0.1HZ–100kHZ	*Swept-Sine*: 0.1HZ–100kHZ
Key Entry	**LF Out Tone 2 Freq**	**LF Out Stop Freq**

:LFOutput:FUNCtion[1]:FREQuency:ALTernate:AMPLitude:PERCent

Supported All with Option UNT

```
[:SOURce]:LFOutput:FUNCtion[1]:FREQuency:ALTernate:AMPLitude:
PERCent <val><units>
[:SOURce]:LFOutput:FUNCtion[1]:FREQuency:ALTernate:AMPLitude:PERCent?
```

This command sets the amplitude of the second tone for a dual–sine waveform as a percentage of the total LF output amplitude. For example, if the second tone makes up 30% of the total amplitude, then the first tone is 70% of the total amplitude. Refer to ":LFOutput:FUNCtion[1]|2:SHAPe" on page 193 for selecting the waveform type.

Example

```
:LFO:FUNC1:FREQ:ALT:AMPL:PERC 50
```

The preceding example sets the alternate frequency to 50% of the total output amplitude.

***RST** +5.00000000E+001

Range 0–100PCT

Key Entry **LF Out Tone 2 Ampl % of Peak**

:LFOutput:FUNCtion[1]|2:SHAPe

Supported All with Option UNT

```
[:SOURce]:LFOutput:FUNCtion[1]|2:SHAPe SINE|DUALsine|SWEPtsine|TRIangle|
SQUare|RAMP|PULSe|NOISe|DC
[:SOURce]:LFOutput:FUNCtion[1]|2:SHAPe?
```

This command selects the waveform type. Function Generator 1 must be the source for the dual–sine or the swept–sine waveform. Refer to “:LFOutput:SOURce” on page 195.

Example

```
:LFO:FUNC2:SHAP TRI
```

The preceding example selects a triangle wave for the Function Generator 2 LF output.

***RST** SINE

Key Entry **Sine** **Dual-Sine** **Swept-Sine** **Triangle** **Square** **Ramp** **Pulse** **Noise** **DC**

:LFOutput:FUNCtion:[1]|2:SHAPe:NOISe

Supported All with Option UNT

```
[:SOURce]:LFOutput:FUNCtion[1]|2:SHAPe:NOISe UNIForm|GAUSsian
[:SOURce]:LFOutput:FUNCtion[1]|2:SHAPe:NOISe?
```

This command selects a gaussian or uniform noise modulation for the LF output.

Refer to “:LFOutput:FUNCtion[1]|2:SHAPe” on page 193 for selecting the waveform type.

Example

```
:LFO:FUNC1:SHAP:NOIS GAUS
```

The preceding example selects a gaussian noise modulation for the Function Generator 1 LF output.

***RST** UNIF

Key Entry **Uniform** **Gaussian**

:LFOutput:FUNCtion[1]|2:SHAPe:RAMP

Supported All with Option UNT

```
[:SOURce]:LFOutput:FUNCtion[1]|2SHAPe:RAMP POSitive|NEGative
[:SOURce]:LFOutput:FUNCtion[1]|2SHAPe:RAMP?
```

This command selects a positive or negative slope for the ramp modulation on the LF output.

Refer to ":LFOutput:FUNCtion[1]|2:SHAPe" on page 193 for selecting the waveform type.

Example

```
:LFO:FUNC1:SHAP:RAMP POS
```

The preceding example selects a positive ramp slope modulation for the Function Generator 1 LF output.

***RST** POS

Key Entry **Positive** **Negative**

:LFOutput:FUNCtion[1]:SWEep:RATE

Supported All with Option UNT

```
[:SOURce]:LFOutput:FUNCtion[1]:SWEep:RATE <val><units>
[:SOURce]:LFOutput:FUNCtion[1]:SWEep:RATE?
```

This command sets the sweep rate for an internally generated swept–sine signal.

Example

```
:LFO:FUNC1:SWE:RATE 1E5
```

The preceding example sets the sweep rate for the swept–sine waveform to 100 kHz.

***RST** +4.00000000E+002

Range 0.5HZ–100kHZ

Key Entry **LF Out Sweep Rate**

:FUNCtion[1]:SWEep:TRIGger

Supported All with Option UNT

```
[:SOURce]:LFOutput:FUNCtion[1]:SWEep:TRIGger BUS|IMMediate|EXTernal|KEY
[:SOURce]:LFOutput:FUNCtion[1]:SWEep:TRIGger?
```

This command sets the trigger source for the internally generated swept–sine signal at the LF output.

BUS	This choice enables GPIB triggering using the `*TRG` or `GET` command or LAN and RS-232 triggering using the `*TRG` command.
IMMediate	This choice enables immediate triggering of the sweep event.
EXTernal	This choice enables the triggering of a sweep event by an externally applied signal at the TRIGGER IN connector.
KEY	This choice enables triggering through front panel interaction by pressing the **Trigger** hardkey.

Refer to ":LFOutput:FUNCtion[1]|2:SHAPe" on page 193 for selecting the waveform type.

Example

```
:LFO:FUNC1:SWE:TRIG EXT
```

The preceding example sets an external trigger as the trigger for the swept–sine signal.

***RST** Free Run

Key Entry **Bus** **Free Run** **Ext** **Trigger Key**

:LFOutput:SOURce

Supported All with Option UNT

```
[:SOURce]:LFOutput:SOURce INT[1]|INT2|FUNCtion[1]|FUNCtion2
[:SOURce]:LFOutput:SOURce?
```

This command selects the source for the LF output.

INT	This choice enables you to output a signal where the frequency and shape of the signal is set by internal source 1 or 2. 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.
FUNCtion	This choice enables the selection of an internal function generator.

Example

```
:LFO:SOUR FUNC1
```

The preceding example selects Function Generator 1 as the active LF source.

***RST** INT

Key Entry **Internal 1 Monitor** **Internal 2 Monitor** **Function Generator 1** **Function Generator 2**

:LFOutput:STATe

Supported All with Option UNT

```
[:SOURce]:LFOutput:STATe ON|OFF|1|0
[:SOURce]:LFOutput:STATe?
```

This command enables or disables the low frequency output.

Example

```
:LFO:STAT ON
```

The preceding example enables the source.

***RST** 0

Key Entry **LF Out Off On**

Phase Modulation Subsystem ([:SOURce])

:PM[1]|2...

Supported	E8257D and E8267D

`[:SOURce]:PM[1]|2...`

This prefix enables the selection of the ΦM path and associated with all SCPI commands in this subsystem. The two paths are equivalent to the **ΦM Path 1 2** softkey.

PM1	**ΦM Path 1 2** with 1 selected
PM2	**ΦM Path 1 2** with 2 selected

When just PM is shown in a command, this means the command applies to path 1 only.

Each path is set up separately. When a SCPI command uses PM1, only path one is affected. Consequently, when PM2 is selected, only path two is set up. However, the deviation of the signals for the two paths can be coupled.

Deviation coupling links the deviation value of PM1 to PM2. Changing the deviation value for one path will change it for the other path. These two paths can be on at the same time provided the following conditions have been met:

- dual–sine or Sweep–Sine is not the selection for the waveform type
- each path uses a different source (Internal 1, Internal 2, Ext1, or Ext2)
- PM2 must be set to a deviation less than or equal to PM1

:PM:INTernal:FREQuency:STEP[:INCRement]

Supported All with Option UNT

```
[:SOURce]:PM:INTernal:FREQuency:STEP[:INCRement]<num>|MAXimum|MINimum|DE
Fault
[:SOURce]:PM:INTernal:FREQuency:STEP[:INCRement]?
```

This command sets the step value of the phase modulation internal frequency.

The step value set by this command is used with the UP and DOWN choices for the :PM[1]|2:INTernal[1]:FREQuency command on page 199.

The setting enabled by this command is not affected by a signal generator power-on, preset, or *RST command.

Example

```
:PM:INT:FREQ:STEP 1E5
```

The preceding example sets the step value to 100 kHz.

Range 0.5–1E6

Key Entry **Incr Set**

:PM[1]|2:BANDwidth|BWIDth

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:BANDwidth|BWIDth NORMal|HIGH
[:SOURce]:PM[1]|2:BANDwidth|BWIDth?
```

This command selects normal phase modulation or high bandwidth phase modulation. The command can use either the BANDwidth or BWIDth paths.

Example

```
:PM1:BAND NORM
```

The preceding example selects normal phase modulation for ΦM path 1.

***RST** NORM

Key Entry **FM ΦM Normal High BW**

:PM[1]|2:EXTernal[1]:COUPling

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:EXTernal[1]:COUPling AC|DC
```

```
[:SOURce]:PM[1]|2:EXTernal[1]:COUPling?
```

This command sets the coupling for the phase modulation source at the selected external input connector.

AC This choice will only pass ac signal components.

DC This choice will pass both ac and dc signal components.

This command does not change the active source or switch modulation on or off. The modulating signal may be the sum of several signals, from either internal or external sources.

Example

```
:PM1:EXT:COUP AC
```

The preceding example selects AC coupling at the external input for ΦM path 1.

***RST** DC

Key Entry **Ext Coupling DC AC**

:PM[1]|2:EXTernal[1]|2:IMPedance

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:EXTernal[1]|2:IMPedance <50|600>
[:SOURce]:PM[1]|2:EXTernal[1]|2:IMPedance?
```

This command selects 50 ohms or 600 ohms as the input impedance for the external input signal.

Example

```
:PM1:EXT2:IMP 600
```

The preceding example sets the ΦM path 1, external 2 source impedance to 600 ohms.

***RST** +5.00000000E+001

Key Entry **Ext Impedance 50 Ohm 600 Ohm**

:PM[1]|2:INTernal[1]:FREQuency

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:INTernal[1]|2:FREQuency <val><units>
[:SOURce]:PM[1]|2:INTernal[1]|2:FREQuency?
```

This command sets the internal modulation frequency rate. The command sets:

- the frequency of the first tone of a dual–sine waveform

- the start frequency for a swept–sine waveform
- the frequency rate for all other waveforms

Refer to ":LFOutput:FUNCtion[1]|2:SHAPe" on page 193 for selecting the waveform type.

Example

```
:PM1:INT1:FREQ 20KHZ
```

The preceding example sets the ΦM path 1, internal source 1 frequency to 20 kHz.

***RST**	+4.00000000E+002
Range	*Dual-Sine*: 0.1HZ–100KHZ *Swept-Sine*: 0.1HZ–100KHZ
	All Other Waveforms: 0.1HZ–20KHZ
Key Entry	**ΦMTone 1 Rate** **ΦM Start Rate** **ΦM Rate**

PM[1]|2:INTernal[1]:FREQuency:ALTernate

Supported   All with Option UNT

```
[:SOURce]:PM[1]|2:INTernal[1]:FREQuency:ALTernate <val><units>
[:SOURce]:PM[1]|2:INTernal[1]:FREQuency:ALTernate?
```

This command sets the frequency rate for the alternate signal. The alternate frequency is the second tone of a dual–sine or the stop frequency of a swept–sine waveform.

Refer to ":PM[1]|2:INTernal[1]:FUNCtion:SHAPe" on page 202 for the waveform selection.

Example

```
:PM1:INT1:FREQ:ALT 50KHZ
```

The preceding example sets the alternate frequency rate for the ΦM tone 2, ΦM path 1, source 1 to 50 kHz.

***RST**	+4.00000000E+002
Range	*Dual-Sine*: 0.1HZ–100KHZ *Swept-Sine*: 0.1HZ–100KHZ
Key Entry	**ΦM Stop Rate** **ΦM Tone 2 Rate**

:PM[1]|2:INTernal[1]|2:FUNCtion:NOISe

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:INTernal[1]|2:FUNCtion:NOISe GAUSsian|UNIForm
[:SOURce]:PM[1]|2:INTernal[1]|2:FUNCtion:NOISe?
```

This commands selects a gaussian or uniform noise modulation for the selected path(s).

Example

```
:PM1:INT1:FUNC:NOIS GAUS
```

The preceding example selects a gaussian noise modulation for ΦM path 1, source 1.

***RST** UNIF

Key Entry **Gaussian** **Uniform**

:PM[1]|2:INTernal[1]|2:FUNCtion:RAMP

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:INTernal[1]|2:FUNCtion:RAMP POSitive|NEGative
[:SOURce]:PM[1]|2:INTernal[1]|2:FUNCtion:RAMP?
```

This command selects a positive or negative slope for the ramp modulating waveform.

Example

```
:PM1:INT2:FUNC:RAMP POS
```

The preceding example selects a positive ramp slope for modulating the signal on ΦM path 1, internal source 2.

***RST** POS

Key Entry **Positive** **Negative**

:PM[1]|2:INTernal[1]:FREQuency:ALTernate:AMPLitude:PERCent

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:INTernal[1]:FREQuency:ALTernate:AMPLitude:
PERCent <val>
[:SOURce]:PM[1]|2:INTernal[1]:FREQuency:ALTernate:AMPLitude:PERCent?
```

This command sets the amplitude of the second tone for the dual–sine waveform as a percentage of the total amplitude. For example, if the second tone makes up 30% of the total amplitude, then the first tone is 70% of the total amplitude. Refer to “:PM[1]|2:INTernal[1]:FUNCtion:SHAPe” on page 202

for the waveform selection.

Example

```
:PM2:INT:FREQ:ALT:AMPL:PERC 40
```

The preceding example sets the alternate tone amplitude to 40% of the total amplitude.

***RST** +5.00000000E+001

Range 0–100PCT

Key Entry **ΦM Tone 2 Ampl Percent of Peak**

:PM[1]|2:INTernal[1]:FUNCtion:SHAPe

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:INTernal[1]:FUNCtion:SHAPe SINE|TRIangle|SQUare|RAMP|
NOISe|DUALsine|SWEPtsine
[:SOURce]:PM[1]|2:INTernal[1]:FUNCtion:SHAPe?
```

This command sets the phase modulation waveform type.

Example

```
:PM1:INT:FUNC:SHAP RAMP
```

The preceding example selects a ramp modulation for ΦM path 1, source 1.

***RST** SINE

Key Entry **Sine** **Triangle** **Square** **Ramp** **Noise** **Dual-Sine** **Swept-Sine**

:PM[1]|2:INTernal[1]:SWEep:RATE

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:INTernal[1]:SWEep:RATE <val><units>
[:SOURce]:PM[1]|2:INTernal[1]:SWEep:RATE?
```

This command sets the sweep rate for a phase-modulated, swept–sine waveform. Refer to “:PM[1]|2:INTernal[1]:FUNCtion:SHAPe” on page 202 for the waveform selection.

Example

```
:PM1:INT:SWE:RATE 30KHZ
```

The preceding example sets the sweep rate to 30 kHz.

***RST** +4.00000000E+002

Range 0.5HZ–100kHZ

Key Entry **ΦM Sweep Rate**

:PM[1]|2:INTernal[1]:SWEep:TRIGger

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:INTernal[1]:SWEep:TRIGger BUS|IMMediate|EXTernal|KEY
[:SOURce]:PM[1]|2:INTernal[1]:SWEep:TRIGger?
```

This command sets the trigger source for the phase-modulated, swept–sine waveform.

BUS This choice enables GPIB triggering using the `*TRG` or `GET` command or LAN and RS-232 triggering using the `*TRG` command.

IMMediate This choice enables immediate triggering of the sweep event. This choice is equivalent to pressing the **Free Run** softkey.

EXTernal This choice enables the triggering of a sweep event by an externally applied signal at the TRIGGER IN connector.

KEY This choice enables triggering through front panel interaction by pressing the **Trigger** hardkey.

Refer to ":PM[1]|2:INTernal[1]:FUNCtion:SHAPe" on page 202 for the waveform selection.

Example

```
:PM2:INT:SWE:TRIG BUS
```

The preceding example selects a BUS trigger as the triggering for the internal source 1 swept–sine waveform on ΦM path 2.

***RST** IMM

Key Entry **Bus** **Free Run** **Ext** **Trigger Key**

:PM[1]|2:SOURce

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:SOURce INT[1]|INT2|EXT[1]|EXT2
[:SOURce]:PM[1]|2:SOURce?
```

This command selects the source used to generate the phase modulation.

INT This choice selects internal source 1 or internal source 2 to provide an ac-coupled signal.

EXT This choice selects the EXT 1 INPUT or the EXT 2 INPUT connector to provide an externally applied signal that can be ac- or dc- coupled.

The externally applied, ac-coupled input signal is tested for a voltage level and an annunciator, on the signal generator’s front–panel display, will indicate a high or low condition if that voltage is > ±3% of 1 V_p.

Example

```
:PM2:SOUR EXT1
```

The preceding example selects an external signal on the EXT 1 INPUT connector as the source for ΦM path 2 modulation.

***RST** INT

Key Entry **Internal 1** **Internal 2** **Ext1** **Ext2**

:PM[1]|2:STATe

Supported All with Option UNT

```
[:SOURce]:PM[1]|2:STATe ON|OFF|1|0
[:SOURce]:PM[1]|2:STATe?
```

This command enables or disables the phase modulation for the selected path. The RF carrier is modulated when you set the signal generator’s modulation state to ON, see “:MODulation[:STATe]” on page 74 for more information.

The ΦM annunciator appears on the signal generator’s front-panel display whenever phase modulation is enabled. The two paths for phase modulation can be simultaneously enabled. Refer to “:PM[1]|2...” on page 197 for more information.

Example

```
:PM2:STAT 1
```

The preceding example turns on ΦM path 2 phase modulation.

***RST** 0

Key Entry **ΦM Off On**

:PM[1]|2[:DEViation]

Supported All with Option UNT

```
[:SOURce]:PM[1]|2[:DEViation] <val><units>|UP|DOWN
[:SOURce]:PM[1]|2[:DEViation]?
```

This command sets the deviation of the phase modulation. The variable `<units>` will accept RAD (radians), PIRAD (pi-radians), and DEG (degrees); however, the query will only return values in radians. If deviation tracking is active, a change to the deviation value on one path will apply to both.

The command, used with the `UP|DOWN` parameters, will change the deviation by a user-defined step value. Refer to the :PM[:DEViation]:STEP[:INCRement] command on page 206 for setting the value associated with the `UP` and `DOWN` choices.

Example

```
:PM1 135DEG
```

The preceding example sets the phase modulation to 135 degrees.

***RST** +0.00000000E+000

Range

Frequency	*Normal Bandwidth*	*High Bandwidth*
250KHZ–250MHZ	0–20rad	0–2rad
> 250–500MHZ	0–10rad	0–1rad
> 0.5–1GHZ	0–20rad	0–2rad
> 1–2GHZ	0–40rad	0–4rad
> 2–3.2GHZ	0–80rad	0–8rad
> 3.2–10GHZ	0–160rad	0–16rad
> 10.0–20GHZ	0–320rad	0–32rad
> 20.0–28.5GHZ[a]	0–480rad	0–48rad
> 20.0–40.0GHZ	0–640rad	0–64rad
> 28.5–44.0GHZ[a]	0–800rad	0–80rad
>40–67.0GHZ[b]	0–1280rad	0–128rad

Key Entry **ΦM Dev**

a. E8267D Only
b. Performance is not specified above 50 GHz

:PM[1]|2[:DEViation]:TRACk

Supported All with Option UNT

```
[:SOURce]:PM[1]|2[:DEViation]:TRACk ON|OFF|1|0
[:SOURce]:PM[1]|2[:DEViation]:TRACk?
```

This command enables or disables the deviation coupling between the PM paths 1 and 2.

ON (1) This choice will link the deviation value of PM1 with PM2; PM2 will assume the PM[1] deviation value. For example, if PM1 deviation is set to 500 Hz and PM2 is set to 2 kHZ, enabling the deviation tracking will cause the PM2 deviation value to change to 500 Hz. This applies regardless of the path (PM1 or PM2) selected in this command.

OFF (0) This choice disables the coupling and both paths will have independent deviation values.

This command uses exact match tracking, not offset tracking.

Example

```
:PM1:TRAC OFF
```

The preceding example disables deviation coupling.

***RST** 0

Key Entry **ΦM Dev Couple Off On**

:PM[:DEViation]:STEP[:INCRement]

Supported All with Option UNT

```
[:SOURce]:PM[:DEViation]:STEP[:INCRement]<val><units>|MAXimum|MINimum|DEFault
[:SOURce]:PM[:DEViation]:STEP[:INCRement]?
```

This command sets the phase modulation deviation step value.

The value set by this command is used with the UP and DOWN choices for the FM deviation command. Refer to “:PM[1]|2[:DEViation]” on page 204 for more information.

The setting is not affected by a signal generator power-on, preset, or `*RST` command..

Example

```
:PM:STEP 20RAD
```

The preceding example sets the step value to 20 radians.

Range 0.001–1E3RAD

Pulse Modulation Subsystem ([:SOURce])

:PULM:EXTernal:POLarity NORMal:INVerted

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:EXTernal:POLarity NORMal|INVerted
[:SOURce]:PULM:EXTernal:POLarity?
```

This command selects the polarity of the TTL input signal at the GATE/PULSE/TRIGGER INPUT front panel connector. The signal generator can respond to either a normal (a TTL high) or an inverted (TTL low) signal.

Example

```
:PULM:EXT:POL NORM
```

The preceding example selects normal (TTL high) polarity.

***RST** Normal

Key Entry **Ext Polarity Normal Inverted**

:PULM:INTernal[1]:DELay

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:INTernal[1]:DELay <num><time_suffix>|UP|DOWN
[:SOURce]:PULM:INTernal[1]:DELay?
```

This command sets the pulse delay for the internally-generated pulse modulation using the variable `<num>[<time_suffix>]`. The command, used with the `UP|DOWN` parameters, will change the delay by a user-defined step value. Refer to the :PULM:INTernal[1]:DELay:STEP command on page 208 for setting the value associated with the UP and DOWN choices.

The optional variable `<time_suffix>` accepts nS (nanoseconds) to S (seconds).

The range value is dependent on the pulse period. Refer to ":PULM:INTernal[1]:PERiod" on page 209 for pulse period settings.

Example

```
:PULM:INT:DEL 200E-9
```

The preceding example sets the internal pulse delay to 200 nanoseconds.

***RST**	+0.00000000E+000
Range	*Internal Free Run*: depends on pulse period and pulse width settings
	Internal Triggered & Doublet: 70nS to (42 S - 20 nS - pulse width)
Key Entry	**Pulse Delay**

:PULM:INTernal[1]:DELay:STEP

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:INTernal[1]:DELay:STEP <num><time_suffix>
[:SOURce]:PULM:INTernal[1]:DELay:STEP?
```

This command sets the step increment for the pulse delay.

The step value, set by this command, is used with the `UP` and `DOWN` choices in the ":PULM:INTernal[1]:DELay" on page 207 command.

The step value set with this command is not affected by a signal generator power-on, preset, or `*RST` command.

Example

```
:PULM:INT:DEL:STEP 10NS
```

The preceding example sets the pulse delay step value to 10 nanoseconds.

Range 10nS to (pulse period – 20 nS)

:PULM:INTernal[1]:FREQuency

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:INTernal[1]:FREQuency <val><units>|UP|DOWN
[:SOURce]:PULM:INTernal[1]:FREQuency?
```

This command sets the pulse rate for the internally-generated square wave using the variable `<val><units>`. The command, used with the `UP|DOWN` parameters, will change the frequency by a user-defined step value. Refer to the :PULM:INTernal[1]:FREQuency:STEP command for setting the value associated with the UP and DOWN choices.

This command is used when SQUare is the pulse modulation type. Refer to ":PULM:SOURce" on page 212 for the pulse modulation type selection.

Example

```
:PULM:INT:FREQ 1MHZ
```

The preceding example sets the square wave pulse rate to 1 megahertz.

***RST** +4.00000000E+002

Range 0.1HZ–10MHZ

Key Entry **Pulse Rate**

:PULM:INTernal[1]:FREQuency:STEP

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:INTernal[1]:FREQuency:STEP[:INCRement] <frequency>
[:SOURce]:PULM:INTernal[1]:FREQuency:STEP[INCRement]?
```

This command sets the step value for the internally-generated square wave pulse rate.

This command is used when SQUare is the pulse modulation type. Refer to ":PULM:SOURce" on page 212 for the pulse modulation type selection. The step value, set with this command, is used with the `UP` and `DOWN` choices in the :PULM:INTernal[1]:FREQuency command.

The step value set with this command is not affected by a power-on, preset, or `*RST` command.

Example

```
:PULM:INT:FREQ:STEP MIN
```

The preceding example sets the step value for the square wave pulse rate to 0.1 Hz, the minimum rate.

Range 0.1HZ–10MHZ

:PULM:INTernal[1]:PERiod

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:INTernal[1]:PERiod <val><units>|UP|DOWN
[:SOURce]:PULM:INTernal[1]:PERiod?
```

This command sets the pulse period for the internally–generated pulse modulation using the variables `<val><units>`. The command, used with the `UP|DOWN` parameters, will change the pulse period by a user–defined step value. Refer to the :PULM:INTernal[1]:PERiod:STEP[:INCRement] command for setting the value associated with the UP and DOWN choices.

If the entered value for the pulse period is equal to or less than the value for the pulse width, the pulse width changes to a value that is less than the pulse period. Refer to ":PULM:INTernal[1]:PWIDth" on page 210 for setting the pulse width.

Example

```
:PULM:INT:PER .5S
```

The preceding example sets the period of the internally–generated pulse to 500 milliseconds.

***RST** +2.00000000E–006

Range 70nS–42S

Key Entry **Pulse Period**

:PULM:INTernal[1]:PERiod:STEP[:INCRement]

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:INTernal[1]:PERiod:STEP[:INCRement]<val><units>|MAXimum|
MINimum|DEFault
[:SOURce]:PULM:INTernal[1]:PERiod:STEP[:INCRement]?
```

This command sets the step value for the internal pulse period using the variable `<val><units>`.

The step value, set with this command, is used with the `UP` and `DOWN` choices available in the :PULM:INTernal[1]:PERiod command.

The step value set with this command is not affected by a power-on, preset, or `*RST` command.

Example

```
:PULM:INT:PER:STEP .1S
```

The preceding example sets the square wave pulse rate to 100 milliseconds.

***RST** +1.00000000E-006

Range 10nS–42S

:PULM:INTernal[1]:PWIDth

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:INTernal[1]:PWIDth <num><time_suffix>|UP|DOWN
[:SOURce]:PULM:INTernal[1]:PWIDth?
```

This command sets the pulse width for the internally generated pulse signal.

This command sets the pulse width for the internally-generated pulse modulation using the variables `<num><time_suffix>`. The command, used with the `UP|DOWN` parameters, will change the pulse width by a user-defined step value. Refer to the :PULM:INTernal[1]:PWIDth:STEP command for setting the value associated with the UP and DOWN choices.

If the entered value for the pulse width is equal to or greater than the value for the pulse period, the pulse width changes to a value that is less than the pulse period. For more information, refer to the command ":PULM:INTernal[1]:PWIDth" on page 210.

NOTE A power search is recommended for signals with pulse widths less than one microsecond. Refer to “:ALC:SEARch” on page 155.

Example

```
:PULM:INT:PWIDth 100MS
```

The preceding example sets the pulse width to 100 milliseconds.

***RST** +1.00000000E–006

Range 10nS to (pulse period - 20 nS)

Key Entry **Pulse Width**

:PULM:INTernal[1]:PWIDth:STEP

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:INTernal[1]:PWIDth:STEP<num><time_suffix>|MAXimum|MINimum
|DEFault
[:SOURce]:PULM:INTernal[1]:PWIDth:STEP?
```

This command sets the step increment for the pulse width using the variable `<num><time_suffix>`.

The step value, set by this command, is used with the UP and DOWN choices available in the “:PULM:INTernal[1]:PWIDth” command.

The step value, set with this command, is not affected by a power-on, preset, or `*RST` command.

Example

```
:PULM:INT:PWID:STEP 100NS
```

The preceding example sets the pulse width step to 100 nanoseconds.

***RST** +1.00000000E–006

Range 10nS to (pulse period - 20 nS)

:PULM:INTernal

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:SOURce:INTernal SQUare|FRUN|TRIGgered|DOUBlet|GATEd
[:SOURce]:PULM:SOURce:INTernal?
```

This command selects one of the five internally generated modulation inputs. There are two external sources: Scalar and Ext Pulse which are selected using the `:PULM:SOURce` command.

Example

`:PULM:SOUR:INT SQU`

The preceding example selects the internally–generated square wave pulse modulation format.

***RST** FRUN (Int Free–Run)

Key Entry **Internal Square** **Int Free-Run** **Int Triggered** **Int Doublet** **Int Gated**

:PULM:SOURce

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:SOURce INTernal|EXTernal|SCALar
[:SOURce]:PULM:SOURce?
```

This command sets the source for pulse modulation. The `INTernal` selection accesses one of the five internally generated modulation inputs while `EXTernal` selects an external pulse (Ext Pulse) and `SCALar` selects input from a scalar network analyzer.

Example

`:PULM:SOUR INT`

The preceding example selects the internal free-run, pulse modulation source.

***RST** FRUN (Int Free-Run)

Key Entry **Internal Square** **Int Free-Run** **Int Triggered** **Int Doublet** **Int Gated**
Ext Pulse **Scalar**

:PULM:STATe

Supported All with Option UNU or UNW

```
[:SOURce]:PULM:STATe ON|OFF|1|0
[:SOURce]:PULM:STATe?
```

This command enables or disables pulse modulation for the selected path.

When pulse modulation is enabled, the PULSE annunciator appears on the signal generator's front-panel display.

Example

```
:PULM:STAT ON
```

The preceding example enables the pulse modulation.

***RST**	0
Key Entry	**Pulse Off On**

5 Digital Modulation Commands

In the following sections, this chapter provides SCPI descriptions for subsystems dedicated to the E8267D PSG Vector signal generator:

All Subsystem–Option 601 and 602 ([:SOURce])

:RADio:ALL:OFF

Supported E8267D with Option 601or 602

`[:SOURce]:RADio:ALL:OFF`

This command disables all digital modulation personalities on a particular baseband. This command does not affect analog modulation.

AWGN ARB Subsystem–Option 403 ([:SOURce]:RADio:AWGN:ARB)

:BWIDth

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:BWIDth <val>
[:SOURce]:RADio:AWGN:ARB:BWIDth?
```

This command adjusts the bandwidth of the AWGN waveform.

The variable <val> is expressed in units of hertz (Hz–MHz).

***RST** +1.00000000E+006

Range 5E4–1.5E7

Key Entry **Bandwidth**

:IQ:EXTernal:FILTer

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:IQ:EXTernal:FILTer 40e6|THRough
[:SOURce]:RADio:AWGN:ARB:IQ:EXTernal:FILTer?
```

This command selects the filter or through path for I/Q signals routed to the rear panel I and Q outputs. Selecting a filter setting with this command will automatically set the ‘:IQ:EXTernal:FILTer:AUTO’ on page 217 command to Off mode.

40e6 This choice applies a 40 MHz baseband filter.

THRough This choice bypasses filtering.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:EXTernal:FILTer:AUTO

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:IQ:EXTernal:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:AWGN:ARB:IQ:EXTernal:FILTer:AUTO?
```

This command enables or disables the automatic selection of the filters for I/Q signals routed to the rear panel I/Q outputs.

ON(1)	This choice will automatically select a digital modulation filter optimized for the current signal generator settings.
OFF(0)	This choice disables the auto feature which lets you select a digital modulation filter or through path. Refer to “:IQ:EXTernal:FILTer” on page 217 for selecting a filter or through path.

***RST** ON

Key Entry **I/Q Output Filter Manual Auto**

:HEADer:CLEar

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:HEADer:CLEar
```

This command clears the header information from the header file used by this modulation format. The **AWGN Off On** softkey must be set to On for this command to function.

***RST** N/A

Key Entry **Clear Header**

:HEADer:SAVE

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:HEADer:SAVE
```

This command saves the header information to the header file used by this modulation format. The **AWGN Off On** softkey must be set to On for this command to function.

***RST** N/A

Key Entry **Save Setup To Header**

:IQ:MODulation:ATTen

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:IQ:MODulation:ATTen <val>
[:SOURce]:RADio:AWGN:ARB:IQ:MODulation:ATTen?
```

This command attenuates the I/Q signals being modulated through the signal generator’s RF path.

The variable <val> is expressed in units of decibels (dB).

***RST** +2.00000000E+000

Range 0–40

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:ATTen:AUTO

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:IQ:MODulation:ATTen:AUTO ON|OFF|1|0
[:SOURce]:RADio:AWGN:ARB:IQ:MODulation:ATTen:AUTO?
```

This command enables or disables the I/Q attenuation auto mode.

ON (1) This choice enables the attenuation auto mode which optimizes the modulator attenuation for the current conditions.

OFF (0) This choice holds the attenuator at its current setting or at a selected value. Refer to ‘:IQ:MODulation:ATTen’ on page 218 for setting the attenuation value.

***RST** 1

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:FILTer

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:IQ:MODulation:FILTer 40e6|THRough
[:SOURce]:RADio:AWGN:ARB:IQ:MODulation:FILTer?
```

This command enables you to select a filter or through path for I/Q signals modulated onto the RF carrier. Selecting a filter with this command will automatically set ‘:IQ:MODulation:ATTen:AUTO’ on page 219 to Off(0) mode.

40E6 This choice applies a 40 MHz baseband filter to the I/Q signals.

THRough This choice bypasses filtering.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:MODulation:FILTer:AUTO

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:IQ:MODulation:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:AWGN:ARB:IQ:MODulation:FILTer:AUTO?
```

This command enables or disables the automatic selection of the filters for I/Q signals modulated onto the RF carrier.

ON(1)	This choice will automatically select a digital modulation filter.
OFF(0)	This choice disables the auto feature which lets you select a digital modulation filter or through path. Refer to “:IQ:MODulation:FILTer” on page 277 for selecting a filter or through path.
***RST**	1
Key Entry	**I/Q Mod Filter Manual Auto**

:MDEStination:PULSe

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:MDEStination:PULSe NONE|M1|M2|M3|M4
[:SOURce]:RADio:AWGN:ARB:MDEStination:PULSe?
```

This command routes the selected marker to the Pulse/RF Blanking function. The `NONE` parameter clears the marker for the Pulse/RF Blanking functions.

***RST** NONE

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

:MDEStination:AAMPlitude

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:MDEStination:AAMPlitude NONE|M1|M2|M3|M4
[:SOURce]:RADio:AWGN:ARB:MDEStination:AAMPlitude?
```

This command routes the selected marker to the Alternate Amplitude function. The `NONE` parameter clears the marker for the Alternate Amplitude function.

***RST** NONE

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

:MDEStination:ALCHold

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:MDEStination:ALCHold NONE|M1|M2|M3|M4
[:SOURce]:RADio:AWGN:ARB:MDEStination:ALCHold?
```

This command routes the selected marker to the ALC Hold function. The `NONE` parameter clears the marker for the ALC Hold function.

***RST** NONE

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

:MPOLarity:MARKer1

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:MPOLarity:MARKer1 NEGative|POSitive
[:SOURce]:RADio:AWGN:ARB:MPOLarity:MARKer1?
```

This command sets the polarity for marker 1.

***RST** POS

Key Entry **Marker 1 Polarity Neg Pos**

:MPOLarity:MARKer2

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:MPOLarity:MARKer2 NEGative|POSitive
[:SOURce]:RADio:AWGN:ARB:MPOLarity:MARKer2?
```

This command sets the polarity for marker 2.

***RST** POS

Key Entry **Marker 2 Polarity Neg Pos**

:MPOLarity:MARKer3

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:MPOLarity:MARKer3 NEGative|POSitive
[:SOURce]:RADio:AWGN:ARB:MPOLarity:MARKer3?
```

This command sets the polarity for marker 3.

***RST** POS

Key Entry **Marker 3 Polarity Neg Pos**

:MPOLarity:MARKer4

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:MPOLarity:MARKer4 NEGative|POSitive
[:SOURce]:RADio:AWGN:ARB:MPOLarity:MARKer4?
```

This command sets the polarity for marker 4.

***RST** POS

Key Entry **Marker 4 Polarity Neg Pos**

:LENgth

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:LENgth 1048576|524288|262144|131072|65536|
32768|16384
[:SOURce]:RADio:AWGN:ARB:LENgth?
```

This command specifies the length (number of points) of the AWGN waveform. A longer waveform yields a statistically more correct waveform.

***RST** 524288

Key Entry **1048576** **524288** **262144** **131072** **65536** **32768** **16384**

:REFerence:EXTernal:FREQuency

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:REFerence:EXTernal:FREQuency <val>
[:SOURce]:RADio:AWGN:ARB:REFerence:EXTernal:FREQuency?
```

This command allows you to enter the frequency of the applied external reference. The value specified by this command is effective only when you are using an external ARB reference applied to the BASEBAND GEN REF IN rear panel connector. To specify external as the ARB reference source type, refer to ":REFerence[:SOURce]" on page 315.

The variable <val> is expressed in units of hertz (Hz–MHz).

***RST** +1.00000000E+007

Range 2.5E5–1E8

Key Entry **Reference Freq**

:REFerence[:SOURce]

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:REFerence[:SOURce] INTernal|EXTernal
[:SOURce]:RADio:AWGN:ARB:REFerence[:SOURce]?
```

This command selects either an internal or external reference for the waveform clock. If the EXTernal choice is selected, the external frequency *value must* be entered and the signal must be applied to the

BASEBAND GEN REF IN rear panel connector. Refer to “:REFerence:EXTernal:FREQuency” on page 315 to enter the external reference frequency.

***RST** INT

Key Entry **ARB Reference Ext Int**

:SCLock:RATE

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:SCLock:RATE <val>
[:SOURce]:RADio:AWGN:ARB:SCLock:RATE?
```

This command sets the sample clock rate for the AWGN modulation format. The modulation format should be active before executing this command. If this command is executed before the modulation format is active, the entered value will be overridden by a calculated factory default value. Refer to ‘:BURSt:SHAPe:FALL:DELay’ on page 228 to activate the modulation format.

The variable <val> is expressed in units of hertz.

***RST** +1.00000000E+008

Range 1–1E8

Key Entry **ARB Sample Clock**

:SEED

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB:SEED FIXed|RANDom
[:SOURce]:RADio:AWGN:ARB:SEED?
```

This command toggles the AWGN waveform noise seed value type.

FIXed This choice selects a fixed noise seed value.

RANDom This choice selects a randomly generated noise seed value.

***RST** FIX

Key Entry **Noise Seed Fixed Random**

[:STATe]

Supported All with Option 403

```
[:SOURce]:RADio:AWGN:ARB[:STATe] ON|OFF|1|0
[:SOURce]:RADio:AWGN:ARB[:STATe]?
```

This command enables or disables the AWGN generator function.

***RST** 0

Key Entry **Arb AWGN Off On**

AWGN Real-Time Subsystem–Option 403 ([:SOURce]:RADio:AWGN:RT)

:BWIDth

Supported E8267D with Option 403

```
[:SOURce]:RADio:AWGN:RT:BWIDth <val>
[:SOURce]:RADio:AWGN:RT:BWIDth?
```

This command adjusts the real-time AWGN bandwidth value.

The variable <val> is expressed in units of hertz (Hz–MHz).

***RST** +1.00000000E+006

Range 5E4–8E7

Key Entry **Bandwidth**

[:STATe]

Supported E8267D with Option 403

```
[:SOURce]:RADio:AWGN:RT[:STATe] ON|OFF|1|0
[:SOURce]:RADio:AWGN:RT[:STATe]?
```

This command enables or disables the operating state of real-time AWGN.

***RST** 0

Key Entry **Real-time AWGN Off On**

Custom Subsystem–Option 601 and 602 ([:SOURce]:RADio:CUSTom)

:ALPha

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:ALPHa <val>
[:SOURce]:RADio:CUSTom:ALPHa?
```

This command changes the Nyquist or root Nyquist filter’s alpha value. The filter alpha value can be set to a minimum level (0), a maximum level (1), or in between by using fractional numeric values (0.001–0.999). To change the current filter type, refer to “:FILTer” on page 238.

Example

```
:RAD:CUST:ALPH .65
```

The preceding example sets the filter alpha to .65.

***RST** +3.50000000E–001

Range 0.000–1.000

Key Entry Filter Alpha

:BBCLock

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BBCLock INT[1]|EXT[1]
[:SOURce]:RADio:CUSTom:BBCLock?
```

This command toggles the data (bit) clock input to the baseband generator board to either internal or external. This command is independent in each mode and works for both non-burst (continuous) and burst modes. This allows for a matrix of selections between burst/non-burst, internal/external data generation, internal/external data clock, and external bit/symbol data clock.

INT [1] This choice selects the signal generator internal data clock.

EXT [1] This choice selects an external data clock input.

A data clock or continuous symbol sync input must be supplied when external mode is used. This is ignored if the external reference is set to EXTernal (see “:EREFerence” on page 237).

Example

```
:RAD:CUST:BBCL 1
```

The preceding example selects the signal generator's internal data clock.

***RST** INT

Key Entry **BBG Data Clock Ext Int**

:BBT

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BBT <val>
[:SOURce]:RADio:CUSTom:BBT?
```

This command changes the bandwidth-multiplied-by-bit-time (BbT) filter parameter. The filter BbT value can be set to the maximum level (1) or in between the minimum level (0.100) and maximum level by using fractional numeric values (0.101–0.999). This command is effective only after choosing a Gaussian filter. It does not effect other types of filters (see ":FILTer" on page 238).

Example

```
:RAD:CUST:BBT .300
```

The preceding example selects a 0.300 BbT gaussian filter.

***RST** +5.00000000E–001

Range 0.100–1.000

Key Entry Filter BbT

:BRATe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BRATe <val>
[:SOURce]:RADio:CUSTom:BRATe?
```

This command sets the bit rate. The variable <val> is expressed in bits per second (bps–Mbps) and the maximum range value depends on the data source (internal or external), the modulation type, and filter. When user-defined filters are selected (see ":FILTer" on page 238), the upper bit rate is restricted using the following criteria:

- FIR filter length > 32 symbols: upper limit is 12.5 Msps
- FIR filter length > 16 symbols: upper limit is 25 Msps

When internal FIR filters are used, these limit restrictions always apply. For higher symbol rates, the FIR filter length will be truncated and will impact the relative timing of the modulated data, as well as the actual filter response (see ":SRATe" on page 242).

A change in the bit rate value effects the symbol rate value; refer to ":SRATe" on page 242 for a list of the minimum and maximum symbol rate values.

To change the modulation type, refer to “:MODulation[:TYPE]” on page 241.

Example

```
:RAD:CUST:BRAT 10MBPS
```

The preceding example sets the bit rate to 10 megabits per second.

***RST** +4.86000000E+004

Range

Modulation Type	*Bits per Symbol*	*Internal Data*	*External Serial Data*
BPSK	1	45 bps–50 Mbps	45 bps–50 Mbps
FSK2			
MSK			
C4FM	2	90 bps–100 Mbps	45 bps–50 Mbps
FSK4			
OQPSK			
OQPSK195			
P4QPPSK			
QAM4			
QPSK			
QPSKIS95			
QPSKISAT			
D8PSK	3	135 bps–150 Mbps	45 bps–50 Mbps
EDGE			
FSK8			
PSK8			
FSK16	4	180 bps–200 Mbps	45 bps–50 Mbps
PSK16			
QAM16			
QAM32	5	225 bps–250 Mbps	45 bps–50 Mbps
QAM64	6	270 bps–300 Mbps	45 bps–50 Mbps
QAM128	7	315 bps–350 Mbps	45 bps–50 Mbps
QAM256	8	360 bps–400 Mbps	45 bps–50 Mbps

:BURSt:SHAPe:FALL:DELay

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:FALL:DELay <val>
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:FALL:DELay?
```

This command sets the burst shape fall delay. The variable <val> is expressed in bits with 1 bit = 1/(symbol_rate*bits_per_symbol).

To change the modulation type, refer to ":MODulation[:TYPE]" on page 241. Refer to ":SRATe" on page 242 for a list of the minimum and maximum symbol rate values.

":BURSt:SHAPe:FDELay" on page 230 performs the same function; in compliance with the SCPI standard, both commands are listed.

For concept information on burst shaping, refer to the *PSG User's Guide*.

Example

```
:RAD:CUST:BURS:SHAP:FALL:DEL 50
```

The preceding example sets a 50 bit fall delay.

***RST**	+0.00000000E+000
Range	–22.3750 to 99
Key Entry	**Fall Delay**

:BURSt:SHAPe:FALL:TIME

Supported	E8267D with Option 601or 602601 or 602

```
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:FALL:TIME <val>
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:FALL:TIME?
```

This command sets the burst shape fall time. The variable <val> is expressed in bits with 1 bit = 1/(symbol_rate*bits_per_symbol).

To change the modulation type, refer to ":MODulation[:TYPE]" on page 241. Refer to ":SRATe" on page 242 for a list of the minimum and maximum symbol rate values.

":BURSt:SHAPe:FTIMe" on page 230 performs the same function; in compliance with the SCPI standard, both commands are listed.

For concept information on burst shaping, refer to the *PSG User's Guide*.

Example

```
:RAD:CUST:BURS:SHAP:FALL:TIME 100
```

The preceding example sets a 100 bit fall delay.

***RST**	+1.00000000E+001
Range	0.1250–255.8750
Key Entry	**Fall Time**

:BURSt:SHAPe:FDELay

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:FDELay <val>
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:FDELay?
```

This command sets the burst shape fall delay. The variable <val> is expressed in bits with 1 bit = 1/(symbol_rate*bits_per_symbol).

To change the modulation type, refer to ":MODulation[:TYPE]" on page 241. Refer to ":SRATe" on page 242 for a list of the minimum and maximum symbol rate values.

":BURSt:SHAPe:FALL:DELay" on page 228 performs the same function; in compliance with the SCPI standard, both commands are listed.

For concept information on burst shaping, refer to the *PSG User's Guide*.

Example

```
:RAD:CUST:BURS:SHAP:FDEL 45
```

The preceding example sets a 45 bit fall delay.

***RST** +0.00000000E+000

Range –22.3750 to 99

Key Entry **Fall Delay**

:BURSt:SHAPe:FTIMe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:FTIMe <val>
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:FTIMe?
```

This command sets the burst shape fall time. The variable <val> is expressed in bits with 1 bit = 1/(symbol_rate*bits_per_symbol).

To change the modulation type, refer to ":MODulation[:TYPE]" on page 241. Refer to ":SRATe" on page 242 for a list of the minimum and maximum symbol rate values.

":BURSt:SHAPe:FALL:TIME" on page 229 performs the same function; in compliance with the SCPI standard, both commands are listed.

For concept information on burst shaping, refer to the *PSG User's Guide*.

Example

:RAD:CUST:BURS:SHAP:FTIM 20

The preceding example sets a 20 bit fall delay.

***RST** +0.00000000E+000

Range 0.1250–255.8750

Key Entry **Fall Time**

:BURSt:SHAPe:RDELay

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:RDELay <val>
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:RDELay?
```

This command sets the burst shape rise delay. The variable <val> is expressed in bits with 1 bit = 1/(symbol_rate*bits_per_symbol).

To change the modulation type, refer to ":MODulation[:TYPE]" on page 241. Refer to ":SRATe" on page 242 for a list of the minimum and maximum symbol rate values.

":BURSt:SHAPe:RISE:DELay" on page 231 performs the same function; in compliance with the SCPI standard, both commands are listed.

For concept information on burst shaping, refer to the *PSG User's Guide*.

Example

:RAD:CUST:BURS:SHAP:RDEL –10

The preceding example sets a –10 bit rise delay.

***RST** +0.00000000E+000

Range –17.3750 to 99

Key Entry **Rise Delay**

:BURSt:SHAPe:RISE:DELay

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:RISE:DELay <val>
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:RISE:DELay?
```

This command sets the burst shape rise delay. The variable <val> is expressed in bits with 1 bit = 1/(symbol_rate*bits_per_symbol).

To change the modulation type, refer to ":MODulation[:TYPE]" on page 241. Refer to ":SRATe" on page 242 for a list of the minimum and maximum symbol rate values.

“:BURSt:SHAPe:RDELay” on page 231 performs the same function; in compliance with the SCPI standard, both commands are listed.

For concept information on burst shaping, refer to the PSG *User's Guide*.

Example

```
:RAD:CUST:BURS:SHAP:RISE:DEL 10
```

The preceding example sets a 10 bit rise delay.

***RST** +0.00000000E+000

Range –17.3750 to 99

Key Entry **Rise Delay**

:BURSt:SHAPe:RISE:TIME

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:RISE:TIME <val>
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:RISE:TIME?
```

This command sets the burst shape rise time. The variable <val> is expressed in bits with 1 bit = 1/(symbol_rate*bits_per_symbol).

To change the modulation type, refer to “:MODulation[:TYPE]” on page 241. Refer to “:SRATe” on page 242 for a list of the minimum and maximum symbol rate values.

“:BURSt:SHAPe:RTIMe” on page 232 performs the same function; in compliance with the SCPI standard, both commands are listed.

For concept information on burst shaping, refer to the *PSG User's Guide*.

Example

```
:RAD:CUST:BURS:SHAP:RISE:TIME .5
```

The preceding example sets a .5 bit rise delay.

***RST** +1.00000000E+001

Range 0.1250–121.5000

Key Entry **Rise Time**

:BURSt:SHAPe:RTIMe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:RTIMe <val>
[:SOURce]:RADio:CUSTom:BURSt:SHAPe:RTIMe?
```

This command sets the burst shape rise time. The variable <val> is expressed in bits with 1 bit = 1/(symbol_rate*bits_per_symbol).

To change the modulation type, refer to ":MODulation[:TYPE]" on page 241. Refer to ":SRATe" on page 242 for a list of the minimum and maximum symbol rate values.

":BURSt:SHAPe:RISE:TIME" on page 232 performs the same function; in compliance with the SCPI standard, both commands are listed.

For concept information on burst shaping, refer to the *PSG User's Guide*.

Example

```
:RAD:CUST:BURS:SHAP:RTIM 100
```

The preceding example sets a 100 bit rise time.

***RST**	+1.00000000E+001
Range	0.1250–121.5000
Key Entry	**Rise Time**

:BURSt:SHAPe[:TYPE]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:BURSt:SHAPe[:TYPE] SINE|"<file_name>"
[:SOURce]:RADio:CUSTom:BURSt:SHAPe[:TYPE]?
```

This command selects a user–defined or a pre–defined burst shape file.

`SINE`	This choice selects the pre–defined Sine burst shape as the burst shape type.
`"<file_name>"`	This variable names the user burst shape file to use. Refer to 'File Name Variables' on page 12 for information on the file name syntax.

Example

```
:RAD:CUST:BURS:SHAP "Test_File"
```

The preceding example selects a file named Test_File from the signal generator's SHAPE directory. The directory path is implied in the command and does not need to be specified.

***RST**	SINE	
Key Entry	**Sine**	**User File**

:CHANnel

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:CHANnel EVM|ACP
[:SOURce]:RADio:CUSTom:CHANnel?
```

This command optimizes the Nyquist and root Nyquist filters to minimize error vector magnitude (EVM) or to minimize adjacent channel power (ACP).

EVM This choice provides the most ideal passband.

ACP This choice improves stopband rejection.

To change the current filter type, refer to ":FILTer" on page 238.

Example

```
:RAD:CUST:CHAN EVM
```

The preceding example uses EVM optimizing.

***RST** EVM

Key Entry **Optimize FIR for EVM ACP**

:DACS:ALIGn

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTOM:DACS:ALIGn
```

This command resets the signal generator's I/Q DAC circuitry. This operation is required any time the external VCO clock signal is lost and re-acquired or when an external VCO clock signal is first applied to the BASEBAND GEN CLK IN connector.

Example

```
:RAD:CUST:DACS ALIG
```

The preceding example resets the I/Q DAC circuitry.

***RST** N/A

Range N/A

Key Entry Align DACs

:DATA

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:DATA PN9|PN11|PN15|PN20|PN23|FIX4|"<file_name>"|
EXT|PRAM File|P4|P8|P16|P32|P64
[:SOURce]:RADio:CUSTom:DATA?
```

This command sets the data pattern for unframed transmission. For information on the file name syntax, see “File Name Variables” on page 12.

Example

```
:RAD:CUST:DATA PN9
```

The preceding example selects a PN9 data pattern for unframed transmission.

***RST** PN23

Key Entry **PN9** **PN11** **PN15** **PN20** **PN23** **FIX4** **User File** **Ext** **4 1’s & 4 0’s** **8 1’s & 8 0’s** **16 1’s & 16 0’s** **32 1’s & 32 0’s** **64 1’s & 64 0’s** **PRAM FILE**

:DATA:FIX4

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:DATA:FIX4 <val>
[:SOURce]:RADio:CUSTom:DATA:FIX4?
```

This command sets the binary, 4–bit repeating sequence data pattern for unframed transmission according to the modulation type, symbol rate, filter, and burst shape selected for the custom modulation format. FIX4 must be selected as the data type.

<val> This variable is an integer value from one to 15 and represents the a four bit pattern.

Example

```
:RAD:CUST:DATA:FIX4 15
```

The preceding example selects a FIX4 data pattern consisting of four 1’s.

***RST** #B0000

Range #B0000–#B1111 or 0–15

Key Entry **FIX4**

:DATA:PRAM

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:DATA:PRAM "<file_name>"
```

```
[:SOURce]:RADio:CUSTom:DATA:PRAM?
```

This command selects PRAM data as the data pattern for unframed transmission. Refer to “:DATA:PRAM:FILE:BLOCk” on page 56 for information on PRAM data. For information on the file name syntax, refer “File Name Variables” on page 12.

Example

```
:RAD:CUST:DATA:PRAM "Test_Data"
```

The preceding example selects the PRAM file, Test_Data, as the data pattern for unframed transmission.

Key Entry **PRAM File**

:DENCode

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:DENCode ON|OFF|1|0
[:SOURce]:RADio:CUSTom:DENCode?
```

This command enables or disables the differential data encoding function. Executing this command encodes the data bits prior to modulation; each modulated bit is 1 if the data bit is different from the previous one or 0 if the data bit is the same as the previous one.

Example

```
:RAD:CUST:DENC 1
```

The preceding example enables differential data encoding for the selected modulation.

***RST** 0

Key Entry **Diff Data Encode Off On**

:EDATa:DELay

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:EDATa:DELay?
```

This query returns the time delay (in symbols) from the external data input to the beginning of the symbol on the I OUT and Q OUT rear-panel connectors and the front panel RF OUTPUT connector. When the format is turned off, the delay value is unchanged; the query will return the same delay value if the format is on or off.

:EDCLock

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:EDCLock SYMBol|NORMal
[:SOURce]:RADio:CUSTom:EDCLock?
```

This command sets the external data clock use. In internal clock mode, neither choice has an effect. Refer to ":BBCLock" on page 226 to select EXT as the data clock type.

SYMBol This choice specifies that a continuous symbol clock signal must be provided to the SYMBOL SYNC input connector.

NORMal This choice specifies that the DATA CLOCK input connector requires a bit clock. The SYMBOL SYNC input connector requires a (one-shot or continuous) symbol sync signal.

Example

```
:RAD:CUST:EDCL NORM
```

The preceding example selects normal mode for the external data clock type.

***RST** NORM

Key Entry **Ext Data Clock Normal Symbol**

:EREFerence

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:EREFerence INTernal|EXTernal
[:SOURce]:RADio:CUSTom:EREFerence?
```

This command selects either an internal or external bit–clock reference for the data generator.

If the EXTernal choice is selected, the external frequency value must be applied to the BASEBAND GEN REF IN rear-panel connector. See ":EREFerence:VALue" on page 238 to enter the external reference frequency.

Example

```
:RAD:CUST:EREF EXT
```

The preceding example selects an external bit–clock reference for the data generator.

***RST** INT

Key Entry **BBG Ref Ext Int**

:EREFerence:VALue

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:EREFerence:VALue <val>
[:SOURce]:RADio:CUSTom:EREFerence:VALue?
```

This command specifies the reference frequency of the externally applied reference. The variable <val> is expressed in hertz (Hz–MHz).

The value specified by this command is valid only when an external reference is applied to the BASEBAND GEN REF IN rear-panel connector. Refer to ":EREFerence" on page 237 to select EXTernal as the reference for the bit clock reference of the data generator.

Example

```
:RAD:CUST:EREF:VAL 10E6
```

The preceding example uses a 10 MHz external reference for the signal generator's baseband generator.

***RST** +1.30000000E+007

Range 2.5E5–1E8

Key Entry **Ext BBG Ref Freq**

:FILTer

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:FILTer RNYQuist|NYQuist|GAUSsian|RECTangle|AC4Fm|
UGGaussian|"<User_FIR>"
[:SOURce]:RADio:CUSTom:FILTer?
```

This command selects the pre-modulation filter type.

RNYQuist	This choice selects a root nyquist filter (root raised cosine).
NYQuist	This choice selects a Nyquist filter (raised cosine).
GAUSsian	This choice selects a gaussian filter.
RECTangle	This choice selects a one–symbol– wide rectangular filter.
AC4Fm	This is a pre-defined Association of Public Safety Communications Officials (APCO) specified compatible 4-level frequency modulation (C4FM) filter.
UGGaussian	This choice selects a GSM Gaussian filter with a fixed Bbt value of 0.300.
"<User_FIR>"	This variable is any filter file stored in the signal generator's catalog of FIR files.

The directory path is implied in the command and does not need to be specified. For information on the file name syntax, see "File Name Variables" on page 12.

***RST** RNYQ

Example

```
:RAD:CUST:FILT GAUS
```

The preceding example selects a gaussian filter as the pre–modulation filter type.

Key Entry **Root Nyquist** **Nyquist** **Gaussian** **Rectangle** **APCO 25 C4FM** **UN3/4 GSM Gaussian** **User FIR**

:IQ:SCALe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:IQ:SCALe <val>
[:SOURce]:RADio:CUSTom:IQ:SCALe?
```

This command sets the amplitude of the I/Q outputs for better adjacent channel power (ACP); lower scaling values equate to better ACP.

The variable <val> is expressed as a percentage.

Example

```
:RAD:CUST:IQ:SCAL 50
```

The preceding example sets I/Q scaling to 50%.

***RST** +70

Range 1–200

Key Entry **I/Q Scaling**

:MODulation:FSK[:DEViation]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:MODulation:FSK[:DEViation] <val>
[:SOURce]:RADio:CUSTom:MODulation:FSK[:DEViation]?
```

This command sets the maximum symmetric FSK frequency deviation value.

The variable <val> is a numeric expression in hertz which specifies the spacing of the two outermost FSK tones. Additional tones are evenly spaced between the two outermost tones. The maximum range value equals the current symbol rate value multiplied by four and is limited to 20 MHz.

To change the modulation type, refer to ":MODulation[:TYPE]" on page 241. Refer to ":SRATe" on page 242 for a list of the minimum and maximum symbol rate values. Refer to the *PSG User's Guide* for information on setting an asymmetric FSK deviation value.

Example

```
:RAD:CUST:MOD:FSK 50KHZ
```

The preceding example sets the maximum frequency deviation to 50 kHz.

***RST** +4.00000000E+002

Range 0–2E7

Key Entry **Freq Dev**

:MODulation:MSK[:PHASe]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:MODulation:MSK[:PHASe] <val>
[:SOURce]:RADio:CUSTom:MODulation:MSK[:PHASe]?
```

This command sets the MSK phase deviation value.

The variable <val> is expressed in degrees.

Example

```
:RAD:CUST:MOD:MSK 40
```

The preceding example sets the phase deviation to 40 degrees.

***RST** +9.00000000E+001

Range 0–100

Key Entry **Phase Dev**

:MODulation:UFSK

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:MODulation:UFSK "<file_name>"
[:SOURce]:RADio:CUSTom:MODulation:UFSK?
```

This command selects a user-defined FSK file from the signal generator's catalog of FSK files. The directory path is implied in the command and does not need to be specified. For information on the file name syntax, see "File Name Variables" on page 12.

The user-defined FSK file is held in signal generator memory until the command that selects user FSK as the modulation type is sent. Refer to ":MODulation[:TYPE]" on page 241 to change the current modulation type.

Example

`:RAD:CUST:MOD:UFSK` "Test_FSK"

The preceding example selects the file, Test_FSK, from the catalog of FSK files.

Key Entry **User FSK**

:MODulation:UIQ

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:MODulation:UIQ "<file_name>"
[:SOURce]:RADio:CUSTom:MODulation:UIQ?
```

This command selects a user-defined I/Q file from the signal generator's catalog of IQ files. The directory path is implied in the command and does not need to be specified. For information on the file name syntax, see "File Name Variables" on page 12.

The user-defined I/Q file is held in signal generator memory until the command that selects user I/Q as the modulation type is sent. Refer to ":MODulation[:TYPE]" on page 241 to change the current modulation type.

Refer to ":MODulation[:TYPE]" on page 241 to change the current modulation type.

Example

`:RAD:CUST:MOD:UIQ` "Test_IQ"

The preceding example selects the file, Test_IQ, from the catalog of IQ files.

Key Entry **User I/Q**

:MODulation[:TYPE]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:MODulation[:TYPE] BPSK|QPSK|IS95QPSK|GRAYQPSK|
OQPSK|IS95OQPSK|P4DQPSK|PSK8|PSK16|D8PSK|MSK|FSK2|FSK4|FSK8|FSK16|C4FM|
QAM4|QAM16|QAM32|QAM64|QAM128|QAM256
[:SOURce]:RADio:CUSTom:MODulation[:TYPE]?
```

This command sets the modulation type for the Custom personality. For user-defined modulation, UIQ or UFSK, the file must first be specified using the ":MODulation:UFSK" or ":MODulation:UIQ" commands.

Example

```
:RAD:CUST:MOD BPSK
```

The preceding example selects binary phase shift keying (BPSK).

***RST** P4DQPSK

Key Entry **BPSK** **QPSK** **IS-95 QPSK** **Gray Coded QPSK** **OQPSK** **IS-95 OQPSK** **π/4 DQPSK** **8PSK** **16PSK** **D8PSK** **MSK** **2-Lvl FSK** **4-Lvl FSK** **8-Lvl FSK** **16-Lvl FSK** **C4FM** **4QAM** **16QAM** **32QAM** **64QAM** **128QAM** **256QAM** **User I/Q** **User FSK**

:POLarity[:ALL]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:POLarity[:ALL] NORMal|INVerted
[:SOURce]:RADio:CUSTom:POLarity[:ALL]?
```

This command sets the signal phase rotation direction.

NORMal This choice selects normal clockwise phase rotation for the signal.

INVerted This choice reverses the phase rotation of the signal by inverting the Q signal.

Example

```
:RAD:CUST:POL INV
```

The preceding example selects reverse phase rotation for the internal Q signal.

***RST** NORM

Key Entry **Phase Polarity Normal Invert**

:SRATe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:SRATe <val>
[:SOURce]:RADio:CUSTom:SRATe?
```

This command sets the transmission symbol rate.

The variable <val> is expressed in symbols per second (sps–Msps) and the maximum range value is dependent upon the source of data (internal or external), the modulation type, and filter.

When user-defined filters are selected using the command in section ":FILTer" on page 238, the upper symbol rate will be restricted using the following criteria:

- FIR filter length > 32 symbols: upper limit is 12.5 Msps
- FIR filter length > 16 symbols: upper limit is 25 Msps

When internal FIR filters are used, these limit restrictions always apply. For higher symbol rates, the FIR filter length will be truncated as follows:

- Above 12.5 Msps, the FIR length will be truncated to 32 symbols
- Above 25 Msps, the FIR length will be truncated to 16 symbols
  This will impact the relative timing of the modulated data, as well as the actual filter response.

A change in the symbol rate value effects the bit rate value.

To change the modulation type, refer to ":MODulation[:TYPE]" on page 241.

Example

```
:RAD:CUST:SRAT 10KSPS
```

The preceding example sets the symbol rate to 10K symbols per second.

***RST** +2.43000000E+004

Range

Modulation Type	*Bits per Symbol*	*Internal Data*	*External Serial Data*
BPSK	1	45 sps–50 Msps	45 sps–50 Msps
FSK2			
MSK			
C4FM	2	45 sps–50 Msps	45 sps–25 Msps
FSK4			
OQPSK			
OQPSK195			
P4QPPSK			
QAM4			
QPSK			
QPSKIS95			
QPSKISAT	2	45 sps–50 Msps	45 sps–25 Msps
D8PSK	3	45 sps–50 Msps	45 sps–16.67 Msps
EDGE			
FSK8			
PSK8			
FSK16	4	45 sps–50 Msps	45 sps–12.5 Msps
PSK16			
QAM16			
QAM32	5	45 sps–50 Msps	45 sps–10 Msps
QAM64	6	45 sps–50 Msps	45 sps–8.33 Msps
QAM128	7	45 sps–50 Msps	45 sps–7.142857142 Msps
QAM256	8	45 sps–50 Msps	45 sps–6.25 Msps

Key Entry **Symbol Rate**

:STANdard:SELect

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:STANdard:SELect NONE|AC4Fm|ACQPsk|BLUEtooth|CDPD
[:SOURce]:RADio:CUSTom:STANdard:SELect?
```

This command selects a pre-defined setup for Custom (with the appropriate defaults) and/or clears the selection.

NONE This choice clears the current pre-defined Custom format.

AC4Fm This choice sets up an Association of Public Safety Communications Officials (APCO) compliant, compatible 4-level frequency modulation (C4FM) format.

ACQPsk This choice sets up an Association of Public Safety Communications Officials (APCO) compliant, compatible quadrature phase shift keying (CQPSK) format.

BLUEtooth This choice sets up a Bluetooth (2-level frequency shift keying) format.

CDPD This choice sets up a minimum shift keying Cellular Digital Packet Data (CDPD) format.

Example

```
:RAD:CUST:STAN:SEL AC4FM
```

The preceding example selects the AC4FM pre–defined operating mode.

***RST** NONE

Key Entry **None** **APCO 25w/C4FM** **APCO 25 w/CQPSK** **Bluetooth** **CDPD**

:TRIGger:TYPE

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:TRIGger:TYPE CONTinuous|SINGle|GATE
[:SOURce]:RADio:CUSTom:TRIGger:TYPE?
```

This commands sets the trigger mode (type) that controls the data transmission.

Triggers control the data transmission by telling the PSG when to transmit the modulating signal. Depending on the trigger settings for the PSG, the data transmission can occur once, continuously, or the PSG may start and stop the transmission repeatedly (GATE mode).

A trigger signal comprises both positive and negative signal transitions (states), which are also called high and low periods. You can configure the PSG to trigger on either state of the trigger signal. It is common to have multiple triggers, also referred to as trigger occurrences or events, occur when the signal generator requires only a single trigger. In this situation, the PSG recognizes the first trigger and ignores the rest.

When you select a trigger mode, you may lose the signal (carrier plus modulating) from the RF output until you trigger the modulating signal. This is because the PSG sets the I and Q signals to zero volts prior to the first trigger event, which suppresses the carrier. After the first trigger event, the signal's final I and Q levels determine whether you see the carrier signal or not (zero = no carrier, other values = visible carrier). At the end of most data patterns, the final I and Q points are set to a value other than zero. If you create your own data file, you can set the initial I and Q voltages to values other than zero, and set the last I and Q values to zero. Create your own file using the front-panel UI (see the *PSG User's Guide*), or download a file you create external to the PSG (see the *PSG Programming Guide*).

There are four parts to configuring the trigger:

- Choosing the trigger type, which controls the data transmission.
- Setting the data pattern's response to triggers:
  - CONTinuous, see ':TRIGger:TYPE:CONTinuous[:TYPE]' on page 246
  - SINGle, selecting the mode also sets the response (This differs from using the single mode for the ARB formats.)
  - GATE, selecting the mode also sets the response
- Selecting the trigger source (see ':TRIGger[:SOURce]' on page 247), which determines how the PSG receives its trigger signal, internally or externally. The GATE choice requires an external trigger.
- Setting the trigger polarity when using an external source:
  - CONTinuous and SINGle, see ':TRIGger[:SOURce]:EXTernal:SLOPe' on page 250
  - GATE, see ":TRIGger:TYPE:GATE:ACTive" on page 247

For more information on triggering, see the *PSG User's Guide*.

The following list describes the trigger type command choices:

CONTinuous	Upon triggering, the data pattern repeats continuously.
SINGle	Upon triggering, the data pattern plays once.
GATE	An external trigger signal controls the data transmission. The modulating signal waits for the first active trigger signal state to begin. After the initial trigger, the behavior is dependent on whether the signal incorporates framed or unframed data. Because the PSG provides only unframed data for real-time custom, to transmit a framed data signal you must create an external file that incorporates the framing and download it to the PSG. The following list describes the behavior differences between the two types of data transmissions:

- For unframed data, an external trigger signal repeatedly starts and stops the data transmission. The length of each transmission depends on the duty period of the trigger signal and the gate polarity selection (see ":TRIGger:TYPE:GATE:ACTive" on page 247). Data transmits during the active polarity selection state and stops during the inactive state. The active state can be set high or low.

NOTE The real-time custom gating behavior described above is opposite to the ARB gating behavior.

- For framed data, an external trigger signals the PSG to start transmitting at the beginning of a frame during active states, but only stops at the end of a frame when the end occurs during the inactive states. If the end of the frame extends into the next active trigger state, the signal transmits continuously. For information on downloading files, see the *PSG Programming Guide*.

Example

```
:RAD:CUST:TRIG:TYPE SING
```

The preceding example selects the single trigger mode for data transmission.

***RST** CONT

Key Entry **Continuous** **Single** **Gated**

:TRIGger:TYPE:CONTinuous[:TYPE]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:TRIGger:TYPE:CONTinuous[:TYPE] FREE|TRIGger
[:SOURce]:RADio:CUSTom:TRIGger:TYPE:CONTinuous[:TYPE]?
```

This command selects the data pattern's response to a trigger signal while using the continuous trigger mode.

For more information on triggering and to select the continuous trigger mode, see ':TRIGger:TYPE' on page 244.

The following list describes the data pattern's response to each of the command choices:

FREE Turning custom on immediately triggers the modulating signal. The signal repeats the data pattern until you turn the signal off, select another trigger, or choose another data pattern.

TRIGger The modulating signal waits for a trigger before transmission begins. When the signal receives the trigger, it transmits the data continuously until you turn the signal off, select another trigger, or choose another data pattern.

Example

```
:RAD:CUST:TRIG:TYPE:CONT FREE
```

The preceding example selects the free–run mode for continuous data transmission.

***RST** FREE

Key Entry **Free Run** **Trigger & Run**

:TRIGger:TYPE:GATE:ACTive

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive LOW|HIGH
[:SOURce]:RADio:CUSTom:TRIGger:TYPE:GATE:ACTive?
```

This command selects the active state (gate polarity) of the gate while using the gating trigger mode.

The LOW and HIGH selections correspond to the low and high states of an external trigger signal. For example, when you select HIGH, the active state occurs during the high of the trigger signal. The PSG uses the active state to transmit the data pattern. When the inactive state occurs, the transmission stops at the last transmitted symbol, then restarts at the next symbol when the active state occurs. For more information on triggering and to select gating as the trigger mode, see ':TRIGger:TYPE' on page 244.

The following list describes the PSG's gating behavior for the polarity selections:

LOW The PSG transmits the data pattern while the trigger signal is low (active state) and stops when the trigger signal goes high (inactive state).

HIGH The PSG transmits the data pattern while the trigger signal is high (active state) and stops when the trigger signal goes low (inactive state).

Example

```
:RAD:CUST:TRIG:TYPE:GATE:ACT HIGH
```

The preceding example selects a high external signal level as the active state for the gate trigger.

***RST** HIGH

Key Entry **Gate Active Low High**

:TRIGger[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce] KEY|EXT|BUS
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce]?
```

This command sets the trigger source.

For more information on triggering, see ‘:TRIGger:TYPE’ on page 244. The following list describes the command choices:

KEY — This choice enables manual triggering by pressing the front-panel **Trigger** hardkey.

EXT — An externally applied signal triggers the modulating signal. This is the only choice that works with gating. The following settings affect an external trigger:

- The input connector for the trigger signal. You have a choice between the rear-panel PATTERN TRIG IN connector or the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector. To make the connector selection, see ‘:TRIGger[:SOURce]:EXTernal[:SOURce]’ on page 250.

  For more information on the connectors and on connecting the cables, see the *PSG User’s Guide*.

- The trigger signal polarity:
  - gating mode, see ‘:TRIGger:TYPE:GATE:ACTive’ on page 247
  - continuous and single modes, see “:TRIGger[:SOURce]:EXTernal:SLOPe” on page 250

- Any desired delay between when the PSG receives a trigger and when the data pattern responds to the trigger. There are two parts to setting the delay:
  - setting the amount of delay, see ‘:TRIGger[:SOURce]:EXTernal:DELay’ on page 249
  - turning the delay on, see ‘:TRIGger[:SOURce]:EXTernal:DELay:STATe’ on page 249

BUS — This choice enables triggering over the GPIB using the `*TRG` or `GET` command, or the LAN and the AUXILIARY INTERFACE (RS-232) using the `*TRG` command.

Example

```
:RAD:CUST:TRIG BUS
```

The preceding example selects BUS triggering.

***RST** EXT

Key Entry **Trigger Key** **Ext** **Bus**

:TRIGger[:SOURce]:EXTernal:DELay

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay <val>
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay?
```

This command sets the number of bits to delay the PSG’s response to an external trigger.

The bit delay is a delay between when the PSG receives the trigger and when it responds to the trigger. The delay uses the clocks of the bit-clock to time the delay. After the PSG receives the trigger and the set number of delay bits (clocks) occurs, the PSG transmits the data pattern.

The delay occurs after you enable the state. See “:TRIGger[:SOURce]:EXTernal:DELay:STATe” . You can set the number of bits either before or after enabling the state.

For more information on configuring an external trigger source and to select external as the trigger source, see ‘:TRIGger[:SOURce]’ on page 247.

Example

:RAD:CUST:TRIG:EXT:DELay 200000

The preceding example sets the delay for an external trigger for 200K bits.

***RST** +0

Range 0–1048575

Key Entry **Ext Delay Bits**

:TRIGger[:SOURce]:EXTernal:DELay:STATe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay:STATe ON|OFF|1|0
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:DELay:STATe?
```

This command turns the trigger delay on or off when using an external trigger source.

For setting the delay time, see “:TRIGger[:SOURce]:EXTernal:DELay” , and for more information on configuring an external source, see‘:TRIGger[:SOURce]’ on page 247.

Example

```
:RAD:CUST:TRIG:EXT:DEL:STAT 0
```

The preceding example disables the delay state for an external trigger source.

***RST** 0

Key Entry **Ext Delay Off On**

:TRIGger[:SOURce]:EXTernal:SLOPe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:SLOPe POSitive|NEGative
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal:SLOPe?
```

This command sets the polarity for an external trigger signal while using the continuous or single triggering modes. To set the polarity for gating, see ':TRIGger:TYPE:GATE:ACTive' on page 247.

The POSitive and NEGative selections correspond to the high (positive) and low (negative) states of the external trigger signal. For example, when you select POSitive, the waveform responds (transmits) during the high state of the trigger signal. When the PSG receives multiple trigger occurrences when only one is required, the signal generator uses the first trigger and ignores the rest.

For more information on configuring an external trigger source and to select external as the trigger source, see ':TRIGger[:SOURce]' on page 247.

Example

```
:RAD:CUST:TRIG:EXT:SLOP NEG
```

The preceding example selects the negative trigger as the active state for data transmission.

***RST** NEG

Key Entry **Ext Polarity Neg Pos**

:TRIGger[:SOURce]:EXTernal[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal[:SOURce] EPT1|EPT2|
EPTRIGGER1|EPTRIGGER2
[:SOURce]:RADio:CUSTom:TRIGger[:SOURce]:EXTernal[:SOURce]?
```

This command selects which rear-panel connector the PSG uses to accept an externally applied trigger signal when external is the trigger source selection.

For more information on configuring an external trigger source and to select external as the trigger source, see ':TRIGger[:SOURce]' on page 247. For more information on the rear-panel connectors, see the *PSG User's Guide*.

The following list describes the command choices:

EPT1 This choice is synonymous with EPTRIGGER1 and selects the PATTERN TRIG IN rear-panel connector.

EPT2 This choice is synonymous with EPTRIGGER2 and selects the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector.

EPTRIGGER1	This choice is synonymous with EPT1 and selects the PATTERN TRIG IN rear-panel connector.
EPTRIGGER2	This choice is synonymous with EPT2 and selects the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector.

Example

:RAD:CUST:TRIG:EXT EPT2

The preceding example selects an external trigger from the PATTERN TRIG IN 2 rear-panel connector.

***RST** EPT1

Key Entry **Patt Trig In 1** **Patt Trig In 2**

[:STATe]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom[:STATe] ON|OFF|1|0
[:SOURce]:RADio:CUSTom[:STATe]?
```

This command enables or disables the Custom modulation format.

Although the Custom modulation is enabled with this command, the RF carrier is not modulated unless you activate the front panel **Mod On/Off** hardkey.

Example

```
:RAD:CUST OFF
```

The preceding example turns off the custom modulation format.

***RST** 0

Key Entry **Custom Off On**

:VCO:CLOCk

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:CUSTom:VCO:CLOCk INTernal|EXTernal
[:SOURce]:RADio:CUSTom:VCO:CLOCk?
```

This command enables an internal or external VCO clock. The external VCO clock is connected to the rear-panel BASEBAND GEN CLK IN connector. Use the :DACS:ALIGn command after an external VCO clock is first applied to the BASEBAND GEN CLK IN connector or when the VCO signal is lost and then re-acquired.

Example

`:RAD:CUST:VCO:CLOC EXT`

The preceding example selects an external VCO clock.

***RST**	Int
Key Entry	**VCO Clock Ext Int**

Digital Modulation Subsystem ([:SOURce]:DM)

:EXTernal:Filter

Supported E8267D

```
[:SOURce]:DM:EXTernal:FILTer 40e6|THRough
[:SOURce]:DM:EXTernal:FILTer?
```

This command selects the filter or through path for I/Q signals routed to the rear-panel I and Q outputs.

40e6	This choice applies a 40 MHz baseband filter.
THRough	This choice bypasses filtering.

Example

```
:DM:EXT:FILT 40E6
```

The preceding example selects the 40 MHz baseband filter.

***RST** THR

Key Entry **40.000 MHz** **Through**

:EXTernal:Filter:AUTO

Supported E8267D

```
[:SOURce]:DM:EXTernal:FILTer:AUTO ON|OFF|1|0
[:SOURce]:DM:EXTernal:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals routed to the rear-panel I/Q outputs.

ON(1)	This choice automatically selects the 40 MHz filter optimized for the current signal generator settings.
OFF(0)	This choice disables the auto feature and allows you to select the 40 MHz filter or a through path. Refer to ":IQ:EXTernal:FILTer" on page 304 for selecting a filter or through path.

Example

```
:DM:EXT:FILT:AUTO 1
```

The preceding example allows automatic selection of the 40 MHz I/Q filter.

***RST** 1 (ON)

Key Entry **I/Q Output Filter Manual Auto**

:EXTernal:HCRest

Supported E8267D

```
[:SOURce]:DM:EXTernal:HCRest[STATe] ON|OFF|1|0
[:SOURce]:DM:EXTernal:HCRest [STATe]?
```

This command changes the operating condition to accommodate I/Q inputs with a high crest factor.

ON (1) This choice turns high crest mode on for externally applied signals with high crest factors. High crest mode allows the signal generator to process these signals with less distortion. For crest factors higher than 4 dB, I/Q drive levels should be reduced by 1 dB for each dB above that level. In high crest mode, the maximum output level is reduced and power level accuracy is degraded.

OFF (0) This choice disables the high crest mode.

Example

```
:DM:EXT:HCR 0
```

The preceding example disables the high crest mode.

***RST** NORM

Key Entry **High Crest Mode Off On**

:EXTernal:POLarity

Supported E8267D

```
[:SOURce]:DM:EXTernal:POLarity NORMal|INVert|INVerted
[:SOURce]:DM:EXTernal:POLarity?
```

This command, for backward compatibility with older ESG E44xxB models, selects normal or inverted I/Q signal routing. In inverted mode, the Q input is routed to the I modulator and the I input is routed to the Q modulator.

Example

```
:DM:EXT:POL INV
```

The preceding example inverts I and Q signal routing.

***RST** NORM

Key Entry **Int Phase Polarity Normal Invert**

:EXTernal:SOURce

Supported E8267D

```
[:SOURce]:DM:EXTernal:SOURce EXTernal|INTernal|BBG1|EXT600|OFF|SUM
[:SOURce]:DM:EXTernal:SOURce?
```

This command selects the I/Q signal source that is routed to the rear-panel I and Q output connectors.

EXTernal	This choice routes a portion of the externally applied signals at the 50 ohm I and Q input connectors to the rear-panel I and Q output connectors.
INTernal	This choice is for backward compatibility and performs the same function as the BBG1 selection.
BBG1	This choice routes a portion of the baseband generator I/Q signals to the rear-panel I and Q connectors and requires Option 602.
EXT600	This choice routes a portion of the externally applied signals at the 600 ohm I and Q input connectors to the rear-panel I and Q output connectors.
OFF	This choice disables the output to the rear-panel I and Q output connectors.

The output is the analog component of the I and Q signals.

For selecting the I/Q source, refer to ":SOURce" on page 269.

Example

```
:DM:EXT:SOUR EXT
```

The preceding example routes the I/Q signals to the external 50 ohm rear–panel output.

***RST** EXT

Key Entry **Ext 50 Ohm** **BBG1** **Ext 600 Ohm** **Off**

:IQADjustment:DELay

Supported E8267D

```
[:SOURce]:DM:IQADjustment:DELay <delay_val>
[:SOURce]:DM:IQADjustment:DELay?
```

This command sets a delay for both I and Q from the baseband to the I/Q outputs and to the RF output. This will allow you to time shift the I/Q with respect to triggering and markers. The absolute phase of both I and Q will change with respect to triggers and markers. A positive value advances the I and Q phase. The range limits are dependent on the current modulation format.

This feature is not compatible with constant envelope modulations and signals connected to the external I/Q inputs.

The <delay_val> variable is expressed in seconds.

Example

```
:DM:IQAD:DEL .05SEC
```

The preceding example sets a 50 millisecond delay for the I and Q signals.

***RST** +0.00000000E+000

Key Entry **I/Q Delay**

:IQADjustment:EXTernal:COFFset

Supported E8267D

```
[:SOURce]:DM:IQADjustment:EXTernal:COFFset <units>
[:SOURce]:DM:IQADjustment:EXTernal:COFFset?
```

This command sets the common mode offset voltage for both the in-phase (I) and quadrature-phase (Q) signals going to the rear-panel I and Q output connectors.

The <units> variable is expressed in volts (mV–V). his command is effective only if the state of the I/Q adjustment function is set to ON. Refer to ":IQADjustment[:STATe]" on page 263.

Example

```
:DM:IQAD:EXT:COFF -.1
```

The preceding example sets a negative .1 volt common mode offset voltage for the I and Q signals.

***RST** +0.00000000E+000

Range –3 to 3

Key Entry **Common Mode I/Q Offset**

:IQADjustment:EXTernal:DIOFfset

Supported E8267D

```
[:SOURce]:DM:IQADjustment:EXTernal:DIOFfset <val><units>
[:SOURce]:DM:IQADjustment:EXTernal:DIOFfset?
```

This command sets the differential offset voltage for an in-phase (I) signal routed to the I output connectors.

The variable <val> is a numeric expression. The <units> variable is expressed in volts (mV–V).

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to ":IQADjustment[:STATe]" on page 263.

Example

```
:DM:IQAD:EXT:DIOF 1
```

The preceding example sets a 1 volt differential offset voltage for the I signal at the rear–panel I output connector.

***RST**	+0.00000000E+000
Range	–3 to 3
Key Entry	**Diff. Mode I Offset**

:IQADjustment:EXTernal:DQOFfset

Supported	E8267D

```
[:SOURce]:DM:IQADjustment:EXTernal:DQOFfset <val><units>
[:SOURce]:DM:IQADjustment:EXTernal:DQOFfset?
```

This command sets the differential offset voltage for a quadrature-phase (Q) signal routed to the Q output connectors.

The variable <val> is a numeric expression. The <units> variable is expressed in volts (mV–V).

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to ":IQADjustment[:STATe]" on page 263.

Example

```
:DM:IQAD:EXT:DQOF 1
```

The preceding example sets a 1 volt differential offset voltage for the Q signal at the rear-panel Q connector.

***RST**	+0.00000000E+000
Range	–3 to 3
Key Entry	**Diff. Mode Q Offset**

:IQADjustment:EXTernal:GAIN

Supported	E8267D

```
[:SOURce]:DM:IQADjustment:EXTernal:GAIN[1|2] <val><units>
[:SOURce]:DM:IQADjustment:EXTernal:GAIN?
```

This command sets the I/Q gain ratio (I/Q balance) for signals routed to the rear-panel I and Q output connectors. The I signal (GAIN 1) is increased for positive values and the Q signal (GAIN 2) level increases with negative values

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to “:IQADjustment[:STATe]” on page 263.

Example

`:DM:IQAD:EXT:GAIN2 1`

The preceding example sets a Q gain ratio of 1 volt.

***RST** +0.00000000E+000

Range –4 to 4

Key Entry **I/Q Out Gain Balance**

:IQADjustment:EXTernal:IOFFset

Supported E8267D

```
[:SOURce]:DM:IQADjustment:EXTernal:IOFFset <val><units>
[:SOURce]:DM:IQADjustment:EXTernal:IOFFset?
```

This command sets the offset voltage for a signal applied to the 600 ohm I input connector.

The variable <val> is a numeric expression. The <units> variable is expressed in volts (mV–V).

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to “:IQADjustment[:STATe]” on page 263.

Example

`:DM:IQAD:EXT:IOFF 200MV`

The preceding example sets a 200 millivolt offset for the signal applied to the I 600 ohm input connector.

***RST** +0.00000000E+000

Range –5 to 5

Key Entry **Ext In 600 Ohm I Offset**

:IQADjustment:EXTernal:IQATten

Supported E8267D

```
[:SOURce]:DM:IQADjustment:EXTernal:IQATten <val><units>
[:SOURce]:DM:IQADjustment:EXTernal:IQATten?
```

This command sets the I/Q output attenuation level.

The variable <val> is a numeric expression. The <units> variable is expressed in decibels (dB).

The value set by this command is active even if the I/Q adjustment function is off.

Example

```
:DM:IQAD:EXT:IQAT 10.1
```

The preceding example sets the IQ attenuator level to 10.1 dB.

***RST** +6.00000000E+000

Range 0–40

Key Entry **I/Q Output Atten**

:IQADjustment:EXTernal:QOFFset

Supported E8267D

```
[:SOURce]:DM:IQADjustment:EXTernal:QOFFset <val><units>
[:SOURce]:DM:IQADjustment:EXTernal:QOFFset?
```

This command sets the offset voltage for a signal applied to the 600 ohm Q input connector. The variable <val> is a numeric expression. The <units> variable is expressed in volts (mV–V).

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to ":IQADjustment[:STATe]" on page 263.

Example

```
:DM:IQAD:EXT:QOFF 200MV
```

The preceding example sets a 200 millivolt offset for the signal applied to the Q 600 ohm input connector.

***RST** +0.00000000E+000

Range –5 to 5

Key Entry **Ext In 600 Ohm Q Offset**

:IQADjustment:GAIN

Supported E8267D

```
[:SOURce]:DM:IQADjustment:GAIN[1|2] <val>
[:SOURce]:DM:IQADjustment:GAIN?
```

This command sets the gain for the I signal (GAIN 1) relative to the Q signal, (GAIN 2). The gain ratio is expressed in decibels (dB).

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to

“:IQADjustment[:STATe]” on page 263.

Example

```
:DM:IQAD:GAIN2 -3
```

The preceding example sets a gain of –3 dB for the Q signal relative to the I signal.

***RST** +0.00000000E+000

Range –4 to 4 dB

Key Entry **I/Q Gain Balance Source 1**

:IQADjustment:IOFFset

Supported E8267D

```
[:SOURce]:DM:IQADjustment:IOFFset <val>
[:SOURce]:DM:IQADjustment:IOFFset?
```

This command adjusts the I channel offset value.

The <val> variable is expressed as a percent with 100% equivalent to 500 mV DC at the input connector. The minimum resolution is 0.025 percent.

When using this command to minimize the LO feedthrough signal, optimum performance is achieved when the command is sent after all other I/Q path commands are executed, such as those that change the internal phase polarity or adjust the modulator attenuator. If other adjustments are made after minimizing is performed, the LO feedthrough signal may increase.

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to “:IQADjustment[:STATe]” on page 263.

Example

```
:DM:IQAD:IOFF -30
```

The preceding example sets the I channel offset to –30%.

***RST** +0.00000000E+000

Range –5E1 to +5E1

Key Entry **I Offset**

:IQADjustment:QOFFset

Supported E8267D

```
[:SOURce]:DM:IQADjustment:QOFFset <val>
[:SOURce]:DM:IQADjustment:QOFFset?
```

This command adjusts the Q channel offset value.

The <val> variable is expressed as a percent with 100% equivalent to 500 mV DC at the input connector. The minimum resolution is 0.025 percent.

When using this command to minimize the LO feedthrough signal, optimum performance is achieved when the command is sent after all other I/Q path commands are executed, such as those that change the internal phase polarity or adjust the modulator attenuator. If other adjustments are made after minimizing is performed, the LO feedthrough signal may increase.

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to ":IQADjustment[:STATe]" on page 263.

Example

```
:DM:IQAD:QOFF -30
```

The preceding example sets the Q channel offset to –30%.

***RST** +0.00000000E+000

Range –5E1 to +5E1

Key Entry **Q Offset**

:IQADjustment:QSKew

Supported E8267D

```
[:SOURce]:DM:IQADjustment:QSKew <val>
[:SOURce]:DM:IQADjustment:QSKew?
```

This command adjusts the phase angle (quadrature skew) between the I and Q vectors by increasing or decreasing the Q phase angle.

The <val> variable is expressed in degrees with a minimum resolution of 0.1.

If the signal generator is operating at frequencies greater than 3.3 GHz, quadrature skew settings greater than ±5 degrees will not be within specifications.

Positive skew increases the angle from 90 degrees while negative skew decreases the angle from 90 degrees. When the quadrature skew is zero, the phase angle between the I and Q vectors is 90 degrees.

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to

":IQADjustment[:STATe]" on page 263.

Example

```
:DM:IQAD:QSKew 4.5
```

The preceding example increases the phase angle by 4.5 degrees.

***RST**	+0.00000000E+000
Range	–1E1 to +1E1
Key Entry	**Quadrature Angle Adjustment**

:IQADjustment:SKEW

Supported	E8267D

```
[:SOURce]:DM:IQADjustment:SKEW[:DELay] <val>
[:SOURce]:DM:IQADjustment:SKEW?
```

This command changes the input skew which is a time delay difference between the I and Q signals. Equal and opposite skew is applied to both I and Q and affects the RF Output and I/Q output paths simultaneously. A positive value delays the I signal relative to the Q signal, and a negative value delays the Q signal relative to the I signal.

If the internal I/Q correction path is set to RF or BB the I/Q signals are already optimized and adjusting I/Q skew would add an impairment to the signals. If the internal I/Q correction path is set to Off, then adjusting the I/Q skew could improve the I/Q signals. The I/Q skew adjustment cannot be performed on the MSK, FSK, and C4FM constant envelope modulations.

I/Q skew adjustments are preserved when the instrument state is saved. I/Q skew adjustments are also preserved when instrument settings are changed. If the signal generator is calibrated, the skew adjustments are added to the calibration value used for the given signal generator state. If the signal generator is uncalibrated, the skew adjustments are re-applied directly.

Using I/Q skew while playing a user FIR file greater than 32 symbols will generate an error.

The variable <val> is expressed in seconds. Range limits are determined by the modulation configuration but is limited to a maximum of ± 2 seconds.

Example

```
:DM:IQAD:SKEW .5
```

The preceding example sets the time delay difference between the I and Q signals to 500 milliseconds.

***RST**	+0.00000000E+000
Key Entry	**I/Q Timing Skew**

:IQADjustment:SKEW:Path

Supported E8267D

```
[:SOURce]:DM:IQADjustment:SKEW:PATH RF BB
[:SOURce]:DM:IQADjustment:SKEW?
```

This command selects either the RF or BB (baseband) path as the path to which skew timing corrections will be applied. If there are no factory I/Q timing skew corrections data, then adjusting the I/Q timing skew for the selected path may improve the error vector magnitude (EVM) of the signal. Refer to the ":IQADjustment:SKEW" on page 262 for more information.

If internal I/Q corrections are available for the RF or external I/Q output (BB) path then the I/Q signals are already optimized and adjusting I/Q skew for either path would add an impairment to the signal.

Example

```
:DM:IQAD:SKEW:PATH RF
```

The preceding example selects the RF path as the path to which skew timing adjustments will be made.

***RST** +0.00000000E+000

Key Entry **I/Q Timing Skew Path**

:IQADjustment[:STATe]

Supported E8267D

```
[:SOURce]:DM:IQADjustment[:STATe] ON|OFF|1|0
[:SOURce]:DM:IQADjustment[:STATe]?
```

This command enables or disables the I/Q adjustments.

Example

```
:DM:IQAD 1
```

The preceding example enables I/Q adjustments.

***RST** 0 (OFF)

Key Entry **I/Q Adjustments Off On**

:MODulation:ATTen

Supported E8267D

```
[:SOURce]:DM:MODulation:ATTen <val>
```

```
[:SOURce]:DM:MODulation:ATTen?
```

This command sets the attenuation level for the I/Q signals being modulated through the signal generator RF path. The variable <val> is expressed in decibels (dB).

Example

```
:DM:MOD:ATT 10
```

The preceding example sets the modulator attenuator to 10 dB.

***RST**	+2.00000000E+000
Range	0–40 dB
Key Entry	**Modulator Atten Manual Auto**

:MODulation:ATTen:AUTO

Supported E8267D

```
[:SOURce]:DM:MODulation:ATTen:AUTO ON|OFF|1|0
[:SOURce]:DM:MODulation:ATTen:AUTO?
```

This command enables or disables the modulator attenuator auto mode. The auto mode will be switched to manual if the signal generator receives a AUTO OFF or AUTO 0 command.

ON (1)	This choice sets the modulator attenuator to auto mode which optimizes the attenuation setting for the current signal generator settings.
OFF (0)	This choice sets the attenuator to manual mode and holds the attenuator at its current setting. Refer to ':MODulation:ATTen' on page 263 for setting the attenuation value.

Example

```
:DM:MOD:ATT:AUTO OFF
```

The preceding example sets the modulator attenuator to manual mode.

***RST**	1
Key Entry	**Modulator Atten Manual Auto**

:MODulation:ATTen:EXTernal

Supported E8267D

```
[:SOURce]:DM:MODulation:ATTen:EXTernal DEFault|MANual|MEASure
[:SOURce]:DM:MODulation:ATTen:EXTernal?
```

This command selects the external measurement used to set the attenuator level. Modulation attenuation ":MODulation:ATTen:AUTO" on page 264 must be in auto mode.

DEFault	Use this choice to set the external I/Q input level to the default value of 500.0 mV.
MANual	Use this choice to manually set the external input level. Refer to ':MODulation:ATTenn:EXTernal:LEVel' on page 265 to set the input level.
MEASurement	This choice uses a real–time measurement of the external input level to set the attenuator level. The measurement will be used to set the attenuator level setting. To perform this measurement, refer to ':MODulation:ATTenn:EXTernal:LEVel:MEASurement' on page 266.

Example

```
:DM:MOD:ATT:EXT MAN
```

The preceding example sets manual as the method for setting the external I/Q input level.

***RST** DEFault

Key Entry **Ext Input Level (nnn mV) Default Man Meas**

:MODulation:ATTenn:EXTernal:LEVel

Supported E8267D

```
[:SOURce]:DM:MODulation:ATTen:EXTernal:LEVel <val><volt_units>
[:SOURce]:DM:MODulation:ATTen:EXTernal:LEVel?
```

This command sets the I/Q signal voltage level at the external I/Q inputs. The voltage level set with this command is used as the input level setting for automatic attenuation.

Example

```
:DM:MOD:ATT:EXT:LEV 100MV
```

The preceding example sets the voltage level for the I and Q inputs to 100 millivolts.

***RST** +4.00000000E–001

Range .05–1 Volt

Key Entry **I/Q Output Atten**

:MODulation:ATTenn:EXTernal:LEVel:MEASurement

Supported E8267D

`[:SOURce]:DM:MODulation:ATTen:EXTernal:LEVel:MEASurement`

This command measures the RMS value of the external I/Q signal. The external input level must be set to **Meas**.

Key Entry **Do External Input Level Measurement**

:MODulation:ATTen:OPTimize:BANDwidth

Supported E8267D

```
[:SOURce]:DM:MODulation:ATTen:OPTimize:BANDwidth <val><rate>
[:SOURce]:DM:MODulation:ATTen:OPTimize:BANDwidth?
```

This command sets the expected bandwidth of the external I/Q signal. The bandwidth set with this command be used by the modulator attenuator for level setting.

The variable <val> is a number within the range limits and the variable <rate> is expressed as samples per second (sps, ksps, or msps).

Example

`:DM:MOD:ATT:OPT:BAND .250MSPS`

The preceding example measures the voltage level at the external I/Q inputs.

***RST** +1.00000000E+006

Range 1E3–100E6

Key Entry **Optimize for (nnn sps) Bandwidth**

:MODulation:FILTer

Supported E8267

```
[:SOURce]:DM:MODulation:FILTer 40e6|THRough
[:SOURce]:DM:MODulation:FILTer?
```

This command enables you to select a filter or through path for I/Q signals modulated onto the RF carrier. Selecting a filter with this command automatically sets “:MODulation:FILTer:AUTO” to OFF (0).

40E6 This choice applies a 40 MHz baseband filter to the I/Q signals.

THRough This choice uses through path filtering.

Example

```
:DM:MOD:FILT 40E6
```

The preceding example selects the 40 MHz filter for I/Q signals.

***RST** THR

Key Entry **40.000 MHz** **Through**

:MODulation:FILTer:AUTO

Supported E8267D

```
[:SOURce]:DM:MODulation:FILTer:AUTO ON|OFF|1|0
[:SOURce]:DM:MODulation:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals modulated onto the RF carrier.

ON (1)	This choice will automatically select the optimal filter.
OFF (0)	This choice disables the automatic filter selection and allows you to select a filter or through path. Refer to ":IQ:MODulation:FILTer" on page 277 for selecting a filter or through path.

Example

```
:DM:MOD:FILT:AUTO 0
```

The preceding example disables the automatic filter selection for I/Q signals.

***RST** 1

Key Entry **I/Q Mod Filter Manual Auto**

:POLarity[:ALL]

Supported E8267D

```
[:SOURce]:DM:POLarity[:ALL] NORMal|INVert|INVerted
[:SOURce]:DM:POLarity?
```

This command selects normal or inverted I/Q signal routing. In inverted mode, the Q input is routed to the I modulator and the I input is routed to the Q modulator, inverting the phase polarity.

NORMal	This choice selects normal routing for the I and Q signals.
INVert (ed)	This choice inverts the phase polarity by routing the I signal to the Q input of the I/Q modulator and the Q signal to the I input.

Example

```
:DM:POL INV
```

The preceding example swaps the I and Q routing paths.

***RST**	NORM
Key Entry	**Int Phase Polarity Normal Invert**

:SKEW:PATH

Supported	E8267D

```
[:SOURce]:DM:SKEW:PATH RF|BB
[:SOURce]:DM:SKEW:PATH?
```

This command selects the signal path that will be optimized using I/Q skew corrections. The other path maybe degraded.

RF	When RF is selected, the skew is optimized for the I/Q signal applied to the RF Output. The baseband (BB) output will be functional, but the I/Q skew applied will be optimized for the RF path. When using this choice, seven symbols of latency are added to the Arb based waveform. While in real-time mode, the maximum number of user symbols for the FIR is limited to 32.
BB	When BB is selected, the skew is optimized for the I/Q signal outputs on the rear-panel. The RF Output will be functional, but the I/Q skew applied will be optimized for the BB path. When using this choice, seven symbols of latency are added to the ARB based waveform. While in real-time mode, the maximum number of user symbols for the FIR is limited to 32.

NOTE You must have a skew calibration to use this command. I/Q skew corrections and calibration must be performed at an Agilent factory or service center

Example

```
:DM:SKEW:PATH BB
```

The preceding example selects the baseband path for I/Q skew and calibration.

***RST**	RF
Key Entry	**Int I/Q Skew Corrections RF BB Off**

:SKEW[:STATe]

Supported E8267D

```
[:SOURce]:DM:SKEW[:STATe] ON|OFF|1|0
[:SOURce]:DM:SKEW[:STATe]?
```

This command enables or disables the I/Q skew correction function.

Example

```
:DM:SKEW:STAT 0
```

The preceding example disables I/Q skew corrections.

***RST** 1

Key Entry **Int I / Q Skew Corrections RF BB Off**

:SOURce

Supported E8267D

```
[:SOURce]:DM:SOURce[1]|2 EXTernal|INTernal|BBG1|EXT600|OFF
[:SOURce]:DM:SOURce?
```

This command selects the I/Q modulator source for one of the two possible paths.

EXTernal	This choice selects an external 50 ohm source as the I/Q input to I/Q modulator.
INTernal	This choice is for backward compatibility with ESG E44xxB models and performs the same function as the BBG1 selection.
BBG1	This choice selects the baseband generator as the source for the I/Q modulator.
EXT600	This choice selects a 600 ohm impedance for the I and Q input connectors and routes the applied signals to the I/Q modulator.
OFF	This choice disables the I/Q input.

Example

```
:DM:SOURCE1 BBG1
```

The preceding example selects BBG1, the baseband generator, as the modulation source for path 1.

***RST** EXT

Key Entry **Ext 50 Ohm** **BBG1** **Ext 600 Ohm** **Off**

:SRATio

Supported All

```
[:SOURce]:DM:SRATio <val><units>
[:SOURce]:DM:SRATio?
```

This command enables you to set the power level difference (ratio) between the source one and the source two signals when the two signals are summed together. A positive ratio value reduces the amplitude for source two while a negative ratio value reduces the amplitude for source one.

The range for the summing ratio is dependent on the modulator attenuator setting for the signal generator that is summing the signals together. The minimum range is achieved when the modulator attenuator setting is zero and the maximum range is reached when the maximum attenuator value is used. The range can be calculated using the following formula:

± Range = 50 dB + Mod Atten

The variable <val> is expressed as a number. The variable <units> is expressed in decibels (dB).

For setting the modulator attenuator for real-time modulation formats, see ':IQ:MODulation:ATTen' on page 308 and ':IQ:MODulation:ATTen:AUTO' on page 308. For setting the modulator attenuator for Arb modulation formats, refer to the SCPI command subsystem for the Arb format being used and find the commands that contain the command mnemonics IQ:MODulation:ATTen.

Example

```
:DM:SRAT 3DB
```

The preceding example sets the summing ratio for source 1 and source 2 to 3 dB.

***RST** +0.00000000E+000

Range *Min*: ± 50 dB *Max*: ± 90 dB

Key Entry **Summing Ratio (SRC1 / SRC2) x.xx dB**

:STATe

Supported E8267D

```
[:SOURce]:DM:STATe ON|OFF|1|0
[:SOURce]:DM:STATe?
```

This command enables or disables the internal I/Q modulator. The signal generator I/Q annunciator is displayed when the I/Q modulator is on.

The I/Q modulator is enabled whenever a digital format is turned on.

Example

```
:DM:STAT OFF
```

The preceding example turns off the I/Q modulator.

***RST** 0

Key Entry **I/Q Off On**

Dual ARB Subsystem–Option 601or 602 ([:SOURce]:RADio:ARB)

:CLIPping

Supported E8267D with Option 601or 602

`[:SOURce]:RADio:ARB:CLIPping "<file_name>",IJQ|IORQ,<val>[,<val>]`

This command sets the clipping level of the selected waveform segment to a percentage of its highest peak. The waveform must be selected before the clipping command is executed. For more information about clipping, refer to the *PSG User's Guide*.

The variable `<val>` is expressed as a percentage within a 10–100% range.

IJQ This choice clips the composite I/Q waveform.

IORQ This choice clips I and Q separately. When this choice is enabled, percentage values for both I and Q must be specified.

A value of 100 percent equates to no clipping.

For information on the file name syntax, see "File Name Variables" on page 12.

Example

```
:RAD:ARB:CLIP "ramp_test_wfm",IJQ,50
```

```
:RAD:ARB:CLIP "ramp_test_wfm",IORQ,50,60
```

The preceding examples clip the `ramp_test_wfm` waveform data file. The second example clips I and Q separately to 50% and 60% respectively.

***RST** IJQ <*val*>: +100

Range <*val*>: 10–100 (0.1% resolution)

Key Entry **Clipping** **Clipping Type |I+jQ| |I|,|Q|** **Clip |I+jQ| To**
Clip |I| To **Clip |Q| To**

:DACS:ALIGn

Supported E8267D with Option 601or 602

`[:SOURce]:RADio:ARB:DACS:ALIGn`

This command resets the signal generator's I/Q DAC circuitry. This operation is required any time the external VCO clock signal is lost and re-acquired or when an external VCO clock signal is first applied to the BASEBAND GEN CLK IN connector.

***RST**	N/A
Range	N/A
Key Entry	**Align DACs**

:GENerate:SINE

Supported	E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:GENerate:SINE ["<file_name>"][,<osr>],[<scale>],
[I|Q|IQ]
```

This command creates a sine wave waveform file and saves it in the signal generator's volatile waveform memory (WFM1).

`"<file_name>"`	This variable names the file used to save the generated sine wave data.
`<osr>`	This variable sets the oversample ratio, which must be a value that is ≥ 4. If the specified over sample ratio is < 60 (the minimum number of samples or I/Q points), multiple periods are generated to create a waveform with at least 60 samples. The number of periods that will be created is 60 ÷ <osr> (quotient will round off to a whole number). A waveform with an oversample ratio ≥ 60 has one period.
`<scale>`	This variable sets the scale factor for the waveform. The scale factor must be between 0–1.
`I|Q|IQ`	The sine wave data can be applied to the I, Q, or IQ paths.

Executing this command without the `"<file_name>"` variable will generate a factory default `SINE_TEST_WFM` file. When using the variable `"<file_name>"` for this command, the "@" or ":" characters are not allowed.

Example

```
:RAD:ARB:GEN:SINE "Sine_Wave",20,.5,IQ
```

The preceding example generates an IQ sine wave and saves the data to a file named `Sine_Wave`. The oversampling ratio is 20, the scaling is set for 50%, and the data is applied to both the I and Q paths.

Range	4–32 Msamples (limited to available baseband memory)

:HEADer:CLEar

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:HEADer:CLEar
```

This command clears the header information from the header file used by this modulation format. Header information consists of signal generator settings and marker routings associated with the waveform file. Refer to the *PSG User's Guide* for information on header files.

The dual ARB must be on for this command to function. To turn on the dual ARB, see "[:STATe]" on page 303

***RST** N/A

Key Entry **Clear Header**

:HEADer:RMS

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:HEADER:RMS <"file_name">,<val>|UNSPecified
[:SOURce]:RADio:ARB:HEADER:RMS? <"file_name">
```

This command sets the RMS value in the header file for the waveform designated by the `<"file_name">` variable. The RMS value is expressed in volts. The filename variable includes the directory path and can designate a file in either the WFM1, NVWFM, or SEQ directories. For information on the file name syntax, refer to "File Name Variables" on page 12 and "ARB Waveform File Directories" on page 13. When a file is created with no header information then a header is automatically generated with all fields set to unspecified.

The `<val>` variable is the user-measured RMS value for the specified waveform. The `UNSPecified` parameter means that the signal generator will calculate the RMS value when it is needed. The signal generator calculation includes rise times and does not include consecutive zero level samples. DC offsets and noise are also included in the RMS measurement. Because the RMS calculation, done by the signal generator, is slow and may not be appropriate for your application it is recommended that the user calculate and enter in their measured RMS value for the waveform file.

The RMS value is calculated as:

$$\sqrt{\sum_{n=1}^{N}\left(i_n^2 + q_n^2\right) * \frac{1}{N}}$$

N = # of Samples

Example

`[:SOURce]:RADio:ARB:HEADER:RMS "WFM1:Sine_Wave",.835`

`:RAD:ARB:HEADER:RMS "WFM1:Sine_Wave",UNSP`

The first example sets a user-measured RMS value for the `Sine_Wave` waveform file in the waveform's header file. The second example, the signal generator will calculate the RMS value when needed.

***RST**	N/A
Range	0 – 1.414213562373095
Key Entry	N/A

:HEADer:SAVE

Supported	E8267D with Option 601or 602

`[:SOURce]:RADio:ARB:HEADer:SAVE`

This command saves the header information to the header file used by this modulation format. Header information consists of signal generator settings and marker routings associated with the waveform file. Refer to the *PSG User's Guide* for information on header files.

The dual ARB must be on for this command to function. To turn on the dual ARB, see '[:STATe]' on page 303

***RST**	N/A
Key Entry	**Save Setup To Header**

:IQ:EXTernal:FILTer

Supported	E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:IQ:EXTernal:FILTer 40e6|THRough
[:SOURce]:RADio:ARB:IQ:EXTernal:FILTer?
```

This command selects the filter or through path for I/Q signals routed to the rear–panel I and Q outputs. The filter has no effect on the modulated RF signal. Selecting a filter using this command will automatically set ':IQ:EXTernal:FILTer:AUTO' on page 276 to OFF(0) mode.

40e6	This choice applies a 40 MHz baseband filter.
THRough	This choice selects the through path.

Example

```
:RAD:ARB:IQ:EXT:FILT 40E6
```

The preceding example selects a 40 MHz filter for the I/Q signals routed to the rear panel.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:EXTernal:FILTer:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:IQ:EXTernal:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:ARB:IQ:EXTernal:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals routed to the rear-panel I/Q outputs.

`ON(1)` This choice automatically selects the 40 MHz filter optimized for the current signal generator settings.

`OFF(0)` This choice disables the auto feature and allows you to select the 40 MHz filter or a through path. Refer to “:IQ:EXTernal:FILTer” on page 304 for selecting a filter or through path.

Example

```
:RAD:ARB:IQ:EXT:FILT:AUTO OFF
```

The preceding example disables the automatic filter selection.

***RST** 1

Key Entry **I/Q Output Filter Manual Auto**

:IQ:MODulation:ATTen

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:IQ:MODulation:ATTen <val><units>
[:SOURce]:RADio:ARB:IQ:MODulation:ATTen?
```

This command sets the attenuation level of the I/Q signals being modulated through the signal generator RF path. The variable `<val>` is expressed in decibels (dB)

Example

```
:RAD:ARB:IQ:MOD:ATT 20
```

The preceding example sets the attenuator level to 20 dB.

***RST** +2.00000000E+000

Range 0–40

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:ATTen:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:IQ:MODulation:ATTen:AUTO ON|OFF|1|0
[:SOURce]:RADio:ARB:IQ:MODulation:ATTen:AUTO?
```

This command enables or disables the modulator attenuator auto mode. The auto mode will be switched to manual if the signal generator receives an `AUTO OFF` or `AUTO 0` command.

`ON` (`1`) This choice sets the modulator attenuator to auto mode which optimizes the attenuation setting for the current signal generator settings.

`OFF` (`0`) This choice sets the attenuator to manual mode and holds the attenuator at its current setting. Refer to ‘:IQ:MODulation:ATTen’ on page 276 for setting the attenuation value.

Example

```
:RAD:ARB:IQ:MOD:ATT:AUTO 0
```

The preceding example selects the modulator attenuator manual mode.

***RST** 1

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:FILTer

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:IQ:MODulation:FILTer 40e6|THRough
[:SOURce]:RADio:ARB:IQ:MODulation:FILTer?
```

This command enables you to select a filter or through path for I/Q signals modulated onto the RF carrier. This filter has no effect on the I/Q signal out the rear-panel. Selecting a filter using this command will automatically set ‘:IQ:MODulation:FILTer:AUTO’ on page 278 to OFF(0) mode.

40E6 This choice applies a 40 MHz baseband filter to the I/Q signals.

THRough This choice selects the through path.

Example

```
:RAD:ARB:IQ:MOD:FILT 40E6
```

The preceding example selects a 40 MHz filter.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:MODulation:FILTer:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:IQ:MODulation:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:ARB:IQ:MODulation:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals modulated onto the RF carrier.

`ON (1)` This choice will automatically select optimized filters for the current signal generator setting.

`OFF (0)` This choice disables the automatic filter selection and allows you to select a digital modulation filter or through path. Refer to ":IQ:MODulation:FILTer" on page 277 for selecting a filter or through path.

Example

```
:RAD:ARB:IQ:MOD:FILT:AUTO 1
```

The preceding example allows for automatic filter selection.

***RST** 1

Key Entry **I/Q Mod Filter Manual Auto**

:MARKer:CLEar

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:MARKer:CLEar "<file_name>",<marker>,<first_point>,
<last_point>
```

This command clears a single marker point or a range of marker points on a waveform segment for the selected marker (1–4). The Dual ARB mode and all of the ARB modes use this command.

`"<file_name>"` This variable specifies the name of the waveform file in volatile waveform memory (WFM1). Use the AUTOGEN_WAVEFORM file when clearing marker points for the currently active ARB format and then save the file using a different file name. The PSG automatically creates a file, using current settings, and names it AUTOGEN_WAVEFORM whenever an ARB format is turned on (except Dual ARB); the same file name is used for all ARB formats. When all ARB formats are off, this file will still be in waveform memory (WFM1) and is available for use by the Dual ARB. For information on the file name syntax, see “File Name Variables” on page 12.

`<marker>` This variable selects the marker number; an integer value from one to four.

`<first_point>` This variable defines the first point in a range of points. The number must be greater than or equal to one, and less than or equal to the total number of waveform points.

If you enter a value for either the first marker point or the last marker point that would make the first marker point occur after the last, the last marker point automatically adjusts to match the first marker point.

`<last_point>` This variable defines the last point in a range of points. The number must be greater than or equal to the first point, and less than or equal to the total number of waveform points.

To clear a single marker point, use the same marker point for the first and last point variables. For more information on markers and ARB files, refer to the *PSG User's Guide*.

Example

```
:RAD:ARB:MARK:CLE "Test_Data",1,1,300
```

The preceding example clears marker 1 from the first point through the 300th point in the `Test_Data` file.

Range *<marker>*: 1–4

<first_Point>: 1–number of waveform points

<last_point>: <first_Point>–number of waveform points

Key Entry **Set Marker Off Range Of Points** **Marker 1 2 3 4** **First Mkr Point** **Last Mkr Point**

:MARKer:CLEar:ALL

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:MARKer:CLEar:ALL "<file_name>",<marker>
```

This command clears all marker points on a waveform segment for the selected marker (1–4). The Dual ARB player and all of the ARB formats use this command. With all marker points cleared, the event output signal level is set low.

`"<file_name>"` This variable specifies the name of the waveform file in volatile waveform memory (WFM1). Use the AUTOGEN_WAVEFORM file when clearing all marker points for the currently active ARB format and then save the file using a different file name. The PSG automatically creates a file, using current settings, and names it AUTOGEN_WAVEFORM whenever an ARB format is turned on (except Dual ARB); the same file name is used for all ARB formats. When all ARB formats are off, this file will still be in waveform memory (WFM1) and is available for use by the Dual ARB. For information on the file name syntax, see “File Name Variables” on page 12

`<marker>` This variable selects the marker number; an integer value from one to four.

Example

```
:RAD:ARB:MARK:CLE:ALL "Test_Data",1
```

The preceding example clears marker 1 from the all waveform points in the `Test_Data` file.

Range 1–4

Key Entry **Marker 1 2 3 4** **Set Marker Off All Points**

:MARKer:ROTate

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:MARKer:ROTate "<file_name>",<rotate_count>
```

This command shifts the marker points for all markers in a waveform earlier or later by the value of the `<rotate_count>` variable. The Dual ARB player and all of the ARB formats use this command.

You can use a positive or negative value. When a marker point is close to the end of the waveform and the `<rotate_count>` value is greater than the number of remaining marker points, but less than the total number of marker points, the marker points that would move beyond the end of the waveform wrap to the beginning of the waveform. For example, if a marker point resides at sample point 195 out of 200, and the `<rotate_count>` value is twenty-five, the marker point wraps to the beginning of the waveform and continues out to the twentieth waveform point.

To set the marker points in a waveform, refer to “:MARKer:[SET]” on page 281.

`"<file_name>"` This variable specifies the name of the waveform file in volatile waveform memory (WFM1). Use the AUTOGEN_WAVEFORM file when rotating marker points for the currently active ARB format and then save the file using a different file name. The PSG automatically creates a file, using current settings, and names it AUTOGEN_WAVEFORM whenever an ARB format is turned on (except Dual ARB); the same file name is used for all ARB formats. When all ARB formats are off, this file will still be in waveform memory (WFM1) and is available for use by the Dual ARB. For information on the file name syntax, see “File Name Variables” on page 12.

Example

```
:RAD:ARB:MARK:ROT "Test_Data",100
```

The preceding example shifts all markers set in the `Test_Data` file 100 points later. If the first set point in the file is at 50, then after sending this command, the first set point will be 150 (assuming the `Test_Data` file has at least 150 points) and no later set points wrapped around to the beginning of the file.

Range – (n – 1) to (n – 1)

n = number of points in the waveform

:MARKer:[SET]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:MARKer:[SET] "<file_name>",<marker>,<first_point>,
<last_point>,<skip_count>
```

This command sets a single marker point or a range of marker points on a waveform segment for the selected marker (1–4). The Dual ARB player and all of the ARB formats use this command.

The PSG provides four independent markers. Each marker routes an output signal to the rear-panel event connector number (BNC—EVENT 1 and EVENT 2 or AUXILIARY I/O—EVENT 3 and EVENT 4) that corresponds to the marker number. A marker consists of marker points placed at defined sample points in a waveform segment. This means that a marker point cannot be less than one or greater than the last sample point in the waveform. Marker points are cumulative, so multiple command executions with different range values, without first clearing the existing points, places additional marker points on the waveform. Because of this cumulative behavior, it is a good practice to clear existing marker points prior to setting new points. This will eliminate unexpected marker pulses. Refer to ‘:MARKer:CLEar’ on page 278 and ‘:MARKer:CLEar:ALL’ on page 279 for information on clearing marker points.

For waveforms generated on the signal generator (baseband generator), the PSG automatically places a marker point at the first waveform sample for markers one and two.

NOTE You can set markers for either positive or negative polarity. The following discussions for this command assume positive marker polarity. When using negative marker polarity, the marker pulses occur during the periods of no marker points.

There are three ways to place marker points using this command:

- consecutive marker points over a range that collectively create a single marker pulse that spans the range
- equally spaced marker points over a range, so that a marker pulse occurs at each sample point that coincides with a marker point (Using this method, you can configure a clock signal by setting the <skip_count> variable to one.)
- a single marker point placed at a specific sample point in the waveform, which outputs a single

pulse relative to the marker point location (To configure a single marker point, set the first and last points to the same number.)

For more information on markers, refer to the *PSG User's Guide*.

The following list describes the command variables:

`"<file_name>"`	This variable specifies the name of the waveform file in volatile waveform memory (WFM1). Use the AUTOGEN_WAVEFORM file when setting marker points for the currently active ARB format and then save the file using a different file name. The PSG automatically creates a file, using current settings, and names it AUTOGEN_WAVEFORM whenever an ARB format is turned on (except Dual ARB); the same file name is used for all ARB formats. When all ARB formats are off, this file will still be in waveform memory (WFM1) and is available for use by the Dual ARB. For information on the file name syntax, see "File Name Variables" on page 12
`<marker>`	This variable selects the marker number; an integer value from one to four.
`<first_point>`	This variable defines the first point in the range over which the marker is placed. This number must be greater than or equal to one, and less than or equal to the total number of waveform points. If you enter a value for either the first marker point or the last marker point that would make the first marker point occur after the last, the last marker point is automatically adjusted to match the first marker point.
`<last_point>`	This variable defines the last point in the range over which the marker will be placed. This value must be greater than or equal to the first point, and less than or equal to the total number of waveform points.
`<skip_count>`	This variable defines the marker point pattern across the range. A zero value means the marker points occur consecutively across the range. A value greater than zero creates a repeating marker point pattern across the range, where the gap between the marker points is equal to the `<skip_count>` value. The gaps begin after the first marker point. Each marker point in the pattern, which is only one point wide, produces a marker pulse.

Example

```
:RAD:ARB:MARK "Test_Data",1,40,100,2
```

The preceding example sets marker 1 on the first point, `40`, the last point, `100`, and every third point (skip `2`) between 40 and 100 (assuming the `Test_Data` file has at least 100 points).

Range *<marker>*: 1–4

<first_Point>: 1–number of waveform points

<last_point>: <first_Point>–number of waveform points

<*skip_count*>: 0–number of points in the range

Key Entry **Set Marker on Range Of Points** **Marker 1 2 3 4** **First Mkr Point** **Last Mkr Point** **# Skipped Points** **Apply to Waveform**

:MDEStination:ALCHold

Supported E8267D with Option 601or 602

CAUTION Incorrect ALC sampling can create a sudden unleveled condition that may create a spike in the RF output potentially damaging a DUT or connected instrument. Ensure that you set markers to let the ALC sample over an amplitude that accounts for the high power levels within the signal.

```
[:SOURce]:RADio:ARB:MDEStination:ALCHold NONE|M1|M2|M3|M4
[:SOURce]:RADio:ARB:MDEStination:ALCHold?
```

This command disables the marker ALC hold function, or it enables the marker hold function for the selected marker. For setting markers, see “:MARKer:[SET]” on page 281.

Use the ALC hold function when you have a waveform signal that incorporates idle periods, or when the increased dynamic range encountered with RF blanking is not desired. The ALC leveling circuitry responds to the marker signal during the marker pulse (marker signal high), averaging the modulated signal level during this period.

The ALC hold function operates during the low periods of the marker signal. The marker polarity determines when the marker signal is high. For a positive polarity, this is during the marker points. For a negative polarity, this is when there are no marker points. For setting a marker’s polarity, see ‘:MPOLarity:MARKer1|2|3|4’ on page 285.

NOTE Do not use the ALC hold for more than 100 ms, because it can affect the waveform’s output amplitude.

The marker signal has a minimum of a two-sample delay in its response relative to the waveform signal response. To compensate for the marker signal delay, offset marker points from the waveform sample point at which you want the ALC sampling to begin.

The ALC hold setting is part of the file header information, so saving the setting to the file header saves the current marker routing for the waveform file.

NOTE A waveform file that has unspecified settings in the file header uses the previous waveform's routing settings.

For more information on the marker ALC hold function, see the *PSG User's Guide*. For setting the marker points, see ':MARKer:[SET]' on page 281.

NONE This terminates the marker ALC hold function.

M1–M4 These are the marker choices. The ALC hold feature uses only one marker at a time.

***RST** NONE

Example

```
:RAD:ARB:MDES:ALCH M1
```

The preceding example routes marker 1 to the ALC Hold function.

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

Remarks N/A

:MDEStination:PULSe

Supported E8267D with Option 601or 602

CAUTION The pulse function incorporates ALC hold. Incorrect ALC sampling can create a sudden unleveled condition that may create a spike in the RF output potentially damaging a DUT or connected instrument. Ensure that you set markers to let the ALC sample over an amplitude that accounts for the high power levels within the signal.

```
[:SOURce]:RADio:ARB:MDEStination:PULSe NONE|M1|M2|M3|M4
[:SOURce]:RADio:ARB:MDEStination:PULSe?
```

This command disables the marker pulse/RF blanking function, or it enables the marker pulse/RF blanking function for the selected marker.

This function automatically incorporates the ALC hold function, so there is no need to select both functions for the same marker.

NOTE Do not use ALC hold for more than 100 ms, because it can affect the waveform's output amplitude.

The signal generator blanks the RF output when the marker signal goes low. The marker polarity determines when the marker signal is low. For a positive polarity, this is during the marker points. For a negative polarity, this is when there are no marker points. For setting a marker's polarity, see ':MPOLarity:MARKer1|2|3|4' on page 285.

NOTE Set marker points prior to using this function. Enabling this function without setting marker points may create a continuous low or high marker signal, depending on the marker polarity. This causes either no RF output or a continuous RF output. See ':MARKer:[SET]' on page 281 for setting the marker points.

The marker signal has a minimum of a two-sample delay in its response relative to the waveform signal response. To compensate for the marker signal delay, offset marker points from the waveform sample point at which you want the RF blanking to begin. The RF blanking setting is part of the file header information, so saving the setting to the file header saves the current marker routing for the waveform file.

NOTE A waveform file that has unspecified settings in the file header uses the previous waveform's routing settings. This could create the situation where there is no RF output signal, because the previous waveform used RF blanking.

For more information on the marker RF blanking function, see the *PSG User's Guide*.

NONE This terminates the marker RF blanking/pulse function.

M1–M4 These are the marker choices. The RF blanking/pulse feature uses only one marker at a time.

Example

```
:RAD:ARB:MDES:PULS M2
```

The preceding example routes marker 2 to Pulse/RF Blanking.

***RST** NONE

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

:MPOLarity:MARKer1|2|3|4

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:MPOLarity:MARKer1|2|3|4 NEGative|POSitive
[:SOURce]:RADio:ARB:MPOLarity:MARKer1|2|3|4?
```

This command sets the polarity for the selected marker. For a positive marker polarity, the marker signal is high during the marker points. For a negative marker polarity, the marker signal is high during the period of no marker points.

Example

```
:RAD:ARB:MPOL:MARK3 NEG
```

The preceding example sets the polarity for marker 3 to negative.

***RST**	POS
Key Entry	**Marker 1 Polarity Neg Pos** **Marker 2 Polarity Neg Pos** **Marker 3 Polarity Neg Pos** **Marker 4 Polarity Neg Pos**

:NOISe

Supported E8267D with Option 601or 602 and Option 403

```
[:SOURce]:RADio:ARB:NOISe[:STATe] ON|OFF|1|0
[:SOURce]:RADio:ARB:NOISe[:STATe]?
```

This command enables or disables adding real-time, non-repeating additive white gaussian noise (AWGN) to the carrier modulated by the waveform being played by the Dual ARB waveform player. The noise bandwidth will be at least 0.8 times the sample rate, or 1.6 times the sample rate depending on the bandwidth factor. For information on the bandwidth factor, refer to ":NOISe:BFACtor".

When the bandwidth factor is 2, and the sample rate is greater than 50 Msamples/sec, noise cannot be enabled. Maximum bandwidth cannot exceed 80 MHz. Any oversampling in the waveform increases the noise bandwidth by a factor equal to the oversampling.

Example

```
:RAD:ARB:NOIS ON
```

The preceding example applies real-time AWGN to the carrier.

***RST**	0
Key Entry	**Real-time Noise Off On**

:NOISe:BFACtor

Supported E8267D with Option 601or 602 and Option 403

```
[:SOURce]:RADio:ARB:NOISe:BFACtor <1 - 2>
[:SOURce]:RADio:ARB:NOISe:BFACtor?
```

This command sets the flat noise bandwidth for applied real time noise. The bandwidth factor will set the noise bandwidth to at least 0.8 times the sample rate when the bandwidth factor is 1 or to 1.6 times the sample rate if the bandwidth factor is 2. Maximum bandwidth cannot exceed 80 MHz.

When the bandwidth factor is 2, and the sample rate is greater than 50 megasamples/sec, noise cannot be enabled. Any oversampling in the waveform increases the noise bandwidth by a factor equal to the oversampling.

Example

```
:RAD:ARB:NOIS:BFAC 2
```

The preceding example sets the bandwidth factor to 2 and increases the flat noise bandwidth by at least 1.6 times the ARB sample clock rate.

***RST** +1

Key Entry **Noise Bandwidth Factor**

:NOISe:CBWidth

```
[:SOURce]:RADio:ARB:NOISe:CBWidth <1Hz-80MHz>
[:SOURce]:RADio:ARB:NOISe:CBWidth?
```

Supported E8267D with Option 601or 602 and Option 403

This command selects the carrier bandwidth over which the AWGN (additive white gaussian noise) is applied. The noise power will be integrated over the selected bandwidth for the purposes of calculating C/N (carrier to noise ratio). The carrier bandwidth is limited to the ARB sample rate but cannot exceed 80 MHz. For more information refer to ":NOISe" and ":NOISe:BFACtor" on page 286.

***RST** +1.00000000E+000

1.0 Hz

Range 1Hz – 80 MHz

Key Entry **Carrier Bandwidth**

:NOISe:CN

Supported E8267D with Option 601or 602 and Option 403

```
[:SOURce]:RADio:ARB:NOISe:CN <-100dB - 100dB>
[:SOURce]:RADio:ARB:NOISe:CN?
```

This command sets the carrier to noise ratio in dB. The carrier power is defined as the total modulated signal power without noise power added. The noise power is applied over the specified bandwidth of the carrier signal. For more information, refer to ":NOISe:CBWidth".

Example

```
:RAD:ARB:NOIS:CN 50DB
```

The preceding example sets the carrier to noise ratio to 50 dB.

***RST** +0.00000000E+000

Key Entry **Carrier to Noise Ratio**

:REFerence:EXTernal:FREQuency

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:REFerence:EXTernal:FREQuency <val>
[:SOURce]:RADio:ARB:REFerence:EXTernal:FREQuency?
```

This command allows you to enter the frequency of the external reference.

The variable `<val>` is expressed in hertz (Hz–MHz).

The value specified by this command is effective only when you are using an external ARB reference applied to the BASEBAND GEN REF IN rear-panel connector.

To specify external as the ARB reference frequency you must set the ARB reference to external. Refer to “:REFerence[:SOURce]” on page 288 for more information.

Example

```
:RAD:ARB:REF:EXT:FREQ 500KHZ
```

The preceding example sets the external clock frequency reference to 500 kHz.

***RST** +1.00000000E+007

Range 2.5E5–1E8

Key Entry **Reference Freq**

:REFerence[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:REFerence[:SOURce] INTernal|EXTernal
[:SOURce]:RADio:ARB:REFerence[:SOURce]?
```

This command selects either an internal or external reference for the waveform clock.

If the EXTernal choice is selected, the external frequency value *must* be entered and the signal must be applied to the BASEBAND GEN REF IN rear-panel connector.

Refer to “:REFerence:EXTernal:FREQuency” on page 288 to enter the external reference frequency.

Example

```
:RAD:ARB:REF EXT
```

The preceding example sets the ARB reference to external.

***RST** INT

Key Entry **ARB Reference Ext Int**

:RETRigger

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:RETRigger ON|OFF|IMMediate
[:SOURce]:RADio:ARB:RETRigger?
```

This commands selects the signal generator's response to a trigger signal while using the single trigger mode.

When the PSG receives multiple trigger occurrences, when only one is required, it uses the first trigger and ignores the rest. For more information on triggering and to select the single trigger mode, see ':TRIGger:TYPE' on page 293.

The following list describes the waveform's response to each of the command choices:

ON The waveform waits for a trigger before play begins and accepts a subsequent trigger during playback. If there is a subsequent trigger during playback, the waveform completes its current playback and then plays one more time. If there is no subsequent trigger, the waveform plays once and stops until it receives another trigger.

OFF The waveform waits for a trigger before play begins and ignores triggers during playback. To restart the waveform, you must send a trigger after the playback completes.

IMMediate The waveform waits for a trigger before play begins and accepts a subsequent trigger during playback. Upon receipt of the subsequent trigger, the waveform immediately resets and begins playing from the beginning of the file. For a waveform sequence, this means to the beginning of the first segment in the sequence.

Example

```
:RAD:ARB:RETR IMM
```

The preceding example selects the immediate mode for the single mode trigger.

***RST** ON

Key Entry **On** **Off** **Immediate**

:RSCAling

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:RSCaling <val>
[:SOURce]:RADio:ARB:RSCaling?
```

This command adjusts the scaling value that is applied to a waveform while it is playing. The variable `<val>` is expressed as a percentage. Runtime scaling does not alter the waveform data file. For more information about runtime scaling, refer to the *PSG User's Guide*.

Example

```
:RAD:ARB:RSC 50
```

The preceding example applies a 50% scaling factor to the selected waveform.

***RST** +7.00000000E+001

Range 1–100

Key Entry **Waveform Runtime Scaling**

:SCALing

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:SCALing "<file_name>",<val>
```

This command scales the designated `"<file_name>"` waveform file while it is being played by the Dual ARB player. The variable `<val>` is expressed as a percentage, 1–100%. For information on file name syntax, see
"File Name Variables" on page 12.

Scaling is additive and permanent. You cannot scale up. If you scale a waveform file by 60% and then scale it again to 80% you will scale down the 60% waveform file. For more information about waveform file scaling, refer to the *PSG User's Guide*.

Example

```
:RAD:ARB:SCAL "Test_Data", 50
```

The preceding example applies a 50% scaling factor to the `Test_Data` waveform file.

Range 1–100

Key Entry **Scaling** **Scale Waveform Data**

:SCLock:RATE

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:SCLock:RATE <sample_clock_rate>
[:SOURce]:RADio:ARB:SCLock:RATE?
```

This command sets the ARB sample clock rate. The `sample_clock_rate` variable can be set from 1 hertz to 100 megahertz.

Example

```
:RAD:ARB:SCL:RATE 1E6
```

The preceding example sets the ARB sample clock for 1 MHz.

***RST**	+1.00000000E+008
Range	1–1.0E8 Hz
Key Entry	**ARB Sample Clock**

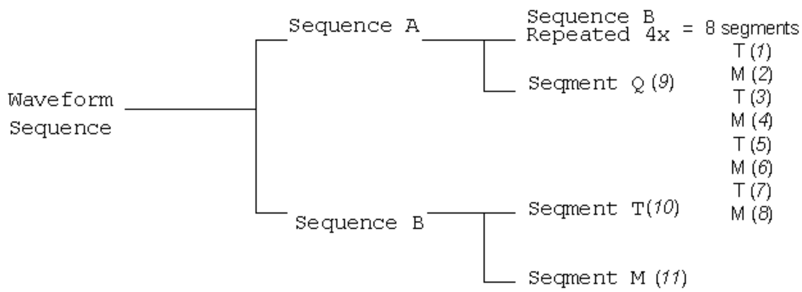
:SEQuence

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:SEQuence
"<file_name>","<waveform1>",<reps>,NONE|M1|M2|M3|M4|M1M2|M1M3|M1M4|M2M3|
M2M4|M3M4|M1M2M3|M1M2M4|M1M3M4|M2M3M4|ALL,{"<waveform2>",<reps>,NONE|M1|
M2|M3|M4|M1M2|M1M3|M1M4|M2M3|M2M4|M3M4|M1M2M3|M1M2M4|M1M3M4|M2M3M4|ALL}
[:SOURce]:RADio:ARB:SEQuence? "<file_name>"
```

This command creates a waveform sequence. A waveform sequence is made up of segments and other sequences. Any number of segments, up to a segment count limit of 32768, can be used to create a sequence. The count limit is determined by the number of segments in the waveform sequence. Repeated segments are included in the count limit.

For example, using the figure below, suppose a waveform is created using two sequences: Sequence_A and Sequence_B. Sequence_A consists of Sequence_B and Segment_Q with Sequence_B repeated four times. The total segment count for this waveform sequence would be eleven.



The query returns the contents and segment settings of the waveform sequence file

The segments and sequences play in the same order as placed into the waveform sequence by the command. Once you create the file, you cannot edit the segment settings or add further waveform segments unless you use the signal generator's front panel. Using the same waveform sequence name overwrites the existing file with that name. To use a segment's marker settings, you must enable the segment's markers within the segment or within the waveform sequence. A sequence is stored in the catalog of SEQ files `USER/SEQ` or `SEQ:` directory.

When you create a waveform sequence, the PSG also creates a file header for the sequence. This file header takes priority over segment or nested sequence file headers. Refer to the *PSG User's Guide* for more information on file headers. To save the file header, see ':HEADer:SAVE' on page 275.

`"<file_name>"` This variable names the waveform *sequence* file. For information on the file name syntax, see "File Name Variables" on page 12.

`"<waveform1>"` This variable specifies the name of an existing waveform *segment* or *sequence* file. A waveform segment or the waveform segments in a specified sequence must reside in volatile memory, WFM1, before it can be played by the Dual ARB player. For information on the file name syntax, see "File Name Variables" on page 12, and for more information on waveform segments, see the *PSG User's Guide*.

`"<waveform2>"` This variable specifies the name of a second existing waveform *segment* or *sequence* file. The same conditions required for `waveform1` apply for this segment or sequence. Additional segments and other sequences can be inserted into the file.

`<reps>`	This variable sets the number of times a segment or sequence plays (repeats) before the next segment or sequence plays.
`NONE`	This choice disables all four markers for the waveform. Disabling markers means that the waveform sequence ignores the segment's or sequence's marker settings.
`M1, M2, M3, M4`	These choices, either individually or a combination of them, enable the markers for the waveform segment or sequence. Markers not specified are ignored for that segment or sequence.
`ALL`	This choice enables all four markers in the waveform segment or sequence.

Example

```
:RAD:ARB:SEQ "SEQ:Test_Data","WFM1:ramp_test_wfm",25,M1M4,
"WFM1:sine_test_wfm",100,ALL
```

NOTE A carriage return or line feed is never included in a SCPI command. The example above contains a carriage return so that the text will fit on the page.

The preceding example creates a waveform sequence file named `Test_Data`. This file consists of the factory-supplied waveform segments, `ramp_test_wfm` and `sine_test_wfm`. The waveform is stored in the signal generator's `SEQ:` directory.

- The first segment, ramp_test_wfm, has 25 repetitions with markers 1 and 4 enabled.
- The second segment, sine_test_wfm, has 100 repetitions with all four markers enabled.

Range <reps>: 1–65535

Key Entry **Build New Waveform Sequence** **Name and Store** **Insert Waveform** **Edit Repetitions** **Toggle Marker 1** **Toggle Marker 2** **Toggle Marker 3** **Toggle Marker 4**

:TRIGger:TYPE

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:TRIGger:TYPE CONTinuous|SINGle|GATE|SADVance
[:SOURce]:RADio:ARB:TRIGger:TYPE?
```

This command sets the trigger mode (type) that controls how the waveform plays.

Triggers control the playback by telling the PSG when to transmit the modulating signal (waveform). Depending on the trigger settings for the PSG, the waveform transmission can occur once, continuously, or the PSG may start and stop the transmission repeatedly (GATE mode). For waveform sequences, you can even control when each segment plays (SADVance—segment advance mode).

A trigger signal comprises both positive and negative signal transitions (states), which are also called high and low periods. You can configure the PSG to trigger on either state. It is common to have multiple triggers occur when the signal generator requires only a single trigger. In this situation, the PSG recognizes the first trigger and ignores the rest.

When you select a trigger mode, you may lose the signal (carrier plus modulating) from the RF output until you trigger the waveform. This is because the PSG sets the I and Q signals to zero volts prior to the first trigger event, which suppresses the carrier. After the first trigger event, the waveform's final I and Q levels determine whether you will see the carrier signal or not (zero = no carrier, other values = carrier visible). At the end of most files, the final I and Q points are set to a value other than zero. If desired, you can create and download an external file (see the *PSG Programming Guide*) with the initial I and Q voltages set to values other than zero. Conversely, you can set the last I and Q points to zero.

There are four parts to configuring the trigger:

- Choosing the trigger type, which controls the waveform's transmission.
- Setting the waveform's response to triggers:
  - — CONTinuous, see ':TRIGger:TYPE:CONTinuous[:TYPE]' on page 295
  - — SINGle, see ':TRIGger:TYPE:CONTinuous[:TYPE]' on page 246
  - — SADVance, see ":TRIGger:TYPE:SADVance[:TYPE]" on page 296
  - — GATE, selecting the mode also sets the response
- Selecting the trigger source (see ':TRIGger[:SOURce]' on page 298), which determines how the PSG receives its trigger signal, internally or externally. The GATE choice requires an external trigger.
- Setting the trigger polarity when using an external source:
  - — CONTinuous, SINGle, and SADVance, see ':TRIGger[:SOURce]:EXTernal:SLOPe' on page 301
  - — GATE, see ":TRIGger:TYPE:GATE:ACTive" on page 296

For more information on triggering, see the *PSG User's Guide*.

The following list describes the trigger type command choices:

CONTinuous	Upon triggering, the waveform repeats continuously.
SINGle	Upon triggering, the waveform segment or sequence plays once.
GATE	An external trigger signal repeatedly starts and stops the waveform's playback (transmission). The length of each transmission depends on the duty period of the trigger signal and the gate polarity selection (see ":TRIGger:TYPE:GATE:ACTive" on page 296). The waveform plays during the inactive state and stops during the active polarity selection state. The active state

can be set high or low. The gate mode works only with an external trigger source.

NOTE The ARB gating behavior described above is opposite to the gating behavior for real-time custom.

SADVance	The trigger controls the segment advance within a waveform sequence. To use this choice, a waveform sequence must be the active waveform. Ensure that all segments in the sequence reside in volatile memory.
***RST**	CONT

Example

```
:RAD:ARB:TRIG:TYPE GATE
```

The preceding example selects the gated trigger mode.

Key Entry **Continuous** **Single** **Gated** **Segment Advance**

:TRIGger:TYPE:CONTinuous[:TYPE]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:TRIGger:TYPE:CONTinuous[:TYPE] FREE|TRIGger|RESet
[:SOURce]:RADio:ARB:TRIGger:TYPE:CONTinuous[:TYPE]?
```

This commands selects the waveform's response to a trigger signal while using the continuous trigger mode.

For more information on triggering and to select the continuous trigger mode, see ":TRIGger:TYPE" on page 293.

The following list describes the waveform's response to each of the command choices:

FREE	Turning the ARB format on immediately triggers the waveform. The waveform repeats until you turn the format off, select another trigger, or choose another waveform file.
TRIGger	The waveform waits for a trigger before play begins. When the waveform receives the trigger, it plays continuously until you turn the format off, select another trigger, or choose another waveform file.
RESet	The waveform waits for a trigger before play begins. When the waveform receives the trigger, it plays continuously. Subsequent triggers reset the waveform to the beginning. For a waveform sequence, this means to the beginning of the first segment in the sequence.

Example

```
:RAD:ARB:TRIG:TYPE:CONT TRIG
```

The preceding example selects the trigger continuous mode.

***RST** FREE

Key Entry **Free Run** **Trigger & Run** **Reset & Run**

:TRIGger:TYPE:GATE:ACTive

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:TRIGger:TYPE:GATE:ACTive LOW|HIGH
[:SOURce]:RADio:ARB:TRIGger:TYPE:GATE:ACTive?
```

This command selects the active state (gate polarity) of the gate while using the gating trigger mode.

The LOW and HIGH selections correspond to the low and high states of an external trigger signal. For example, when you select HIGH, the active state occurs during the high of the trigger signal. When the active state occurs, the PSG stops the waveform playback at the last played sample point, then restarts the playback at the next sample point when the inactive state occurs. For more information on triggering and to select gating as the trigger mode, see ":TRIGger:TYPE" on page 293.

The following list describes the PSG's gating behavior for the polarity selections:

LOW The waveform playback stops when the trigger signal goes low (active state) and restarts when the trigger signal goes high (inactive state).

HIGH The waveform playback stops when the trigger signal goes high (active state) and restarts when the trigger signal goes low (inactive state).

Example

```
:RAD:ARB:TRIG:TYPE:GATE:ACTIVE HIGH
```

The preceding example sets the active gate state to high.

***RST** HIGH

Key Entry **Gate Active Low High**

:TRIGger:TYPE:SADVance[:TYPE]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:TRIGger:TYPE:SADVance[:TYPE] SINGle|CONTinuous
[:SOURce]:RADio:ARB:TRIGger:TYPE:SADVance[:TYPE]?
```

This commands selects the waveform's response to a trigger signal while using the segment advance

(SADVance) trigger mode.

When the PSG receives multiple trigger occurrences when only one is required, the signal generator uses the first trigger and ignores the rest. For more information on triggering and to select segment advance as the trigger mode, see ‘:TRIGger:TYPE’ on page 293.

The following list describes the waveform’s response to each of the command choices:

SINGle — Each segment in the sequence requires a trigger to play, and a segment plays only once, ignoring a segment’s repetition value (see ‘:SEQuence’ on page 291 for repetition information). The following list describes a sequence’s playback behavior with this choice:

- After receiving the first trigger, the first segment plays to completion.
- When the waveform receives a trigger after a segment completes, the sequence advances to the next segment and plays that segment to completion.
- When the waveform receives a trigger during play, the current segment plays to completion. Then the sequence advances to the next segment, and it plays to completion.
- When the waveform receives a trigger either during or after the last segment in a sequence plays, the sequence resets and the first segment plays to completion.

CONTinuous — Each segment in the sequence requires a trigger to play. After receiving a trigger, a segment plays continuously until the waveform receives another trigger. The following list describes a sequence’s playback behavior with this choice:

- After receiving the first trigger, the first segment plays continuously.
- A trigger during the current segment play causes the segment to play to the end of the segment file, then the sequence advances to the next segment, which plays continuously.
- When last segment in the sequence receives a trigger, the sequence resets and the first segment plays continuously.

Example

```
:RAD:ARB:TRIG:TYPE:SADV CONT
```

The preceding example selects the continuous segment advance mode.

***RST** CONT

Key Entry **Single** **Continuous**

:TRIGger[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:TRIGger[:SOURce] KEY|EXT|BUS
[:SOURce]:RADio:ARB:TRIGger[:SOURce]?
```

This command sets the trigger source.

For more information on triggering, see ‘:TRIGger:TYPE’ on page 293. The following list describes the command choices:

KEY This choice enables manual triggering by pressing the front-panel **Trigger** hardkey.

EXT An externally applied signal triggers the waveform. This is the only choice that works with gating. The following settings affect an external trigger:

- The input connector for the trigger signal. You have a choice between the rear-panel PATTERN TRIG IN connector or the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector. To make the connector selection, see ‘:TRIGger[:SOURce]:EXTernal[:SOURce]’ on page 299.

  For more information on the connectors and on connecting the cables, see the *PSG User’s Guide*.

- The trigger signal polarity:

  — gating mode, see ‘:TRIGger:TYPE:GATE:ACTive’ on page 296
  — continuous, single, and segment advance modes, see ‘:TRIGger[:SOURce]:EXTernal:SLOPe’ on page 301

- The time delay between when the PSG receives a trigger and when the waveform responds to the trigger. There are two parts to setting the delay:

  — setting the amount of delay, see ‘:TRIGger[SOURce]:EXTernal:DELay’ on page 299
  — turning the delay on, see ‘:TRIGger[:SOURce]:EXTernal:DELay:STATe’ on page 300

BUS This choice enables triggering over the GPIB using the `*TRG` or `GET` commands, or the LAN and the AUXILIARY INTERFACE (RS-232) using the `*TRG` command.

Example

```
:RAD:ARB:TRIG KEY
```

The preceding example sets the trigger source to manual, front–panel key operation.

***RST** EXT

Key Entry **Trigger Key** **Ext** **Bus**

:TRIGger[:SOURce]:EXTernal[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:TRIGger[:SOURce]:EXTernal[:SOURce] EPT1|EPT2|
EPTRIGGER1|EPTRIGGER2
[:SOURce]:RADio:ARB:TRIGger[:SOURce]:EXTernal[:SOURce]?
```

This command selects which PATTERN TRIG IN connection the PSG uses to accept an externally applied trigger signal when external is the trigger source selection.

For more information on configuring an external trigger source and to select external as the trigger source, see “:TRIGger[:SOURce]” on page 298. For more information on the rear-panel connectors, see the *PSG User’s Guide*.

The following list describes the command choices:

EPT1	This choice is synonymous with EPTRIGGER1 and selects the PATTERN TRIG IN rear-panel connector.
EPT2	This choice is synonymous with EPTRIGGER2 and selects the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector.
EPTRIGGER1	This choice is synonymous with EPT1 and selects the PATTERN TRIG IN rear-panel connector.
EPTRIGGER2	This choice is synonymous with EPT2 and selects the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector.

Example

```
:RAD:ARB:TRIG:EXT EPT2
```

The preceding example sets the trigger source to the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector.

***RST** EPT1

Key Entry **Patt Trig In 1** **Patt Trig In 2**

:TRIGger[SOURce]:EXTernal:DELay

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay <val>
[:SOURce]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay?
```

This command sets the amount of time to delay the PSG's response to an external trigger.

The delay is a path (time) delay between when the PSG receives the trigger and when it responds to the trigger. For example, configuring a trigger delay of two seconds, causes the PSG to wait two seconds after receipt of the trigger before the PSG responds and transmits the waveform.

The delay does not occur until you enable it (see ":TRIGger[:SOURce]:EXTernal:DELay:STATe"). You can set the delay value either before or after turning it on.

For more information on configuring an external trigger source and to select external as the trigger source, see ":TRIGger[:SOURce]" on page 298.

The unit of measurement for the variable <val> is in seconds (nsec–sec).

Example

```
:RAD:ARB:TRIG:EXT:DEL .2
```

The preceding example sets the external delay to 200 milliseconds.

***RST**	+1.00000000E–003
Range	1E–8 to 4E1
Key Entry	**Ext Delay Time**

:TRIGger[:SOURce]:EXTernal:DELay:STATe

Supported	E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay:STATe ON|OFF|1|0
[:SOURce]:RADio:ARB:TRIGger[:SOURce]:EXTernal:DELay:STATe?
```

This command turns the trigger delay on or off when using an external trigger source.

For setting the delay time, see ":TRIGger[SOURce]:EXTernal:DELay" , and for more information on configuring an external source, see":TRIGger[:SOURce]" on page 298.

Example

```
:RAD:ARB:TRIG:EXT:DEL:STAT OFF
```

The preceding example disables the external delay function.

***RST**	0
Key Entry	**Ext Delay Off On**

:TRIGger[:SOURce]:EXTernal:SLOPe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:TRIGger[:SOURce]:EXTernal:SLOPe POSitive|NEGative
[:SOURce]:RADio:ARB:TRIGger[:SOURce]:EXTernal:SLOPe?
```

This command sets the polarity for an external trigger signal while using the continuous, single, or segment advance triggering modes. To set the polarity for gating, see ':TRIGger:TYPE:GATE:ACTive' on page 296.

The POSitive and NEGative selections correspond to the high (positive) and low (negative) states of the external trigger signal. For example, when you select POSitive, the waveform responds (plays) during the high state of the trigger signal. When the PSG receives multiple trigger occurrences when only one is required, the signal generator uses the first trigger and ignores the rest.

For more information on configuring an external trigger source and to select external as the trigger source, see ":TRIGger[:SOURce]" on page 298.

Example

```
:RAD:ARB:TRIG:EXT:SLOP NEG
```

The preceding example sets the external trigger slope to negative.

***RST** NEG

Key Entry **Ext Polarity Neg Pos**

:VCO:CLOCk

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:VCO:CLOCk INTernal|EXTernal
[:SOURce]:RADio:ARB:VCO:CLOCk?
```

This command selects an internal or external VCO clock. the external VCO clock is connected to the rear-panel `BASEBAND GEN CLK IN` connector. Use the `:DACS:ALIGn` command after an external VCO clock is first applied to the `BASEBAND GEN CLK IN` connector or when the VCO signal is lost and then re-acquired.

Example

```
:RAD:ARB:VCO:CLOC EXT
```

The preceding example selects an external VCO clock.

***RST** Int

Key Entry **VCO Clock Ext Int**

:WAVeform

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:WAVeform "WFM1:file_name"|"SEQ:filename"
[:SOURce]:RADio:ARB:WAVeform?
```

This command, for the Dual ARB mode, selects a waveform file or sequence, for the Dual ARB player to play. The file must be present in volatile memory, `WFM1:` or in the SEQ directory. If a file is in non-volatile memory (NVWFM), use the command ":COPY" on page 67 to copy the file to WFM1.

`"WFM1:file_name"` This variable names a waveform file residing in volatile memory:WFM1. For information on the file name syntax, see "File Name Variables" on page 12.

`"SEQ:filename"` This variable names a sequence file residing in the catalog of sequence files. For more information on the file name syntax, see "File Name Variables" on page 12.

Example

```
:RAD:ARB:WAV "WFM1:Test_Data"
```

The preceding example selects the file `Test_Data` from the list of files in volatile waveform memory, WFM1 and applies its header settings.

Key Entry **Select Waveform**

:Waveform:NHEAders

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB:WAVeform:NHEAders "WFM1:file_name"|"SEQ:filename"
[:SOURce]:RADio:ARB:WAVeform:NHEAders?
```

This command, for the Dual ARB mode, allows for a fast selection of a waveform file or sequence. No header information or settings are applied to the waveform or sequence when this command is used. This will improve the access or loading speed of the waveform file or sequence to approximately 100 mS for a single segment. The file must be in volatile waveform memory, `WFM1:` or in the SEQ directory. If a file is in non-volatile memory (NVWFM), use the command ":COPY" on page 67 to copy files to WFM1.

`"WFM1:file_name"` This variable names a waveform file residing in volatile memory:WFM1. For information on the file name syntax, see "File Name Variables" on page 12.

`"SEQ:filename"` This variable names a sequence file residing in the catalog of sequence files. For more information on the file name syntax, see "File Name Variables" on page 12.

Example

```
:RAD:ARB:WAV:NHEA "Test_Data"
```

The preceding example selects the file `Test_Data`, without applying header settings.

[:STATe]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:ARB[:STATe] ON|OFF|1|0
[:SOURce]:RADio:ARB[:STATe]?
```

This command enables or disables the operating state of the signal generator's dual arbitrary waveform (ARB) generator.

Example

```
:RAD:ARB 1
```

The preceding example turns on the signal generator's ARB generator personality.

***RST** 0

Key Entry **ARB Off On**

Dmodulation Subsystem–Option 601or 602 ([:SOURce]:RADio:DMODulation:ARB)

:IQ:EXTernal:FILTer

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:IQ:EXTernal:FILTer 40e6|THRough
[:SOURce]:RADio:DMODulation:ARB:IQ:EXTernal:FILTer?
```

This command selects a 40 MHz filter or a through path for I/Q signals routed to the rear-panel I and Q outputs. Selecting a filter using this command will automatically set ‘:IQ:EXTernal:FILTer:AUTO’ on page 304 to `OFF(0)` mode.

40e6 This choice selects the 40 MHz baseband filter.

THRough This choice selects a through path and bypasses filtering.

Example

```
:RAD:DMOD:ARB:IQ:EXT:FILT 40E6
```

The preceding example selects a 40 MHz filter.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:EXTernal:FILTer:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:IQ:EXTernal:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:DMODulation:ARB:IQ:EXTernal:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals routed to the rear-panel I/Q outputs.

`ON(1)` This choice automatically selects a filter that is optimized for the current signal generator settings.

`OFF(0)` This choice disables the auto feature and allows you to select the 40 MHz filter or a through path. Refer to “:IQ:EXTernal:FILTer” on page 304 for selecting a filter or through path.

Example

```
:RAD:DMOD:ARB:IQ:EXT:FILT:AUTO 0
```

The preceding example disables the auto mode filter selection.

***RST** 1

Key Entry **I/Q Output Filter Manual Auto**

:FILTer

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:FILTer RNYQuist|NYQuist|GAUSsian|
RECTangle|AC4Fm|UGGaussian|"<user_FIR>"
[:SOURce]:RADio:DMODulation:ARB:FILTer?
```

This command specifies the pre-modulation filter type.

`RNYQuist`	This choice selects a Root Nyquist (root raised cosine) filter. This filter is adjusted using Alpha.
`NYQuist`	This choice selects a Nyquist (raised cosine) filter. This filter is adjusted using Alpha.
`GAUSsian`	This choice selects a Gaussian Filter which is adjusted using Bbt values.
`RECTangle`	This choice selects a one symbol wide rectangular filter.
`AC4Fm`	This choice selects a pre-defined Association of Public Safety Communications officials (APCO) specified compatible 4-level frequency modulation (C4FM) filter.
`UGGAUSsian`	This choice selects a UN3/4 delay-compatible, GSM, 0.300 Bbt Gaussian filter. The Bbt value is not adjustable.
"<User_FIR>"	This variable is any filter file that you have stored in memory. For information on the file name syntax, see “File Name Variables” on page 12.

Example

```
:RAD:DMOD:ARB:FILTer "FIR:FIR_Data"
```

The preceding example selects a file named `FIR_Data`, from the catalog of FIR files, as the filter type.

***RST** RNYQuist

Key Entry **Root Nyquist** **Nyquist** **Gaussian** **Rectangle** **APCO 25 C4FM** **UN3/4 GSM Gaussian** **User FIR**

:FILTer:ALPHa

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:FILTer:ALPHa <val>
[:SOURce]:RADio:DMODulation:ARB:FILTer:ALPHa?
```

This command changes the Nyquist or root Nyquist filter alpha value.

The filter alpha value can be set to the minimum level (`0`), the maximum level (`1`), or in between by using numeric values (0.001–0.999).

To change the current filter type, refer to ":FILTer" on page 305.

Example

```
:RAD:DMOD:ARB:FILT:ALPH .33
```

The preceding example sets .33 as the filter alpha.

***RST** +3.50000000E–001

Range 0.000–1.000

Key Entry **Filter Alpha**

:FILTer:BBT

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:FILTer:BBT <val>
[:SOURce]:RADio:DMODulation:ARB:FILTer:BBT?
```

This command changes the bandwidth-multiplied-by-bit-time (BbT) filter parameter for a Gaussian filter. It has no effect on other types of filters.

The filter BbT value can be set to the minimum level (0), the maximum level (1), or in between by using fractional numeric values (0.001–0.999).

To change the current filter type, refer to ":FILTer" on page 305.

Example

```
:RAD:DMOD:ARB:FILT:BBT .52
```

The preceding example sets .52 as the filter BbT.

***RST** +5.00000000E–001

Range 0.000–1.000

Key Entry **Filter BbT**

:FILTer:CHANnel

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:FILTer:CHANnel EVM|ACP
[:SOURce]:RADio:DMODulation:ARB:FILTer:CHANnel?
```

This command optimizes the Nyquist and root Nyquist filters to minimize error vector magnitude (EVM) or to minimize adjacent channel power (ACP). To change the current filter type, refer to ":FILTer" on page 305.

Example

```
:RAD:DMOD:ARB:FILT:CHAN ACP
```

The preceding example selects ACP optimization.

EVM This choice provides the most ideal passband.

ACP This choice improves stopband rejection.

***RST** EVM

Key Entry **Optimize FIR For EVM ACP**

:HEADer:CLEar

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:HEADer:CLEar
```

This command clears the header information from the header file used by this modulation format. Header information consists of signal generator settings and marker routings associated with the waveform file. Refer to the *PSG User's Guide* for information on header files.

For this command to function, the Arb Waveform Generator's Digital Modulation must be on. To turn Digital Modulation on, see '[:STATe]' on page 329.

***RST** N/A

Key Entry **Clear Header**

:HEADer:SAVE

Supported E8267D with Option 601or 602

`[:SOURce]:RADio:DMODulation:ARB:HEADer:SAVE`

This command saves the header information to the header file for the active modulation file. Header information consists of signal generator settings and marker routings associated with the waveform file. Refer to the *PSG User's Guide* for information on header files.

For this command to function, the Arb Waveform Generator's Digital Modulation must be on. To turn Digital Modulation on, see '[:STATe]' on page 329.

***RST** N/A

Key Entry **Save Setup To Header**

:IQ:MODulation:ATTen

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:IQ:MODulation:ATTen <val><unit>
[:SOURce]:RADio:DMODulation:ARB:IQ:MODulation:ATTen?
```

This command sets the attenuation level of the I/Q signals being modulated through the signal generator RF path. The variable `<val>` is expressed in decibels (dB).

Example

`:RAD:DMOD:ARB:IQ:MOD:ATT 20`

The preceding example sets the modulator attenuator level to 20 dB.

***RST** +2.00000000E+000

Range 0–40 dB

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:ATTen:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:IQ:MODulation:ATTen:AUTO ON|OFF|1|0
[:SOURce]:RADio:DMODulation:ARB:IQ:MODulation:ATTen:AUTO?
```

This command enables or disables the modulator attenuator auto mode. The auto mode will be switched to manual if the signal generator receives a `AUTO OFF` or `AUTO 0` command.

`ON (1)` This choice enables the attenuation auto mode which optimizes the modulator attenuation for the current conditions.

`OFF` (`0`) This choice holds the attenuator at its current setting or at a selected value. Refer to ‘:IQ:MODulation:ATTen’ on page 308 for setting the attenuation value.

Example

```
:RAD:DMOD:ARB:IQ:MOD:ATT:AUTO ON
```

The preceding example selects the modulator attenuator auto mode.

***RST** 1

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:FILTer

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:IQ:MODulation:FILTer 40e6|THRough
[:SOURce]:RADio:DMODulation:ARB:IQ:MODulation:FILTer?
```

This command enables you to select a filter or through path for I/Q signals modulated onto the RF carrier. Selecting a filter using this command will automatically set ‘:IQ:MODulation:FILTer:AUTO’ on page 309 to OFF(0) mode.

40E6 This choice applies a 40 MHz baseband filter to the I/Q signals.

THRough This choice bypasses filtering.

Example

```
:RAD:DMOD:ARB:IQ:MOD:FILT THR
```

The preceding example selects the through path and bypasses filtering.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:MODulation:FILTer:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:IQ:MODulation:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:DMODulation:ARB:IQ:MODulation:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals modulated onto the RF carrier.

`ON` (`1`) This choice will automatically select a filter that is optimized for the current signal generator setting.

`OFF` (`0`) This choice disables the automatic filter selection and allows you to select a digital modulation filter or through path. Refer to “:IQ:MODulation:FILTer” on page 277 for selecting a filter or through path.

Example

`:RAD:DMOD:ARB:IQ:MOD:FILT:AUTO ON`

The preceding example sets the automatic filter selection function.

***RST** 1

Key Entry **I/Q Mod Filter Manual Auto**

:MDEStination:ALCHold

Supported E8267D with Option 601or 602

CAUTION Incorrect ALC sampling can create a sudden unleveled condition that may create a spike in the RF output potentially damaging a DUT or connected instrument. Ensure that you set markers to let the ALC sample over an amplitude that accounts for the high power levels within the signal.

```
[:SOURce]:RADio:DMODulation:ARB:MDEStination:ALCHold NONE|M1|M2|M3|M4
[:SOURce]:RADio:DMODulation:ARB:MDEStination:ALCHold?
```

This command disables the marker ALC hold function, or it enables the marker hold function for the selected marker.

Use the ALC hold function when you have a waveform signal that uses idle periods, or when the increased dynamic range encountered with RF blanking is not desired. The ALC circuitry responds to the marker signal during the marker pulse (marker signal high), averaging the modulated signal level during this period.

The ALC hold function operates during the low periods of the marker signal. The marker polarity determines when the marker signal is high. For a positive polarity, this is during the marker points. For a negative polarity, this is when there are no marker points. To set a marker’s polarity, see ‘:MPOLarity:MARKer1|2|3|4’ on page 314. For more information on markers, see ‘:MARKer:[SET]’ on page 281.

NOTE Do not use the ALC hold for more than 100 ms, because it can affect the waveform’s output amplitude.

The marker signal has a minimum of a two-sample delay in its response relative to the waveform signal response. To compensate for the marker signal delay, offset marker points from the waveform sample point at which you want the ALC sampling to begin.

The ALC hold setting is part of the file header information, so saving the setting to the file header saves the current marker routing for the waveform file.

NOTE A waveform file that has unspecified settings in the file header uses the previous waveform’s routing settings.

For more information on the marker ALC hold function, see the *PSG User’s Guide*. To configure marker points, refer to the following sections located in the Dual ARB subsystem:

- For clearing a single marker point or a range of marker points, see ‘:MARKer:CLEar’ on page 278.
- For clearing all marker points, see ‘:MARKer:CLEar:ALL’ on page 279.
- For shifting marker points, see ‘:MARKer:ROTate’ on page 280.
- For setting marker points, see ‘:MARKer:[SET]’ on page 281.

NONE This terminates the marker ALC hold function.

M1–M4 These are the marker choices. The ALC hold feature uses only one marker at a time.

Example

```
:RAD:DMOD:ARB:MDES:ALCH M1
```

The preceding example routes marker 1 to the ALC Hold function.

***RST** NONE

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

Remarks N/A

:MDEStination:PULSe

Supported E8267D with Option 601or 602

CAUTION The pulse function incorporates ALC hold. Incorrect ALC sampling can create a sudden unleveled condition that may create a spike in the RF output potentially damaging a DUT or connected instrument. Ensure that you set markers to let the ALC sample over an amplitude that accounts for the high power levels within the signal.

```
[:SOURce]:RADio:DMODulation:ARB:MDEStination:PULSe NONE|M1|M2|M3|M4
[:SOURce]:RADio:DMODulation:ARB:MDEStination:PULSe?
```

This command disables the marker RF blanking/pulse function, or it enables the marker RF blanking/pulse function for the selected marker.

This function automatically incorporates the ALC hold function, so there is no need to select both functions for the same marker.

NOTE Do not use ALC hold for more than 100 ms, because it can affect the waveform's output amplitude.

The signal generator blanks the RF output when the marker signal goes low. The marker polarity determines when the marker signal is low. For a positive polarity, this is during the marker points. For a negative polarity, this is when there are no marker points. To set a marker's polarity, see ':MPOLarity:MARKer1|2|3|4' on page 314. For more information on markers, see ':MARKer:[SET]' on page 281.

NOTE Set marker points prior to using this function. Enabling this function without setting marker points may create a continuous low or high marker signal, depending on the marker polarity. This creates the condition where there is either no RF output or a continuous RF output.

To configure marker points, refer to the following sections located in the Dual ARB subsystem:

- For clearing a single marker point or a range of marker points, see':MARKer:CLEar' on page 278.
- For clearing all marker points, see ':MARKer:CLEar:ALL' on page 279.
- For shifting marker points, see ':MARKer:ROTate' on page 280.
- For setting marker points, see ':MARKer:[SET]' on page 281.

The marker signal has a minimum of a two-sample delay in its response relative to the waveform signal response. To compensate for the marker signal delay, offset marker points from the waveform sample point at which you want the RF blanking to begin.

The RF blanking setting is part of the file header information, so saving the setting to the file header saves the current marker routing for the waveform file.

NOTE A waveform file that has unspecified settings in the file header uses the previous waveform's routing settings. This could create the situation where there is no RF output signal, because the previous waveform used RF blanking

For more information on the marker RF blanking function, see the *PSG User's Guide*.

NONE	This terminates the marker RF blanking/pulse function.
M1–M4	These are the marker choices. The RF blanking/pulse feature uses only one marker at a time.

Example

```
:RAD:DMOD:ARB:MDES:PULS M2
```

The preceding example routes marker 2 to the Pulse/RF Blanking function.

***RST** NONE

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

:MODulation:FSK[:DEViation]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:MODulation:FSK[:DEViation] <val><units>
[:SOURce]:RADio:DMODulation:ARB:MODulation:FSK[:DEViation]?
```

This command sets the symmetric FSK frequency deviation value.

The variable `<val>` is a numeric expression with a maximum range equal to the current symbol rate value multiplied by ten, limited to 20 MHz. The variable `<units>` is expressed in hertz.

To change the modulation type, refer to the command ":MODulation[:TYPE]" on page 314. Refer to the command ":SRATe" on page 321 for a list of the minimum and maximum symbol rate values.

For more information on setting an asymmetric FSK deviation value, refer to the *PSG User's Guide*.

Example

```
:RAD:DMOD:ARB:MOD:FSK 50KHZ
```

The preceding example sets the maximum frequency deviation to 50 kHz.

***RST** +4.00000000E+002

Range 0–2E7

Key Entry **Freq Dev**

:MODulation[:TYPE]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:MODulation[:TYPE] BPSK|QPSK|IS95QPSK|
GRAYQPSK|OQPSK|IS95OQPSK|P4DQPSK|PSK8|PSK16|D8PSK|EDGE|MSK|FSK2|FSK4|
FSK8|FSK16|C4FM|QAM4|QAM16|QAM32|QAM64|QAM128|QAM256
[:SOURce]:RADio:DMODulation:ARB:MODulation[:TYPE]?
```

This command sets the modulation type for the digital modulation personality.

Example

```
:RAD:DMOD:ARB:MOD BPSK
```

The preceding example selects binary phase shift keying (BPSK) as the modulation type.

***RST** P4DQPSK

Key Entry **BPSK** **QPSK** **IS-95 QPSK** **Gray Coded QPSK** **OQPSK**
IS-95 OQPSK **π/4 DQPSK** **8PSK** **16PSK** **D8PSK** **EDGE** **MSK**
2-Lvl FSK **4-Lvl FSK** **8-Lvl FSK** **16-Lvl FSK** **C4FM** **4QAM** **16QAM**
32QAM **64QAM** **128QAM** **256QAM** **User I/Q** **User FSK**

:MPOLarity:MARKer1|2|3|4

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:MPOLarity:MARKer1|2|3|4 NEGative|
POSitive
[:SOURce]:RADio:DMODulation:ARB:MPOLarity:MARKer1|2|3|4?
```

This command sets the polarity for the selected marker.

For a positive marker polarity, the marker signal is high during the marker points. For a negative marker polarity, the marker signal is high during the period of no marker points. To configure marker points, refer to the following sections located in the Dual ARB subsystem:

- For clearing a single marker point or a range of marker points, see‘:MARKer:CLEar’ on page 278.
- For clearing all marker points, see ‘:MARKer:CLEar:ALL’ on page 279.
- For shifting marker points, see ‘:MARKer:ROTate’ on page 280.
- For information on markers and setting marker points, see ‘:MARKer:[SET]’ on page 281.

Example

```
:RAD:DMOD:ARB:MPOL:MARK2 NEG
```

The preceding example sets the polarity for marker 2 to negative.

***RST** POS

Key Entry **Marker 1 Polarity Neg Pos** **Marker 2 Polarity Neg Pos** **Marker 3 Polarity Neg Pos** **Marker 4 Polarity Neg Pos**

:REFerence:EXTernal:FREQuency

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:REFerence:EXTernal:FREQuency <val>
[:SOURce]:RADio:DMODulation:ARB:REFerence:EXTernal:FREQuency?
```

This command sets or retrieves the reference frequency value of an externally applied reference to the signal generator. The variable `<val>` is expressed in hertz (Hz–MHz).

The value specified by this command is effective only when you are using an external ARB reference applied to the BASEBAND GEN REF IN rear-panel connector.

To specify external as the ARB reference source type, refer to “:REFerence[:SOURce]” on page 315.

Example

```
:RAD:DMOD:ARB:REF:EXT:FREQ 10MHZ
```

The preceding example sets the external reference to 10 MHz.

***RST** +1.00000000E+007

Range 2.5E5–1E8

Key Entry **Reference Freq**

:REFerence[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:REFerence[:SOURce] INTernal|EXTernal
[:SOURce]:RADio:DMODulation:ARB:REFerence[:SOURce]?
```

This command selects either an internal or external reference for the waveform clock.

If the EXTernal choice is selected, the external frequency value *must* be entered and the signal must be applied to the BASEBAND GEN REF IN rear-panel connector.

Refer to “:REFerence:EXTernal:FREQuency” on page 315 to enter the external reference frequency.

Example

```
:RAD:DMOD:ARB:REF INT
```

The preceding example sets an internal clock reference.

***RST**	INT
Key Entry	**ARB Reference Ext Int**

:RETRigger

Supported	E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:RETRigger ON|OFF|IMMediate
[:SOURce]:RADio:DMODulation:ARB:RETRigger?
```

This commands selects the waveform's response to a trigger signal while using the single trigger mode.

When the PSG receives multiple trigger occurrences when only one is required, the signal generator uses the first trigger and ignores the rest. For more information on triggering and to select the single trigger mode, see ':TRIGger:TYPE' on page 322.

The following list describes the waveform's response to each of the command choices:

ON	The waveform waits for a trigger before play begins and accepts a subsequent trigger during playback. If there is a subsequent trigger during playback, the waveform completes its current playback and then plays one more time. If there is no subsequent trigger, the waveform plays once and stops until it receives another trigger.
OFF	The waveform waits for a trigger before play begins and ignores triggers during playback. To restart the waveform, you must send a trigger after the playback completes.
IMMediate	The waveform waits for a trigger before play begins and accepts a subsequent trigger during playback. Upon receipt of the subsequent trigger, the waveform immediately resets and begins playing from the beginning of the file. For a waveform sequence, this means to the beginning of the first segment in the sequence.

Example

```
:RAD:DMOD:ARB:RETR ON
```

The preceding example selects the ON mode for the single mode trigger.

***RST**	ON
Key Entry	**On** **Off** **Immediate**

:SCLock:RATE

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:SCLock:RATE <sample_clock_rate>
[:SOURce]:RADio:DMODulation:ARB:SCLock:RATE?
```

This command sets the sample clock rate in hertz. The modulation format should be active before executing this command. If this command is executed before the modulation format is active, the entered value will be overridden by a calculated factory default value. Refer to ‘[:STATe]’ on page 329 to activate the modulation format.

Example

```
:RAD:DMOD:ARB:SCL:RATE 50E6
```

The preceding example sets the sample clock rate to 50 MHz.

***RST**	+1.00000000E+008
Range	1–1E8
Key Entry	**ARB Sample Clock**

:SETup

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:SETup GSM|NADC|PDC|PHS|DECT|AC4Fm|
ACQPsk|CDPD|PWT|EDGE|TETRa|MCARrier|"<file_name>"
[:SOURce]:RADio:DMODulation:ARB:SETup?
```

This command selects the digital modulation format type. For information on the file name syntax, see “File Name Variables” on page 12.

Example

```
:RAD:DMOD:ARB:SET CDPD
```

The preceding example selects cellular digital packet data (CDPD) as the modulation format.

***RST**	NADC
Key Entry	**GSM** **NADC** **PDC** **PHS** **DECT** **APCO 25 w/ C4FM** **APCO w/ CQPSK** **CDPD** **PWT** **EDGE** **TETRA** **Multicarrier Off On** **Select File**

:SETup:MCARrier

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:SETup:MCARrier (GSM|NADC|PDC|PHS|DECT|
AC4Fm|ACQPsk|CDPD|PWT|EDGE|TETRa,<num_carriers>,<freq_spacing>)|
"<file_name>"
[:SOURce]:RADio:DMODulation:ARB:SETup:MCARrier?
```

This command builds a table with the specified number of carriers and frequency spacing or retrieves the setup stored in the specified user file. The query returns the carrier type, number of carriers, and frequency spacing. The output format is as follows:

`<carrier_type>,<num_carriers>,<freq_spacing>`

If a specific file is loaded and then queried, only the file name is returned. For information on the file name syntax, see “File Name Variables” on page 12. To store a multicarrier setup refer to ‘:SETup:MCARrier:STORe’ on page 319.

The variable `<freq_spacing>` is expressed in hertz (kHz–MHz).

Example

`:RAD:DMOD:ARB:SET:MCAR` NADC, 2, 10MHZ

`:RAD:DMOD:ARB:SET:MCAR "<file_name>"`

The preceding examples show the syntax used to select a North American Digital Cellular (NADC) modulation format with two carriers and 10 MHz frequency spacing and the syntax for selecting an existing multicarrier file.

***RST** *Carrier*: NADC
<*num carriers*>:2
<*freq spacing*>: +1.0000000000000E+06

Range <*num carriers*>:2–100
<*freq spacing*>: 2 ÷ (<num carriers> – 1) × 80 MHz

Key Entry **GSM** **NADC** **PDC** **PHS** **DECT** **APCO 25 w/C4FM** **APCO w/CQPSK** **CDPD** **PWT** **EDGE** **TETRA** **# of Carriers** **Freq Spacing** **Custom Digital Mod State**

:SETup:MCARrier:PHASe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:SETup:MCARrier:PHASe FIXed|RANDom
[:SOURce]:RADio:DMODulation:ARB:SETup:MCARrier:PHASe?
```

This command sets the phase difference between carriers for multicarrier digital modulation.

FIXed This choice sets the phase of all carriers to 0.

RANDom This choice sets random phase values for all of the carriers.

Example

```
:RAD:DMOD:ARB:SET:MCAR:PHAS RAND
```

The preceding example sets the phase difference between carriers to a random value.

***RST** FIX

Key Entry **Carrier Phases Fixed Random**

:SETup:MCARrier:STORe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:SETup:MCARrier:STORe "<file_name>"
```

This command stores the current multicarrier setup information.

The stored file contains information that includes the digital modulation format, number of carriers, frequency spacing, and power settings for the multicarrier setup.

The setting enabled by this command is not affected by signal generator power-on, preset, or *RST. For information on the file name syntax, see “File Name Variables” on page 12.

Example

```
:RAD:DMOD:ARB:SET:MCAR:STOR "NADC_Data"
```

The preceding example saves the multicarrier setup information to a file called `NADC_Data` and stores the file in the catalog of MDMOD files.

***RST** N/A

Key Entry **Load / Store**

:SETup:MCARrier:TABLe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:SETup:MCARrier:TABLe INIT|APPend
<carrier_num>,GSM|NADC|PDC|PHS|DECT|AC4Fm|ACQPsk|CDPD|PWT|EDGE|TETRa|
"<file_name>",<freq_offset>,<power>
[:SOURce]:RADio:DMODulation:ARB:SETup:MCARrier:TABLe? <carrier_num>
```

This command modifies the parameters of one of the available multicarrier digital modulation formats.

The variable `<freq_offset>` is expressed in units of hertz (kHz to MHz).

The variable `<power>` is expressed in units of decibels (dB).

The carrier type, carrier name, frequency offset, and power level are returned when a query is initiated. The output format is as follows:

`<carrier_type>,<carrier_name>,<freq_offset>,<power>`

`INIT` This choice clears the current information and creates a new one-row table, allowing for further definition using additional parameters.

`APPend` This choice adds rows to an existing table.

`<carrier_num>` This variable specifies the number of the carriers in the multicarrier table that will be modified. The value of the variable <carrier_num> must be specified prior to selecting the digital modulation format.

For information on the file name syntax, see “File Name Variables” on page 12. To store a multicarrier setup refer to ‘:SETup:MCARrier:STORe’ on page 319.

When a query is initiated, carrier type, frequency offset, and power level are returned in the following format: `<carrier_type>,<freq_offset>,<power>`

***RST** *carrier type*: NADC

<freq_offset>: 5.00000000E+004

<power>: +0.00000000E+000

Range *<freq_offset>*: –1E5 to 1E6
<power>: –40 to 0

Key Entry **Initialize Table** **Insert Row** **GSM** **NADC** **PDC** **PHS** **DECT**

APCO 25 w/C4FM **APCO w/CQPSK** **CDPD** **PWT** **EDGE** **TETRA**

Custom Digital Mod State

:SETup:MCARrier:TABLe:NCARriers

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:SETup:MCARrier:TABLe:NCARriers?
```

This query returns the number of carriers in the current multicarrier setup.

***RST** +2

Range 1–100

Key Entry **# of Carriers**

:SETup:STORe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:SETup:STORe "<file_name>"
```

This command stores the current custom digital modulation state using the `"<file_name>"` file name.

The saved file contains information that includes the modulation type, filter and symbol rate for the custom modulation setup.

For information on the file name syntax, see “File Name Variables” on page 12.

Example

```
:RAD:DMOD:ARB:SET:STOR "Setup_Data"
```

The preceding example saves the modulation format setup to a file named `Setup_Data` and stores the file in the catalog of DMOD files.

***RST** N/A

Range N/A

Key Entry **Store Custom Dig Mod State**

:SRATe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:SRATe <val>
[:SOURce]:RADio:DMODulation:ARB:SRATe?
```

This command sets the transmission symbol rate. The variable `<val>` is expressed in bits per second (bps–Mbps) and the maximum range value is dependent upon the source of data (internal or external), the modulation type, and filter.

When user-defined filters are selected using the command in section ":FILTer" on page 305, the upper bit rate will be restricted using the following criteria:

- FIR filter length > 32 symbols: upper limit is 12.5 Msps
- FIR filter length > 16 symbols: upper limit is 25 Msps

When internal FIR filters are used, these limit restrictions always apply. For higher symbol rates, the FIR filter length will be truncated as follows:

- Above 12.5 Msps, the FIR length is truncated to 32 symbols
- Above 25 Msps, the FIR length is truncated to 16 symbols

  This impacts the relative timing of the modulated data, as well as the actual filter response.

To change the modulation type, refer to ":MODulation[:TYPE]" on page 314.

Example

```
:RAD:DMOD:ARB:SRAT 10KSPS
```

The preceding example sets the symbol rate to 10K symbols per second.

***RST**	+2.43000000E+004
Range	1 ksps–50 Msps
Key Entry	**Symbol Rate**

:TRIGger:TYPE

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:TRIGger:TYPE CONTinuous|SINGle|GATE
[:SOURce]:RADio:DMODulation:ARB:TRIGger:TYPE?
```

This command sets the trigger mode (type) that controls the waveform's playback.

Triggers control the playback by telling the PSG when to play the modulating signal (waveform). Depending on the trigger settings for the PSG, the waveform playback can occur once, continuously, or the PSG may start and stop playing the waveform repeatedly (GATE mode).

A trigger signal comprises both positive and negative signal transitions (states), which are also called high and low periods. You can configure the PSG to trigger on either state of the trigger signal. It is common to have multiple triggers, also referred to as trigger occurrences or events, occur when the signal generator requires only a single trigger. In this situation, the PSG recognizes the first trigger and ignores the rest.

When you select a trigger mode, you may lose the signal (carrier plus modulating) from the RF output until you trigger the waveform. This is because the PSG sets the I and Q signals to zero volts prior to the first trigger event, which suppresses the carrier. After the first trigger event, the waveform's final I and Q levels determine whether you will see the carrier signal or not (zero = no carrier, other values = carrier visible). At the end of most files, the final I and Q points are set to a value other than zero.

There are four parts to configuring the trigger:

- Choosing the trigger type, which controls the waveform's transmission.
- Setting the waveform's response to triggers:
  - CONTinuous, see ':TRIGger:TYPE:CONTinuous[:TYPE]' on page 324
  - SINGle, see ':RETRigger' on page 316
  - GATE, selecting the mode also sets the response
- Selecting the trigger source (see ':TRIGger[:SOURce]' on page 325), which determines how the PSG receives its trigger signal, internally or externally. The GATE choice requires an external trigger.
- Setting the trigger polarity when using an external source:
  - CONTinuous and SINGle see ':TRIGger[:SOURce]:EXTernal:SLOPe' on page 328
  - GATE, see ":TRIGger:TYPE:GATE:ACTive" on page 324

For more information on triggering, see the *PSG User's Guide*.

The following list describes the trigger type command choices:

CONTinuous	Upon triggering, the waveform repeats continuously.
SINGle	Upon triggering, the waveform segment or sequence plays once.
GATE	An external trigger signal repeatedly starts and stops the waveform's playback (transmission). The time duration for playback depends on the duty period of the trigger signal and the gate polarity selection (see ":TRIGger:TYPE:GATE:ACTive" on page 324). The waveform plays during the inactive state and stops during the active polarity selection state. The active state can be set high or low. The gate mode works only with an external trigger source.

NOTE The ARB gating behavior described above is opposite to the gating behavior for real-time custom mode.

Example

```
:RAD:DMOD:ARB:TRIG:TYPE GATE
```

The preceding example selects the gate trigger mode.

***RST** CONT

Key Entry **Continuous** **Single** **Gated**

:TRIGger:TYPE:CONTinuous[:TYPE]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:TRIGger:TYPE:CONTinuous[:TYPE] FREE|
TRIGger|RESet
[:SOURce]:RADio:DMODulation:ARB:TRIGger:TYPE:CONTinuous[:TYPE]?
```

This commands selects the waveform's response to a trigger signal while using the continuous trigger mode.

For more information on triggering and to select the continuous trigger mode, see ":TRIGger:TYPE" on page 322.

The following list describes the waveform's response to each of the command choices:

FREE Turning the ARB format on immediately triggers the waveform. The waveform repeats until you turn the format off, select another trigger, or choose another waveform file.

TRIGger The waveform waits for a trigger before play begins. When the waveform receives the trigger, it plays continuously until you turn the format off, select another trigger, or choose another waveform file.

RESet The waveform waits for a trigger before play begins. When the waveform receives the trigger, it plays continuously. Subsequent triggers reset the waveform to the beginning. For a waveform sequence, this means to the beginning of the first segment in the sequence.

Example

```
:RAD:DMOD:ARB:TRIG:TYPE:CONT FREE
```

The preceding example selects the continuous trigger free mode.

***RST** FREE

Key Entry **Free Run** **Trigger & Run** **Reset & Run**

:TRIGger:TYPE:GATE:ACTive

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:TRIGger:TYPE:GATE:ACTive LOW|HIGH
[:SOURce]:RADio:DMODulation:ARB:TRIGger:TYPE:GATE:ACTive?
```

This command selects the active state (gate polarity) of the gate while using the gating trigger mode.

The LOW and HIGH selections correspond to the low and high states of an external trigger signal. For example, when you select HIGH, the active state occurs during the high of the trigger signal. When the active state occurs, the PSG stops the waveform playback at the last played sample point, then restarts the playback at the next sample point when the inactive state occurs. For more information on triggering and to select gating as the trigger mode, see ":TRIGger:TYPE" on page 322.

The following list describes the PSG's gating behavior for the polarity selections:

LOW — The waveform playback stops when the trigger signal goes low (active state) and restarts when the trigger signal goes high (inactive state).

HIGH — The waveform playback stops when the trigger signal goes high (active state) and restarts when the trigger signal goes low (inactive state).

Example

```
:RAD:DMOD:ARB:TRIG:TYPE:GATE:ACT HIGH
```

The preceding example sets the active gate state to high.

***RST** HIGH

Key Entry **Gate Active Low High**

:TRIGger[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce] KEY|EXT|BUS
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce]?
```

This command sets the trigger source.

For more information on triggering, see ":TRIGger:TYPE" on page 322. The following list describes the command choices:

KEY — This choice enables manual triggering by pressing the front-panel **Trigger** hardkey.

EXT — An externally applied signal triggers the waveform. This is the only choice that works with gating. The following conditions affect an external trigger:

- The input connector selected for the trigger signal. You have a choice between the rear-panel PATTERN TRIG IN connector or the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector. To make the connector selection, see ':TRIGger[:SOURce]:EXTernal[:SOURce]' on page 326.

  For more information on the connectors and on connecting the cables, see the *PSG User's Guide*.

- The trigger signal polarity:
  - gating mode, see ':TRIGger:TYPE:GATE:ACTive' on page 324
  - continuous and single modes, see ':TRIGger[:SOURce]:EXTernal:SLOPe' on page 328
- The time delay between when the PSG receives a trigger and when the waveform responds to the trigger. There are two parts to setting the delay:
  - setting the amount of delay, see ':TRIGger[SOURce]:EXTernal:DELay' on page 327
  - turning the delay on, see ':TRIGger[:SOURce]:EXTernal:DELay:STATe' on page 328

BUS — This choice enables triggering over the GPIB or LAN using the `*TRG` or `GET` commands or the AUXILIARY INTERFACE (RS-232) using the `*TRG` command.

Example

```
:RAD:DMOD:ARB:TRIG EXT
```

The preceding example sets the trigger source to external triggering mode.

***RST** EXT

Key Entry **Trigger Key** **Ext** **Bus**

:TRIGger[:SOURce]:EXTernal[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce]:EXTernal[:SOURce] EPT1|
EPT2|EPTRIGGER1|EPTRIGGER2
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce]:EXTernal[:SOURce]?
```

This command selects which PATTERN TRIG IN connection the PSG uses to accept an externally applied trigger signal when external is the trigger source selection.

For more information on configuring an external trigger source and to select external as the trigger source, see ":TRIGger[:SOURce]" on page 325. For more information on the rear-panel connectors, see the *PSG User's Guide*.

The following list describes the command choices:

EPT1 — This choice is synonymous with EPTRIGGER1 and selects the PATTERN TRIG IN rear-panel connector.

EPT2 — This choice is synonymous with EPTRIGGER2 and selects the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector.

EPTRIGGER1	This choice is synonymous with EPT1 and selects the PATTERN TRIG IN rear-panel connector.
EPTRIGGER2	This choice is synonymous with EPT2 and selects the PATT TRIG IN 2 pin on the rear-panel AUXILIARY I/O connector.

Example

```
:RAD:DMOD:ARB:TRIG:EXT EPT1
```

The preceding example sets the trigger source to the PATTERN TRIG IN rear-panel connector.

***RST** EPT1

Key Entry **Patt Trig In 1** **Patt Trig In 2**

:TRIGger[SOURce]:EXTernal:DELay

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce]:EXTernal:DELay <val>
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce]:EXTernal:DELay?
```

This command sets the amount of time to delay the PSG’s response to an external trigger.

The delay is a path (time) delay between when the PSG receives the trigger and when it responds to the trigger. For example, configuring a trigger delay of two seconds, causes the PSG to wait two seconds after receipt of the trigger before the PSG plays the waveform.

The delay does not occur until you turn it on (see “:TRIGger[:SOURce]:EXTernal:DELay:STATe”). You can set the delay value either before or after turning it on.

For more information on configuring an external trigger source and to select external as the trigger source, see “:TRIGger[:SOURce]” on page 325.

The unit of measurement for the variable <val> is in seconds (nsec–sec).

Example

```
:RAD:DMOD:ARB:TRIG:EXT:DEL 200MS
```

The preceding example sets the delay for an external trigger to .2 seconds.

***RST** +1.00000000E−003

Range 1E−8 to 4E1

Key Entry **Ext Delay Time**

:TRIGger[:SOURce]:EXTernal:DELay:STATe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce]:EXTernal:DELay:
STATe ON|OFF|1|0
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce]:EXTernal:DELay:STATe?
```

This command turns the trigger delay on or off when using an external trigger source.

For setting the delay time, see ":TRIGger[SOURce]:EXTernal:DELay" , and for more information on configuring an external source, see":TRIGger[:SOURce]" on page 325.

Example

```
:RAD:DMOD:ARB:TRIG:EXT:DEL 1
```

The preceding example sets the delay active for an external trigger.

***RST** 0

Key Entry **Ext Delay Off On**

:TRIGger[:SOURce]:EXTernal:SLOPe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce]:EXTernal:
SLOPe POSitive|NEGative
[:SOURce]:RADio:DMODulation:ARB:TRIGger[:SOURce]:EXTernal:SLOPe?
```

This command sets the polarity for an external trigger signal while using the continuous, single triggering mode. To set the polarity for gating, see ':TRIGger:TYPE:GATE:ACTive' on page 324.

The POSitive and NEGative selections correspond to the high (positive) and low (negative) states of the external trigger signal. For example, when you select POSitive, the waveform responds (plays) during the high state of the trigger signal. When the PSG receives multiple trigger occurrences when only one is required, the signal generator uses the first trigger and ignores the rest.

For more information on configuring an external trigger source and to select external as the trigger source, see ':TRIGger[:SOURce]' on page 325.

Example

```
:RAD:DMOD:ARB:TRIG:EXT:SLOPE POS
```

The preceding example sets the polarity of the active triggering state to positive.

***RST** NEG

Key Entry **Ext Polarity Neg Pos**

[:STATe]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:DMODulation:ARB[:STATe] ON|OFF|1|0
[:SOURce]:RADio:DMODulation:ARB[:STATe]?
```

This command enables or disables the digital modulation.

`ON` (`1`) This choice sets up the internal hardware to generate the currently selected digital modulation format. When ON is selected, the I/Q state is activated and the I/Q source is set to internal.

`OFF` (`0`) This choice disables the digital modulation capability.

Example

```
:RAD:DMOD:ARB ON
```

The preceding example turns on the arbitrary waveform generator.

***RST** 0

Key Entry **Digital Modulation Off On**

Multitone Subsystem–Option 601or 602 ([:SOURce]:RADio:MTONe:ARB)

Creating a Multitone Waveform

Use the following steps to create a multitone waveform:

1. Initialize the phase for the multitone waveform (“:SETup:TABLe:PHASe:INITialize” on page 341).
2. Assign the frequency spacing between the tones (“:SETup:TABLe:FSPacing” on page 340).
3. Define the number of tones within the waveform (“:SETup:TABLe:NTONes” on page 341).
4. Modify the power level, phase, and state of any individual tones (“:ROW” on page 343).

:HEADer:CLEar

Supported E8267D with Option 601or 602

`[:SOURce]:RADio:MTONe:ARB:HEADer:CLEar`

This command clears the header information from the header file used by this modulation format. Header information consists of signal generator settings and marker routings associated with the waveform file. Refer to the *PSG User's Guide* for information on header files.

For this command to function, the multitone mode must be on. To turn multitone on, see ‘[:STATe]’ on page 343.

***RST** N/A

Key Entry **Clear Header**

:HEADer:SAVE

Supported E8267D with Option 601or 602

`[:SOURce]:RADio:MTONe:ARB:HEADer:SAVE`

This command saves the header information to the header file used by this modulation format. Header information consists of signal generator settings and marker routings associated with the waveform file. Refer to the *PSG User's Guide* for information on header files.

For this command to function, multitone must be on. To turn multitone on, see ‘[:STATe]’ on page 343.

***RST** N/A

Key Entry **Save Setup To Header**

:IQ:EXTernal:FILTer

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:IQ:EXTernal:FILTer 40e6|THRough
[:SOURce]:RADio:MTONe:ARB:IQ:EXTernal:FILTer?
```

This command selects the filter or through path for I/Q signals routed to the rear-panel I and Q outputs. Selecting a filter using this command will automatically set ‘:IQ:EXTernal:FILTer:AUTO’ on page 331 to OFF(0) mode.

40e6 This choice applies a 40 MHz baseband filter.

THRough This choice bypasses filtering.

Example

```
:RAD:MTON:ARB:IQ:EXT:FILT 40E6
```

The preceding example selects a 40 MHz filter for the I/Q rear–panel signal path.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:EXTernal:FILTer:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:IQ:EXTernal:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:MTONe:ARB:IQ:EXTernal:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals routed to the rear-panel I/Q outputs. The AUTO feature allows the signal generator to select the filter or through path for the signal.

`ON(1)` This choice automatically selects the 40 MHz filter optimized for current signal generator settings.

`OFF(0)` This choice disables the auto feature and allows you to select the 40 MHz filter or a through path. Refer to “:IQ:EXTernal:FILTer” on page 304 for selecting a filter or through path.

Example

```
:RAD:MTON:ARB:IQ:EXT:FILT:AUTO ON
```

The preceding example sets output I/Q filtering to automatic.

***RST** 1

Key Entry **I/Q Output Filter Manual Auto**

:IQ:MODulation:ATTen

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:IQ:MODulation:ATTen <val>
[:SOURce]:RADio:MTONe:ARB:IQ:MODulation:ATTen?
```

This command sets the attenuation level of the I/Q signals being modulated through the signal generator RF path. The variable `<val>` is expressed in decibels (dB).

Example

```
:RAD:MTON:ARB:IQ:MOD:ATT 20
```

The preceding example sets the modulator attenuator level to 20dB.

***RST** +2.00000000E+000

Range 0–40 (.01dB resolution)

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:ATTen:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:IQ:MODulation:ATTen:AUTO ON|OFF|1|0
[:SOURce]:RADio:MTONe:ARB:IQ:MODulation:ATTen:AUTO?
```

This command enables or disables the modulator attenuator auto mode. The AUTO mode allows the signal generator to select the best attenuator level for the current settings. The auto mode will be switched to manual if the signal generator receives an `AUTO OFF` or `AUTO 0` command.

`ON` (`1`) This choice enables the attenuation auto mode which optimizes the modulator attenuation for the current conditions.

`OFF` (`0`) This choice holds the attenuator at its current setting or at a selected value. Refer to ‘:IQ:MODulation:ATTen’ on page 332 for setting the attenuation value.

Example

```
:RAD:MTON:ARB:IQ:MOD:ATT:AUTO OFF
```

The preceding example sets the attenuator in manual mode.

***RST** 1

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:FILTer

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:IQ:MODulation:FILTer 40e6|THRough
[:SOURce]:RADio:MTONe:ARB:IQ:MODulation:FILTer?
```

This command enables you to select a filter or through path for I/Q signals modulated onto the RF carrier. Selecting a filter using this command will automatically set ‘:IQ:MODulation:FILTer:AUTO’ on page 333 to OFF(0) mode.

40E6 This choice applies a 40 MHz baseband filter to the I/Q signals.

THRough This choice bypasses filtering.

Example

```
:RAD:MTON:ARB:IQ:MOD:FILT THR
```

The preceding example selects a through path for I/Q signals routed to the rear–panel outputs.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:MODulation:FILTer:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:IQ:MODulation:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:MTONe:ARB:IQ:MODulation:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals modulated onto the RF carrier.

`ON` (`1`) This choice will automatically select the 40 MHz filter optimized for the current signal generator setting.

`OFF` (`0`) This choice disables the automatic filter selection and allows you to select the 40 MHz filter or the through path. Refer to “:IQ:MODulation:FILTer” on page 277 for selecting a filter or through path.

Example

```
:RAD:MTON:ARB:IQ:MOD:FILT:AUTO OFF
```

The preceding example sets the automatic filter selection off.

:MDEStination:ALCHold

Supported E8267D with Option 601or 602

CAUTION Incorrect ALC sampling can create a sudden unleveled condition that may create a spike in the RF output potentially damaging a DUT or connected instrument. Ensure that you set markers to let the ALC sample over an amplitude that accounts for the high power levels within the signal.

```
[:SOURce]:RADio:MTONe:ARB:MDEStination:ALCHold NONE|M1|M2|M3|M4
[:SOURce]:RADio:MTONe:ARB:MDEStination:ALCHold?
```

This command enables or disables the marker ALC hold function for the selected marker.

Use the ALC hold function when you have a waveform signal that incorporates idle periods, or when the increased dynamic range encountered with RF blanking is not desired. The ALC circuitry responds to the marker signal during the marker pulse (marker signal high), averaging the modulated signal level during this period.

The ALC hold function operates during the low periods of the marker signal. The marker polarity determines when the marker signal is high. For a positive polarity, this is during the marker points. For a negative polarity, this is when there are no marker points. To set a marker's polarity, see ':MPOLarity:MARKer1|2|3|4' on page 337. For more information on markers, see ':MARKer:[SET]' on page 281.

NOTE Do not use the ALC hold for more than 100 ms, because it can affect the waveform's output amplitude.

The marker signal has a minimum of a two-sample delay in its response relative to the waveform signal response. To compensate for the marker signal delay, offset marker points from the waveform sample point at which you want the ALC sampling to begin.

The ALC hold setting is part of the file header information, so saving the setting to the file header saves the current marker routing for the waveform file.

NOTE A waveform file that has unspecified settings in the file header uses the previous waveform's routing settings.

For more information on the marker ALC hold function, see the *PSG User's Guide*. To configure marker points, refer to the following sections located in the Dual ARB subsystem:

- For clearing a single marker point or a range of marker points, see':MARKer:CLEar' on page 278.

- For clearing all marker points, see ‘:MARKer:CLEar:ALL’ on page 279.
- For shifting marker points, see ‘:MARKer:ROTate’ on page 280.
- For setting marker points, see ‘:MARKer:[SET]’ on page 281.

NONE	This terminates the marker ALC hold function.
M1–M4	These are the marker choices. The ALC hold feature uses only one marker at a time.

Example

```
:RAD:MTON:ARB:MDES:ALCH M1
```

The preceding example routes marker one to the ALC hold function.

***RST** NONE

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

:MDEStination:PULSe

Supported E8267D with Option 601or 602

CAUTION The pulse function incorporates ALC hold. Incorrect ALC sampling can create a sudden unleveled condition that may create a spike in the RF output potentially damaging a DUT or connected instrument. Ensure that you set markers to let the ALC sample over an amplitude that accounts for the high power levels within the signal.

```
[:SOURce]:RADio:MTONe:ARB:MDEStination:PULSe NONE|M1|M2|M3|M4
[:SOURce]:RADio:MTONe:ARB:MDEStination:PULSe?
```

This command disables the marker RF blanking/pulse function, or it enables the marker RF blanking/pulse function for the selected marker.

This function automatically incorporates the ALC hold function, so there is no need to select both functions for the same marker.

NOTE Do not use ALC hold for more than 100 ms, because it can affect the waveform’s output amplitude.

The signal generator blanks the RF output when the marker signal goes low. The marker polarity determines when the marker signal is low. For a positive polarity, this is during the marker points. For a negative polarity, this is when there are no marker points. To set a marker’s polarity, see ‘:MPOLarity:MARKer1|2|3|4’ on page 337. For more information on setting markers, see

':MARKer:[SET]' on page 281.

NOTE Set marker points prior to using this function. Enabling this function without setting marker points may create a continuous low or high marker signal, depending on the marker polarity. This creates the condition where there is either no RF output or a continuous RF output.

To configure marker points, refer to the following sections located in the Dual ARB subsystem:

- For clearing a single marker point or a range of marker points, see':MARKer:CLEar' on page 278.
- For clearing all marker points, see ':MARKer:CLEar:ALL' on page 279.
- For shifting marker points, see ':MARKer:ROTate' on page 280.
- For setting marker points, see ':MARKer:[SET]' on page 281.

The marker signal has a minimum of a two-sample delay in its response relative to the waveform signal response. To compensate for the marker signal delay, offset marker points from the waveform sample point at which you want the RF blanking to begin.

The RF blanking setting is part of the file header information, so saving the setting to the file header saves the current marker routing for the waveform file.

NOTE A waveform file that has unspecified settings in the file header uses the previous waveform's routing settings. This could create the situation where there is no RF output signal, because the previous waveform used RF blanking

For more information on the marker RF blanking function, see the *PSG User's Guide*.

NONE	This terminates the marker RF blanking/pulse function.
M1–M4	These are the marker choices. The RF blanking/pulse feature uses only one marker at a time.

Example

```
:RAD:MTON:ARB:MDES:PULSE M1
```

The preceding example routes marker one to the Pulse/RF Blanking function.

***RST** NONE

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

:MPOLarity:MARKer1|2|3|4

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:MPOLarity:MARKer1|2|3|4 NEGative|POSitive
[:SOURce]:RADio:MTONe:ARB:MPOLarity:MARKer1|2|3|4?
```

This command sets the polarity for the selected marker.

For a positive marker polarity, the marker signal is high during the marker points. For a negative marker polarity, the marker signal is high during the period of no marker points. To configure marker points, refer to the following sections located in the Dual ARB subsystem:

- For clearing a single marker point or a range of marker points, see‘:MARKer:CLEar’ on page 278.
- For clearing all marker points, see ‘:MARKer:CLEar:ALL’ on page 279.
- For shifting marker points, see ‘:MARKer:ROTate’ on page 280.
- For information on markers and setting marker points, see ‘:MARKer:[SET]’ on page 281.

Example

```
:RAD:MTON:ARB:MPOL:MARK1 NEG
```

The preceding example sets the polarity for marker one to negative.

***RST** POS

Key Entry **Marker 1 Polarity Neg Pos** **Marker 2 Polarity Neg Pos** **Marker 3 Polarity Neg Pos** **Marker 4 Polarity Neg Pos**

:REFerence:EXTernal:FREQuency

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:REFerence:EXTernal:FREQuency <val>
[:SOURce]:RADio:MTONe:ARB:REFerence:EXTernal:FREQuency?
```

This command allows you to enter the frequency of an external reference. The variable `<val>` is expressed in hertz (Hz–MHz). The value specified by this command is effective only when you are using an external ARB reference applied to the BASEBAND GEN REF IN rear-panel connector. To specify external as the ARB reference source type, refer to “:REFerence[:SOURce]” on page 338.

Example

```
:RAD:MTON:ARB:REF:EXT:FREQ 500KHZ
```

The preceding example sets the external reference to .5 megahertz.

***RST** +1.00000000E+007

Range 2.5E5–1E8

Key Entry **Reference Freq**

:REFerence[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:REFerence[:SOURce] INTernal|EXTernal
[:SOURce]:RADio:MTONe:ARB:REFerence[:SOURce]?
```

This command selects either an internal or external reference for the waveform clock. If EXTernal is selected, the external frequency *value must* be entered and the clock signal must be applied to the BASEBAND GEN REF IN rear-panel connector. See “:REFerence:EXTernal:FREQuency” on page 337 to enter the external reference frequency.

Example

```
:RAD:MTON:ARB:REF EXT
```

The preceding example sets an external reference as the waveform clock.

***RST** INT

Key Entry **ARB Reference Ext Int**

:SCLock:RATE

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:SCLock:RATE <sample_clock_rate>
[:SOURce]:RADio:MTONE:ARB:SCLock:RATE?
```

This command sets the ARB sample clock rate.

The multitone generator should be on before executing this command. If this command is executed before the multitone generator is active, the entered value will be overridden by a calculated factory default value. Refer to ‘[:STATe]’ on page 329 to activate the modulation format.

Example

```
:RAD:MTON:ARB:SCL:RATE 1E6
```

The preceding example sets the sample clock rate to 1 megahertz.

***RST** +1.00000000E+006

Range 1–1E8

Key Entry **ARB Sample Clock**

:SETup

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:SETup "<file_name>"
[:SOURce]:RADio:MTONe:ARB:SETup?
```

This command retrieves a multitone waveform file from the signal generator's MTONE directory. The directory path is implied in the command and does not need to be specified. After the waveform file is loaded into memory you must send the command to turn on the Multitone generator. For information on the file name syntax, see "File Name Variables" on page 12.

Example

```
:RAD:MTON:ARB:SET "Multi_Setup"
```

The preceding example loads the `Multi_Setup` waveform file into the signal generator's memory.

Key Entry **Load From Selected File**

:SETup:STORe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:SETup:STORe "<file_name>"
```

This command stores the current multitone waveform setup in the signal generator's MTONE directory using the `"<file_name>"` file name. The directory path is implied in the command and does not need to be specified.

Example

```
:RAD:MTON:ARB:SET:STOR "Multi_Setup1"
```

The preceding example stores the current multitone setup to the `Multi_Setup1` file and stores it in the signal generator's MTONE directory.

Key Entry **Store To File**

:SETup:TABLe

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe <freq_spacing>,<num_tones>,
{<phase>,<state>}
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe?
```

This command creates and configures a multitone waveform. The frequency offset, power, phase, and state value are returned when a query is initiated. The parameter format is as follows:

`<freq_spacing>`	Spacing is limited by the 80 MHz bandwidth of the arbitrary waveform generator and the number of tones desired. No units are specified.
`<num_tones>`	There must be a minimum of two tones and a maximum of 64.
`<phase>`	0-359
`<state>`	An enabled state is +1. A disabled state is 0.

NOTE Frequency offset is related to frequency spacing. Frequency offset between tones equals the frequency spacing.

To set the frequency spacing, refer to ":SETup:TABLe:FSPacing" on page 340. To set the power level for tones refer to ":ROW" on page 343.

Example

```
:RAD:MTON:ARB:SET:TABL 1000000,3,90,1,60,0,45,1
```

The preceding example creates a multitone setup consisting of 3 tones with 1 megahertz tone spacing. The first tone phase is 90 degrees and the state is on. The second tone phase is 60 degrees and the state is off. The third tone phase is 45 degrees and the state is on.

***RST**

Tone	<frequency offset>	<power>	<phase>	<state>
Tone 1	-35000	+0.00000000E+000	+0	+1
Tone 2	-25000	+0.00000000E+000	+0	+1
Tone 3	-15000	+0.00000000E+000	+0	+1
Tone 4	-5000	+0.00000000E+000	+0	+1
Tone 5	+5000	+0.00000000E+000	+0	+1
Tone 6	+15000	+0.00000000E+000	+0	+1
Tone 7	+25000	+0.00000000E+000	+0	+1
Tone 8	+35000	+0.00000000E+000	+0	+1

Range *<freq_spacing>* (2 tones): 1E4–8E7 *<num_tones>*: 2–64

<freq_spacing> (>2 tones): 1E4 to (80 MHz ÷ (num_tones – 1))

<phase>: 0–359

Key Entry **Freq Spacing** **Number Of Tones** **Toggle State**

:SETup:TABLe:FSPacing

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:FSPacing <freq_spacing>
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:FSPacing?
```

This command sets the frequency spacing between tones. The variable <`freq_spacing`> is expressed in hertz (Hz–MHz) and is limited to the 80 megahertz bandwidth of the arbitrary waveform generator.

To set frequency spacing and additional parameters required to create or configure a multitone waveform, refer to ":SETup:TABLe" on page 339. This command is the second step in creating a multitone waveform. Refer to "Creating a Multitone Waveform" on page 330 for all four steps.

Example

```
:RAD:MTON:ARB:SET:TABL:FSP 100KHZ
```

The preceding example sets a 100 kHz frequency spacing between tones.

***RST**	+1.00000000E+004
Range	<*freq_spacing*> (2 tones): 100 Hz –80 MHz <*freq_spacing*> (>2 tones): 1E2 to (80 MHz ÷ (num_tones – 1))
Key Entry	**Freq Spacing**

:SETup:TABLe:NTONes

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:NTONes <num_tones>
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:NTONes?
```

This command defines the number of tones in the multitone waveform. To specify the number of tones and additional parameters required to create or configure a multitone waveform, refer to ":SETup:TABLe" on page 339. This command is the third step in creating a multitone waveform. Refer to "Creating a Multitone Waveform" on page 330 for all four steps.

Example

```
:RAD:MTON:ARB:SET:TABL:NTON 4
```

The preceding example sets four tones in the current multitone table.

***RST**	+8
Range	2–64
Key Entry	**Number Of Tones**

:SETup:TABLe:PHASe:INITialize

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize FIXed|RANDom
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize?
```

This command initializes the phase in the multitone waveform table.

FIXed	This choice sets the phase of all tones to the fixed value of 0 degrees.
RANDom	This choice sets the phase of all tones to random values based on the setting on the random seed generator.

To change the random number generator seed value, refer to ":SETup:TABLe:PHASe:INITialize:SEED" on page 342.

This command is the first step in creating a multitone waveform. Refer to "Creating a Multitone Waveform" on page 330 for all four steps.

Example

```
:RAD:MTON:ARB:SET:TABL:PHAS:INIT RAND
```

The preceding example sets the phase for the tones to a random number.

***RST** FIX

Key Entry **Initialize Phase Fixed Random**

:SETup:TABLe:PHASe:INITialize:SEED

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:SEED FIXed|RANDom
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:PHASe:INITialize:SEED?
```

This command initializes the random number generator seed that is used to generate phase values for the multitone waveform tones.

FIXed	This choice sets the random number generator seed to a fixed value. This selection will generator random and repeatable phase values: the same phase values will be generated with subsequent execution of the command.
RANDom	This choice sets the random number generator seed to a random value. This changes the phase value after each initialization of the phase.

Example

```
:RAD:MTON:ARB:SET:TABL:PHAS:INIT:SEED RAND
```

The preceding example sets the random number generator seed to a random value.

***RST** FIX

Key Entry **Random Seed Fixed Random**

:ROW

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:ROW <row_number>,<power>,
<phase>,<state>
[:SOURce]:RADio:MTONe:ARB:SETup:TABLe:ROW? <row_number>
```

This command modifies the indicated tone (row) of the multitone waveform.

`<row_number>`	The number of rows for this variable is determined by the :SETup:TABLe command.
`<power>`	The power level of the tone defined in the row number. Power levels for all tones must not exceed the power level of the signal generator. The power variable is expressed in decibels (dB)
`<phase>`	The phase of the tone relative to the carrier. The phase variable is expressed in degrees.
`<state>`	The state of the tone in this row can be enabled or disabled.

Frequency offset, power, phase, and state value are returned when a query is initiated. The output format is as follows:
`<frequency_offset>,<power>,<phase>,<state>`

Refer to ":SETup:TABLe" on page 339 for information on how to change the number of rows. This command is the final step in creating a multitone waveform. Refer to "Creating a Multitone Waveform" on page 330 for all four steps.

Example

```
:RAD:MTON:ARB:SET:TABL:ROW 2,-10,40,0
```

The preceding example modifies row number two in the currently selected multitone table. The power is set to –10 dB, the phase is set to 40 degrees, and the state is off.

***RST** *frequency offset*: –3.50000000E+004 <*power*>: +0.00000000E+000 <*phase*>: +0.00000000E+000 <*state*>: 1

Range *frequency offset*: –4E7 to 4E7 <*power*>: –80 to 0 <*phase*>: 0–359 <*state*>: 1

Key Entry **Goto Row** **Toggle State**

[:STATe]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:MTONe:ARB[:STATe] ON|OFF|1|0
```

`[:SOURce]:RADio:MTONe:ARB[:STATe]?`

This command enables or disables the operating state of the multitone waveform generator.

Example

`:RAD:MTON:ARB ON`

The preceding example turns on the multitone generator.

***RST**	0
Key Entry	**Multitone Off On**

Two Tone Subsystem ([:SOURce]:RADio:TTONe:ARB)

:ALIGnment

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:ALIGnment LEFT|CENTer|RIGHt
[:SOURce]:RADio:TTONe:ARB:ALIGnment?
```

This command will align the two tones either left, center or right of the carrier frequency.

Example

```
:RAD:TTON:ARB:ALIG CENT
```

The preceding example aligns each of the two tones equidistant from the carrier frequency.

Key Entry **Alignment Left Cent Right**

:APPly

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:APPLy
```

This command will cause the two-tone waveform to be regenerated using the current settings.

This command has no effect unless the two-tone waveform generator is enabled and a change has been made to the frequency spacing setting.

Key Entry **Apply Settings**

:FSPacing

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:FSPacing <freq_spacing>
[:SOURce]:RADio:TTONe:ARB:FSPacing?
```

This command sets the frequency spacing between the tones.

The variable <freq_spacing> is expressed in hertz (Hz–MHz).

Example

```
:RAD:TTON:ARB:FSP 10MHZ
```

The preceding example sets a 10 megahertz frequency spacing for the two tones.

***RST**	+1.00000000E+004
Range	1E2–8E7
Key Entry	**Freq Separation**

:HEADer:CLEar

Supported	E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:HEADer:CLEar
```

This command clears the header information from the header file used for the two-tone waveform format. Header information consists of signal generator settings and marker routings associated with the waveform file. Refer to the *PSG User's Guide* for information on header files.

For this command to function, two tone must be on. To turn two tone on, see ':STATe' on page 358.

***RST**	N/A
Key Entry	**Clear Header**

:HEADer:SAVE

Supported	E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:HEADer:SAVE
```

This command saves the header information to the header file used for the two-tone waveform format. Header information consists of signal generator settings and marker routings associated with the waveform file. Refer to the *PSG User's Guide* for information on header files.

For this command to function, two tone must be on. To turn two tone on, see '[:STATe]' on page 354.

***RST**	N/A
Key Entry	**Save Setup To Header**

:IQ:EXTernal:FILTer

Supported	E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:IQ:EXTernal:FILTer 40e6|THRough
[:SOURce]:RADio:TTONe:ARB:IQ:EXTernal:FILTer?
```

This command selects the filter or through path for I/Q signals routed to the rear-panel I and Q outputs. Selecting a filter with this command automatically sets ':IQ:EXTernal:FILTer:AUTO' on page 331 to OFF.

40e6	This choice applies a 40 MHz baseband filter.

THRough — This choice bypasses filtering.

Example

```
:RAD:TTON:ARB:IQ:EXT:FILT THR
```

The preceding example sets the through path for I/Q signal.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:EXTernal:FILTer:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:IQ:EXTernal:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:TTONe:ARB:IQ:EXTernal:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals routed to the rear-panel I/Q outputs.

`ON(1)` This choice automatically selects the 40 MHz filter optimized for the current signal generator settings.

`OFF(0)` This choice disables the auto feature and allows you to select the 40 MHz filter or a through path. Refer to “:IQ:EXTernal:FILTer” on page 304 for selecting a filter or through path.

Example

```
:RAD:TTON:ARB:IQ:EXT:FILT:AUTO ON
```

The preceding example enables the automatic filter selection.

***RST** 1

Key Entry **I/Q Output Filter Manual Auto**

:IQ:MODulation:ATTen

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:IQ:MODulation:ATTen <val><unit>
[:SOURce]:RADio:TTONe:ARB:IQ:MODulation:ATTen?
```

This command sets the attenuation level of the I/Q signals being modulated through the signal generator RF path. The variable `<val>` is expressed in decibels (dB).

Example

```
:RAD:TTON:ARB:IQ:MOD:ATT 20
```

The preceding example sets the modulator attenuator to 20 dB.

***RST** +2.00000000E+000

Range 0–40 dB

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:ATTen:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:IQ:MODulation:ATTen:AUTO ON|OFF|1|0
[:SOURce]:RADio:TTONe:ARB:IQ:MODulation:ATTen:AUTO?
```

This command enables or disables the modulator attenuator auto mode. The auto mode will be switched to manual if the signal generator receives an `AUTO OFF` or `AUTO ON` command.

`ON` (`1`) This choice enables the attenuation auto mode which allows the signal generator to select the attenuation level that optimizes performance based on the current conditions.

`OFF` (`0`) This choice holds the attenuator at its current setting or at a selected value. Refer to ‘:IQ:MODulation:ATTen’ on page 332 for setting the attenuation value.

Example

```
:RAD:TTON:ARB:IQ:MOD:ATT:AUTO ON
```

The preceding example enables the attenuator automatic mode.

***RST** 1

Key Entry **Modulator Atten Manual Auto**

:IQ:MODulation:FILTer

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:IQ:MODulation:FILTer 40e6|THRough
[:SOURce]:RADio:TTONe:ARB:IQ:MODulation:FILTer?
```

This command enables you to select a filter or through path for I/Q signals modulated onto the RF carrier. Selecting a filter using this command will automatically set ‘:IQ:MODulation:FILTer:AUTO’ on page 333 to `OFF` (`0`) mode.

40E6 This choice applies a 40 MHz baseband filter to the I/Q signals.

THRough This choice bypasses filtering.

Example

```
:RAD:TTON:ARB:IQ:MOD:FILT 40E6
```

The preceding example selects the 40 MHz filter.

***RST** THR

Key Entry **40.000 MHz** **Through**

:IQ:MODulation:FILTer:AUTO

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:IQ:MODulation:FILTer:AUTO ON|OFF|1|0
[:SOURce]:RADio:TTONe:ARB:IQ:MODulation:FILTer:AUTO?
```

This command enables or disables the automatic filter selection for I/Q signals modulated onto the RF carrier.

`ON` (1) This choice will automatically select the 40 MHz filter optimized for the current signal generator setting.

`OFF` (0) This choice disables the automatic filter selection and allows you to select a digital modulation filter or through path. Refer to ":IQ:MODulation:FILTer" on page 277 for selecting a filter or through path.

Example

```
:RAD:TTON:ARB:IQ:MOD:FILT:AUTO ON
```

The preceding example enables the automatic filter selection for I/Q signals.

***RST** 1

Key Entry **I/Q Mod Filter Manual Auto**

:MDEStination:ALCHold

Supported E8267D with Option 601or 602

CAUTION Incorrect ALC sampling can create a sudden unleveled condition that may create a spike in the RF output potentially damaging a DUT or connected instrument. Ensure that you set markers to let the ALC sample over an amplitude that accounts for the high power levels within the signal.

```
[:SOURce]:RADio:TTONe:ARB:MDEStination:ALCHold NONE|M1|M2|M3|M4
[:SOURce]:RADio:TTONe:ARB:MDEStination:ALCHold?
```

This command disables the marker ALC hold function, or it enables the marker hold function for the selected marker.

Use the ALC hold function when you have a waveform signal that incorporates idle periods, or when the increased dynamic range encountered with RF blanking is not desired. The ALC circuitry responds to the marker signal during the marker pulse (marker signal high), averaging the modulated signal level during this period.

The ALC hold function operates during the low periods of the marker signal. The marker polarity determines when the marker signal is high. For a positive polarity, this is during the marker points. For a negative polarity, this is when there are no marker points. To set a marker's polarity, see ':MPOLarity:MARKer1|2|3|4' on page 352. For more information on markers, see ':MARKer:[SET]' on page 281.

NOTE	Do not use the ALC hold for more than 100 ms, because it can affect the waveform's output amplitude.

The marker signal has a minimum of a two-sample delay in its response relative to the waveform signal response. To compensate for the marker signal delay, offset marker points from the waveform sample point at which you want the ALC sampling to begin.

The ALC hold setting is part of the file header information, so saving the setting to the file header saves the current marker routing for the waveform file.

NOTE	A waveform file that has unspecified settings in the file header uses the previous waveform's routing settings.

For more information on the marker ALC hold function, see the *PSG User's Guide*. To configure marker points, refer to the following sections located in the Dual ARB subsystem:

- For clearing a single marker point or a range of marker points, see':MARKer:CLEar' on page 278.
- For clearing all marker points, see ':MARKer:CLEar:ALL' on page 279.
- For shifting marker points, see ':MARKer:ROTate' on page 280.
- For setting marker points, see ':MARKer:[SET]' on page 281.

NONE	This terminates the marker ALC hold function.
M1–M4	These are the marker choices. The ALC hold feature uses only one marker at a time.

Example

```
:RAD:TTON:ARB:MDES:ALCH M2
```

The preceding example routes marker two to the ALC hold function.

***RST**	NONE
Key Entry	**None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**
Remarks	N/A

:MDEStination:PULSe

Supported E8267D with Option 601or 602

CAUTION The pulse function incorporates ALC hold. Incorrect ALC sampling can create a sudden unleveled condition that may create a spike in the RF output, potentially damaging a DUT or connected instrument. Ensure that you set markers to let the ALC sample over an amplitude that accounts for the high power levels within the signal.

```
[:SOURce]:RADio:TTONe:ARB:MDEStination:PULSe NONE|M1|M2|M3|M4
[:SOURce]:RADio:TTONe:ARB:MDEStination:PULSe?
```

This command disables the marker RF blanking/pulse function, or it enables the marker RF blanking/pulse function for the selected marker.

This function automatically incorporates the ALC hold function, so there is no need to select both functions for the same marker.

NOTE Do not use ALC hold for more than 100 ms, because it can affect the waveform's output amplitude.

The signal generator blanks the RF output when the marker signal goes low. The marker polarity determines when the marker signal is low. For a positive polarity, this is during the marker points. For a negative polarity, this is when there are no marker points. To set a marker's polarity, see ':MPOLarity:MARKer1|2|3|4' on page 352. For more information on markers, see ':MARKer:[SET]' on page 281.

NOTE Set marker points prior to using this function. Enabling this function without setting marker points may create a continuous low or high marker signal, depending on the marker polarity. This creates the condition where there is either no RF output or a continuous RF output.

To configure marker points, refer to the following sections located in the Dual ARB subsystem:

- For clearing a single marker point or a range of marker points, see':MARKer:CLEar' on page 278.
- For clearing all marker points, see ':MARKer:CLEar:ALL' on page 279.

- For shifting marker points, see ‘:MARKer:ROTate’ on page 280.
- For setting marker points, see ‘:MARKer:[SET]’ on page 281.

The marker signal has a minimum of a two-sample delay in its response relative to the waveform signal response. To compensate for the marker signal delay, offset marker points from the waveform sample point at which you want the RF blanking to begin.

The RF blanking setting is part of the file header information, so saving the setting to the file header saves the current marker routing for the waveform file.

NOTE	A waveform file that has unspecified settings in the file header uses the previous waveform’s routing settings. This could create the situation where there is no RF output signal, because the previous waveform used RF blanking

For more information on the marker RF blanking function, see the *PSG User’s Guide*.

NONE	This terminates the marker RF blanking/pulse function.
M1–M4	These are the marker choices. The RF blanking/pulse feature uses only one marker at a time.

Example

```
:RAD:TTON:ARB:MDES:ALCH M3
```

The preceding example routes marker three to the Pulse/RF Blanking function.

***RST** NONE

Key Entry **None** **Marker 1** **Marker 2** **Marker 3** **Marker 4**

:MPOLarity:MARKer1|2|3|4

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:MPOLarity:MARKer1|2|3|4 NEGative|POSitive
[:SOURce]:RADio:TTONe:ARB:MPOLarity:MARKer1|2|3|4?
```

This command sets the polarity for the selected marker.

For a positive marker polarity, the marker signal is high during the marker points. For a negative marker polarity, the marker signal is high during the period of no marker points. To configure marker points, refer to the following sections located in the Dual ARB subsystem:

- For clearing a single marker point or a range of marker points, see‘:MARKer:CLEar’ on page 278.
- For clearing all marker points, see ‘:MARKer:CLEar:ALL’ on page 279.

- For shifting marker points, see ‘:MARKer:ROTate’ on page 280.
- For information on markers and setting marker points, see ‘:MARKer:[SET]’ on page 281.

Example

```
:RAD:TTON:ARB:MPOL:MARK1 POS
```

The preceding example sets the polarity for marker one to positive.

***RST** POS

Key Entry **Marker 1 Polarity Neg Pos** **Marker 2 Polarity Neg Pos** **Marker 3 Polarity Neg Pos** **Marker 4 Polarity Neg Pos**

:REFerence:EXTernal:FREQuency

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:REFerence:EXTernal:FREQuency <val>
[:SOURce]:RADio:TTONe:ARB:REFerence:EXTernal:FREQuency?
```

This command allows you to enter the frequency of the external reference.

The variable `<val>` is expressed in hertz (Hz–MHz).

The value specified by this command is effective only when you are using an external ARB reference applied to the BASEBAND GEN REF IN rear-panel connector.

To specify external as the ARB reference source type, refer to “:REFerence[:SOURce]” on page 288.

Example

```
:RAD:TTON:ARB:REF:EXT:FREQ 1MHZ
```

The preceding example sets the external reference to 1 megahertz.

***RST** +1.00000000E+007

Range 2.5E5–1E8

Key Entry **Reference Freq**

:REFerence[:SOURce]

Supported E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:REFerence[:SOURce] INTernal|EXTernal
[:SOURce]:RADio:TTONe:ARB:REFerence[:SOURce]?
```

This command selects either an internal or external reference for the waveform clock. If EXTernal is selected, the external frequency *value must* be entered and the clock signal must be applied to the BASEBAND GEN REF IN rear-panel connector. See ":REFerence:EXTernal:FREQuency" on page 337 to enter the external reference frequency.

Example

```
:RAD:TTON:ARB:REF EXT
```

The preceding example sets an external reference as the waveform clock.

***RST**	INT
Key Entry	**ARB Reference Ext Int**

:SCLock:RATE

Supported	E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB:SCLock:RATE <sample_clock_rate>
[:SOURce]:RADio:TTONE:ARB:SCLock:RATE?
```

This command sets the ARB sample clock rate.

The multitone generator should be on before executing this command. If this command is executed before the multitone generator is active, the entered value will be overridden by a calculated factory default value. Refer to '[:STATe]' on page 329 to activate the modulation format.

Example

```
:RAD:TTON:ARB:SCL:RATE 1MHZ
```

The preceding example sets the ARB sample clock to 1 MHz.

***RST**	+1.00000000E+008
Range	1–1E8
Key Entry	**ARB Sample Clock**

[:STATe]

Supported	E8267D with Option 601or 602

```
[:SOURce]:RADio:TTONe:ARB[:STATe] ON|OFF|1|0
[:SOURce]:RADio:TTONe:ARB[:STATe]?
```

This command enables or disables the on/off operational state of the two-tone waveform generator function.

Example

```
:RAD:TTON:ARB ON
```

The preceding example turns on the two-tone generator.

***RST** 0

Key Entry **Two Tone Off On**

Wideband Digital Modulation Subsystem ([:SOURce]:WDM)

:IQADjustment:IOFFset

Supported E8267D with Option 015

```
[:SOURce]:WDM:IQADjustment:IOFFset <val><unit>
[:SOURce]:WDM:IQADjustment:IOFFset?
```

This command sets the I channel offset value, as a percent of the full scale. 100% offset is equivalent to 500 mV DC at the input connector.

Example

```
:WDM:IQAD:IOFF 100MV
```

The preceding example sets an offset of 100 mV DC for the I signal.

***RST** +0.00000000E+000

Range –5E1 to +5E1

Key Entry **I Offset**

:IQADjustment:QOFFset

Supported E8267D with Option 015

```
[:SOURce]:WDM:IQADjustment:QOFFset <val><unit>
[:SOURce]:WDM:IQADjustment:QOFFset?
```

This command sets the Q channel offset value, as a percent of the full scale. 100% offset is equivalent to 500 mV DC at the input connector

Example

```
:WDM:IQAD:QOFF 100MV
```

The preceding example sets an offset of 100 mV DC for the Q signal.

***RST** +0.00000000E+000

Range –5E1 to +5E1

Key Entry **Q Offset**

:IQADjustment:QSKew

Supported E8267D with Option 601or 602 and Option 015

```
[:SOURce]:WDM:IQADjustment:QSKew <val>
[:SOURce]:WDM:IQADjustment:QSKew?
```

This command adjusts the phase angle between the I and Q vectors.
The variable `<val>` is expressed in degrees with a minimum resolution of 0.1.

Positive skew increases the angle from 90 degrees while negative skew decreases the angle from 90 degrees. When the quadrature skew is zero, the phase angle is 90 degrees. If the signal generator is operating at frequencies greater than 3.3 GHz, quadrature skew settings greater than ±5 degrees will not be within specifications.

This command is effective only if the state of the I/Q adjustment function is set to ON. Refer to ":IQADjustment[:STATe]" on page 357.

Example

```
:WDM:IQAD:QSK 3.1
```

The preceding example sets the skew value for the Q signal to 3.1 degrees.

***RST** +0.00000000E+000

Range –1E1 to +1E1

Key Entry **Quadrature Skew**

:IQADjustment[:STATe]

Supported E8267D with Option 015

```
[:SOURce]:WDM:IQADjustment[:STATe] ON|OFF|1|0
[:SOURce]:WDM:IQADjustment[:STATe]?
```

This command enables or disables the wideband I/Q adjustments.

Example

```
:WDM:IQAD ON
```

The preceding example enables I/Q adjustments.

***RST** 0

Key Entry **I/Q Adjustments Off On**

:STATe

Supported E8267D with Option 015

```
[:SOURce]:WDM:STATe ON|OFF|1|0
[:SOURce]:WDM:STATe?
```

This command enables or disables the wideband I/Q modulator.
The I/Q modulator is automatically enabled whenever a digital modulation form is turned on and when active, the `I/Q` annunciator appears on the signal generator’s display.

Example

```
:WDM:STAT ON
```

The preceding example enables the wideband I/Q modulator.

***RST** 0

Key Entry **I/Q Off On**

6 Digital Signal Interface Module Commands

This chapter provides SCPI descriptions for commands available with Agilent's N5102A Digital Signal Interface Module. Refer to the *PSG User's Guide*, *PSG Key Reference*, or *N5101A Digital Signal Interface Module Installation Guide* for more information on the N5102A interface module.

Digital Subsystem ([:SOURce])

:DIGital:CLOCk:CPS

Supported E8267D Option 601or 602 with Option 003

```
[:SOURce]:DIGital:CLOCk:CPS 1|2|4
[:SOURce]:DIGital:CLOCk:CPS?
```

This command selects the number of clock cycles per sample. The command is used with parallel or parallel interleaved port configurations. If this command is executed with a serial port configuration or an IF signal type, the parameter value is changed, but it is not used by the interface module until the port configuration is changed to parallel or parallel interleaved, *and* the signal type is changed to IQ.

The query returns the currently set value, regardless of the port configuration, you must query all four states (clocks per sample, port configuration, data direction, and signal type) to know the interface module's current setup

Example

```
:DIG:CLOC:CPS 2
```

The preceding example sets two clock cycles for each sample.

***RST** `1`

Range 1,2,or 4

Key Entry **Clocks Per Sample**

:DIGital:CLOCk:PHASe

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:CLOCk:PHASe <val>
[:SOURce]:DIGital:CLOCk:PHASe?
```

This command sets the phase for the clock relative to the leading edge transition of the data. At 0 degrees the clock and leading edge of the data signal are aligned. Any phase value between 0 and 360 degrees can be used in the command, however, the signal generator rounds up or down to get 90, 180, 270 and 0 degree settings. For example 140 degrees will cause the signal generator to use the 180 degree setting.

If this command is executed when the clock rate is less than 10 MHz or greater than 200 MHz, the resolution of this setting changes to 180 degrees, and the maximum phase becomes 180 degrees.

Example

```
:DIG:CLOC:PHAS 90DEG
```

The preceding example sets the clock phase to 90 degrees. The clock signal leading edge transition will be delayed by 1/4 of a clock period relative to the leading edge data transition.

***RST**	+0.00000000E+000
Range	0 – 360 deg
Key Entry	**Clock Phase**

:DIGital:CLOCk:POLarity

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:CLOCk:POLarity POSitive|NEGative
[:SOURce]:DIGital:CLOCk:POLarity?
```

This command sets the alignment for the clock signal to positive or negative. Positive selects the leading edge transition of the clock signal to align with the leading edge data transition and negative selects the falling edge transition of the clock signal to align with the leading edge of the data.

Example

```
:DIG:CLOC:POL NEG
```

The preceding example sets the clock falling edge transition to align with the leading edge data transition.

***RST**	POS
Key Entry	**Clock Polarity**

:DIGital:CLOCk:RATE

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:CLOCk:RATE <val>
[:SOURce]:DIGital:CLOCk:RATE?
```

This command sets the clock rate. If an external clock is used, the rate set with this command must match the external clock rate. Only clock phase settings of 0 or 180 degrees are valid for a clock rate setting below 10 MHz or above 200 MHz. The variable `<val>` is a expressed in hertz

Example

```
:DIG:CLOC:RATE 200MHZ
```

The preceding example sets the clock rate to 200 megahertz.

***RST** +1.000000E+008

Range 1 kHz–400 MHz

Key Entry **Clock Rate**

:DIGital:CLOCk:REFerence:FREQuency

Supported E8267D Option 601or 602 with Option 004

```
[:SOURce]:DIGital:CLOCk:REFence:FREQuency <freq>
[:SOURce]:DIG:CLOC:REF:FREQ?
```

This command allows you to specify the frequency of the external reference supplied to the Freq Ref connector. This command is valid only when the clock source is set to internal.

If this command is executed when the clock source is not set to internal, the parameter value is changed, but it is not used by the signal generator until the clock source is changed to internal.

Because a query returns the currently set value, regardless of the clock source, you must query both states (reference frequency and clock source) to know the signal generator's current setup.

Example

```
:DIG:CLOC:REF:FREQ 50MHZ
```

The preceding example specifies a 50 megahertz external reference frequency.

***RST** +1.0000000E+007

Range 1–100 MHz

Key Entry **Reference Frequency**

:DIGital:CLOCk:SOURCe

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:CLOCk:SOURCe INTernal|EXTernal|DEVice
[:SOURce]:DIG:CLOC:SOURCe?
```

This command selects one of three possible clock sources.

Example

```
:DIG:CLOC:SOUR DEV
```

The preceding example uses the "Device Interface Connector" input clock.

***RST** INT

Key Entry **Clock Source**

:DIGital:CLOCk:SKEW

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:CLOCk:SKEW <val>
[:SOURce]:DIGital:CLOCk:SKEW?
```

This command sets the clock signal skew value. The skew is a fine-tune adjustment for the course tune clock phase function and helps to align the clock with valid data states. This is useful at high clock rates and available only for clock frequencies above 10 megahertz. The variable `<val>` is a expressed in nanoseconds.

Example

```
:DIG:CLOC:SKEW 2NS
```

The preceding example sets the clock skew to 2 nanoseconds.

***RST** +0.00000000E+000 ns

Range –5ns to 5ns

Key Entry **Clock Skew**

:DIGital:DATA:ALIGnment

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:ALIGnment MSB|LSB
[:SOURce]:DIGital:DATA:ALIGment?
```

This command selects the bit alignment for word less than 16 bits in length. The MSB (most significant bit) selection maintains the MSB of the word on the same data line while the LSB (least significant bit) will move depending on the word size. The opposite effect occurs when the alignment is set to LSB.

Example

```
:DIG:DATA:ALIG MSB
```

The preceding example sets the MSB word format.

***RST** LSB

Key Entry **Word Alignment**

:DIGital:DATA:BORDer

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:BORDer MSB|LSB
[:SOURce]:DIGital:DATA:BORD?
```

This command selects the bit order for data transmitted through the N5102A module. Data can be in least significant (LSB) bit first or most significant (MSB) bit first.

Example

```
:DIG:DATA:BORD MSB
```

The preceding example specifies data in MSB first format.

***RST** LSB

Key Entry **Bit Order**

:DIGital:DATA:DIRection

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:DIRection OUTPut|INPut
[:SOURce]:DIGital:DATA:DIRection?
```

This command selects an input or output direction for data flow through the N5102A module.

Example

```
:DIG:DATA:DIR INP
```

The preceding example selects input as the direction of data flow.

***RST** Output (unless only Option 004 is installed)

Key Entry **Direction**

:DIGital:DATA:IGain

Supported E8267D Option 601or 602 with Option 003

```
[:SOURce]:DIGital:DATA:IGain <val>
[:SOURce]:DIGital:DATA:IGain?
```

This command adjust the gain of the I data in the N5102A module. The adjustment does not affect the Q data. The variable `<val>` is a expressed as a percentage delta from 100%.

The offset is an adjustment to the analog level that is represented by the digital sample.

The analog voltage is limited to a 16-bit data sample. If the amplitude of the signal, after gain is applied, cannot be represented by 16 bits, the signal will be clipped.

Example

```
:DIG:DATA:IG 10
```

The preceding example turns off wideband amplitude modulation.

***RST**	+0.00000000E+000
Range	–12.5 through 12.5
Key Entry	**I Gain**

:DIGital:DATA:INEGate

Supported	E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:INEGate OFF|ON|0|1
[:SOURce]:DIGital:DATA:INEGate?
```

This command enables or disables the negation of the I data sample. Negation changes the sample by expressing it in two's complement form, multiplying by negative one, and converting back to the selected numeric format. This can be done for I samples, Q samples, or both.

The sample or word represents a quantized analog voltage level. For a 16-bit sample, the range is from 0 to 65535 in offset binary or -32768 to + 32767 in 2’s complement mode.

Example

```
:DIG:DATA:INEG ON
```

The preceding example enables negation of the I data.

***RST**	0
Key Entry	**Negate I**

:DIGital:DATA:IOFFset

Supported	E8267D Option 601or 602 with Option 003

```
[:SOURce]:DIGital:DATA:IOFFset <val>
[:SOURce]:DIGital:DATA:IOFFset?
```

This command adjusts the DC offset for I data. The command is available for the N5102A module output mode. The variable `<val>` is a expressed as a +/– 100% of the full scale value. Refer to the *PSG Key Reference* for more information.

Example

```
:DIG:DATA:IOFF 40
```

The preceding example sets the I offset to 40% of full scale.

***RST**	+0.00000000E+000
Range	–100 to +100
Key Entry	**I Offset**

:DIGital:DATA:IQSWap

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:IQSWap OFF|ON|0|1
[:SOURce]:DIGital:DATA:IQSWap?
```

This command enables or disables swapping of the I and Q data. When enabled, the I data is sent to the N5102A's Q bus and the Q data is sent to the I bus.

Example

```
:DIG:DATA:IQSW ON
```

The preceding example enables swapping of I and Q data.

***RST**	0
Key Entry	**Swap IQ**

:DIGital:DATA:NFORmat

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:NFORMat OBINary|TCOMplement
[:SOURce]:DIGital:DATA:NFORMat?
```

This command selects the binary format used to represent the transmitted data values. The selections are offset binary or 2's complement.

Example

```
:DIG:DATA:NFOR OBIN
```

The preceding example selects the offset binary format to represent data values.

***RST**	TCOM
Key Entry	**Numeric Format**

:DIGital:DATA:POLarity:FRAMe

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:POLarity:FRAMe POSitive|NEGative
[:SOURce]:DIGital:DATA:POLarity:FRAMe?
```

This command selects the polarity of the frame marker for serial transmission. The frame marker indicates the beginning of each sample or byte of data. The command is valid for serial transmission only.

POS This choice selects a positive polarity. The frame marker is high for the first data sample.

NEG This choice selects a negative polarity. The frame marker is low for the first data sample.

Example

```
:DIG:DATA:POL:FRAM NEG
```

The preceding example selects a negative polarity for the frame marker.

***RST** POS

Key Entry **Frame Polarity**

:DIGital:DATA:POLarity:IQ

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:POLarity:IQ POSitive|NEGative
[:SOURce]:DIGital:DATA:POLarity:IQ?
```

This command selects the logic level for I and Q data. Positive selects a high logic level at the output as a digital one and negative selects a low logic level at the output as a digital one.

POS This choice selects a logic high level as digital one.

NEG This choice selects a logic low level as a digital one.

Example

```
:DIG:DATA:POL:IQ NEG
```

The preceding example turns off wideband amplitude modulation.

***RST** POS

Key Entry **IQ Polarity**

:DIGital:DATA:QGain

Supported E8267D Option 601or 602 with Option 003

```
[:SOURce]:DIGital:DATA:QGain <val>
[:SOURce]:DIGital:DATA:QGain?
```

This command adjusts the gain for Q data in the N5102A module. The adjustment does not affect the I data. The variable `<val>` is a expressed as a percentage delta from 100%. The offset is an adjustment to the analog level that is represented by the digital sample. The analog voltage is limited to a 16-bit data sample.

Example

```
:DIG:DATA:QG 10
```

The preceding example increases the gain for Q data by 10% above the nominal value.

***RST** +0.00000000E+000

Range –12.5 through 12.5

Key Entry **Q Gain**

:DIGital:DATA:QNEGate

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:QNEGate OFF|ON|0|1
[:SOURce]:DIGital:DATA:QNEGate?
```

This command enables or disables the negation of the Q data sample. Negation changes the sample by expressing it in two's complement form, multiplying by negative one, and converting back to the selected numeric format.

The sample or word represents a quantized analog voltage level. For a 16-bit sample, the range is from 0 to 65535 in offset binary or -32768 to + 32767 in 2's complement mode.

Example

```
:DIG:DATA:QNEG ON
```

The preceding example enables negation of the Q data.

***RST** 0

Key Entry **Negate Q**

:DIGital:DATA:QOFFset

Supported E8267D Option 601or 602 with Option 003

```
[:SOURce]:DIGital:DATA:QOFFset <val>
[:SOURce]:DIGital:DATA:QOFFset?
```

This command adjusts the DC offset for Q data. The command is available for the N5102A module output mode. The variable `<val>` is a expressed as a +/– 100% of the full scale value.

Example

```
:DIG:DATA:QOFF 40
```

The preceding example sets the Q offset to 40% of full scale.

***RST** +0.00000000E+000

Range –100 through 100

Key Entry **Q Offset**

:DIGital:DATA:ROTation

Supported E8267D Option 601or 602 with Option 003

```
[:SOURce]:DIGital:DATA:ROTation <val>
[:SOURce]:DIGital:DATA:ROTation?
```

This command rotates the IQ data in the IQ plane. This command is valid for the N5102A output mode. The variable `<val>` is expressed in degrees with a range from 0 to 360.

Example

```
:DIG:DATA:ROT 45
```

The preceding example rotates the IQ constellation 45 degrees.

***RST** +1.00000000E+000

Range 0–360

Key Entry **Rotation**

:DIGital:DATA:SCALing

Supported E8267D Option 601or 602 with Option 003

```
[:SOURce]:DIGital:DATA:SCALing <val>
[:SOURce]:DIGital:DATA:SCALing?
```

This command enables scaling of the I and Q data to the level indicated by the `<val>` variable. This command is valid for the N5102A output mode. The variable <val> is expressed as a percentage.

Example

```
:DIG:DATA:SCAL 50
```

The preceding example scales the I and Q data to amplitude to 50% of the nominal value.

***RST**	+1.00000000E+002
Range	–100 through 100
Key Entry	**Scaling**

:DIGital:DATA:SIZE

Supported	E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:SIZE <val>
[:SOURce]:DIGital:DATA:SIZE?
```

This command selects the number of bits in each sample. A sample can have a maximum word length of 16 bits.

Example

```
:DIG:DATA:SIZE 8
```

The preceding example sets the sample word size to eight bits.

***RST**	+1.600000000E+001
Range	4–16
Key Entry	**Word Size**

:DIGital:DATA:STYPe

Supported	E8267D Option 601or 602 with Option 003

```
[:SOURce]:DIGital:DATA:STYPe IQ|IF
[:SOURce]:DIGital:DATA:STYPe?
```

This command selects the output format for the IQ data. The IQ selection outputs digital I and Q data. Whereas the IF (intermediate frequency) selection modulates the I and Q data onto the IF frequency. The IF is calculated as 1/4 the clock sample rate. This command is valid only for the N5102A output mode.

IQ	This choice outputs I and Q digital data.
IF	This choice outputs a modulated signal.

Example

```
:DIG:DATA:STYP IF
```

The preceding example sets the output data to IF.

***RST** IQ

Key Entry **Signal Type**

:DIGital:DATA:TYPE

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DATA:TYPE SAMPles|PFSamples
[:SOURce]:DIGital:DATA:TYPE?
```

This command selects filtered baseband data or unfiltered baseband data as the transmitted data type.

If this command is executed while an ARB modulation format is active, the parameter choice is changed, but it is not *used* by the interface module until a real-time modulation format is turned on.

Because a query returns the current choice, regardless of whether or not an ARB format is active, you must query both states (data type and the modulation format) to know the signal generator’s current setup.

SAMPles This choice selects DAC samples at the data transmitted.

PFSamples This choice selects pre-filtered samples which are unfiltered I and Q data.

Example

```
:DIG:DATA:TYPE PFS
```

The preceding example sets the data type to pre-filtered I and Q data.

***RST** SAMP

Key Entry **Data Type**

:DIGital:DIAGnostic:LOOPback

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:DIAGnostic:LOOPback DIGBus|CABLe|N5102A|DEVice
[:SOURce]:DIGital:DIAGnostic:LOOPback?
```

This command selects a loop back test that validates the integrity of digital data. Refer to the *PSG Key Reference* for more information.

DIGBus This choice selects a loop back test using the Digital Bus Loop Back Fixture test board.

CABLe	This choice selects a loop back test on the PSG Digital Bus connector at the signal generator side.
N5102A	This choice selects a loop back test for the N5102A module.
DEVice	This choice selects a loop back test using the LOOP BACK TEST SINGLE ENDED IO DUAL 40 PIN board.

Example

```
:DIG:DIAG:LOOP?
```

The preceding example runs the diagnostic test for device and returns a pass or fail state.

***RST** Device Intfc

Key Entry **Loop Back Test Type**

:DIGital:LOGic[:TYPE]

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:LOGic[:TYPE] LVDS|LVTT1|CMOS15|CMOS18|CMOS25|CMOS33
[:SOURce]:DIGital:LOGic[:TYPE]?
```

This command selects the logic data type used by the device being tested.

LVDS	This choice selects low voltage differential signaling as the logic data type.
LVTT1	This choice selects a low voltage TTL signal as the logic data type.
CMOS15	This choice selects a 1.5 volt CMOS signal as the logic data type.
CMOS18	This choice selects a 1.8 volt CMOS signal as the logic data type.
CMOS25	This choice selects a 2.5 volt CMOS signal as the logic data type.
CMOS33	This choice selects a 3.3 volt CMOS signal as the logic data type.

Example

```
:DIG:LOG CMOS15
```

The preceding example selects 1.5 volt CMOS as the logic data type.

***RST** CMOS33

Key Entry **Logic Type**

:DIGital:PCONfig

Supported   E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:PCONfig PARallel|SERial|PINTIQ|PINTI
[:SOURce]:DIGital:PCONfig?
```

This command selects the data transmission type used for communication between the N5102A module and the device under test. Refer to the *PSG Key Reference* for more information.

PARallel	This choice selects parallel data transmission.
SERial	This choice selects serial data transmission.
PINTIQ	This choice selects parallel interleaving data transmission. The I data is transmitted on the rising clock edge and the Q data on the falling edge.
PINTI	This choice selects parallel interleaving data transmission. The Q data is transmitted on the rising clock edge and the I data on the falling edge.

Example

```
:DIG:PCON PINTI
```

The preceding example selects parallel interleaving using the QI format

***RST**   PAR

Key Entry   **Port Config**

:DIGital:PRESet:PTHRough

Supported   E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital:PRESet:PTHRough
```

This command sets up the preset condition for the N5102A module and allows transmission of data through the module with no modifications. The command is valid only when a modulation format is active.

Example

```
:DIG:PRES:PTHR
```

The preceding example sets the N5102A module to a preset condition and allows data to pass through unmodified.

Key Entry   **Pass Through Preset**

:DIGital[:STATe]

Supported E8267D Option 601or 602 with Option 003 or 004

```
[:SOURce]:DIGital[:STATe] 0|1|OFF|ON
[:SOURce]:DIGital[:STATe]?
```

This command enables or disables the operating state of the N5102A module.

Example

```
:DIG ON
```

The preceding example turns on the N5102A module.

***RST** OFF

Key Entry **N5102A Off On**

7 SCPI Command Compatibility

This chapter provides a compatibility listing of SCPI commands. Many commands unique to other Agilent signal generator models are also supported by the PSG Signal Generator:

:SYSTem:IDN

Supported All

```
:SYSTem:IDN "<string>"
```

This command modifies the identification string that the `*IDN?` query returns. Sending an empty string returns the query output to its factory shipped setting. The maximum string length is 72 characters.

Modification of the `*IDN?` query output enables the PSG to identify itself as another signal generator when it is used as a backward compatible replacement.

The display diagnostic information, shown by pressing the **Diagnostic Info** softkey, is not affected by this command.

Example

```
:SYST:IDN "Agilent Technologies, Exxxx, US4000000, c.00.00.1234"
```

The preceding example changes and sets the identification string for the signal generator.

E8257D / 67D Compatible Commands

The following commands are compatible with the E8257D and E8267D signal generators.

:DATA:PRAM?

Supported E4438C with Option 601or 602

`:MEMory:DATA:PRAM?`

This query determines whether there is a user-defined pattern in the pattern RAM (PRAM). This command is not compatible with the “:DATA:PRAM:FILE:BLOCk” or “:DATA:PRAM:FILE:LIST” commands.

***RST** 0

:DATA:PRAM:BLOCk

Supported E4438C with Option 601or 602

`:MEMory:DATA:PRAM:BLOCk <data_block>`

This command downloads the block-formatted data directly into pattern RAM. This command is still valid for backward compatibility; however, it has been replaced by the “:DATA:PRAM:FILE:BLOCk” command.

:DATA:PRAM:LIST

Supported All with Option 601or 602

`:MEMory:DATA:PRAM:LIST <uint8>[,<uint8>,<...>]`

This command downloads the list-formatted data directly into pattern RAM. This command is still valid for backward compatibility; however, it has been replaced by the “:DATA:PRAM:FILE:LIST” command.

<uint8> This variable is any of the valid 8-bit, unsigned integer values between 0 and 255.

[,<uint8>,<...>] This variable identifies the value of the second and subsequent 8-bit unsigned integer variables.

Range 0–255

E8241A / 44A / 51A / 54A and the E8247C / 57C / 67C PSG Compatible SCPI Commands

All commands are fully supported. To use the commands, select *SCPI* as the remote language. See “:LANGuage” on page 101 for selecting the language type.

8340B / 41B and 8757D Compatible Commands

The tables in this section provide the following:

Table 7-1 on page 380: a comprehensive list of 8340B/41B and 8757D programming codes, listed in alphabetical order. The equivalent SCPI command sequence for each supported code is provided; codes that are *not* supported by the PSG family are indicated as such in the command column.

Table 7-2 on page 393: a list of the implemented 8340B/41B and 8757D programming codes that set the active function. This table also indicates which codes are compatible with the RB command (knob), and lists the operation active (OA) query, the operation prior (OP) query, and the increment (up), and the decrement (down) SCPI commands.

NOTE Compatibility is provided for GPIB only; RS-232 and LAN are *not* supported.

When using the programming codes in this section, you can:

- set the PSG system language to 8340 or 8757 for the current session:

  Utility > **GPIB / RS-232 LAN** > **Preset Language** > **8340B** (or **8757D**)

  or

  `:SYST:LANG "8340" (or "8757")`

- set the PSG system language to 8340 or 8757 so that it does not reset with either preset, instrument power cycle or `*RST` command:

  Utility > **Power On / Preset** > **Preset Language** > **8340B** (or **8757D**)

  or

  `:SYST:PRESET:LANG "8340"(or "8757")`

- set the `*IDN?` response to any 8340-like response you prefer:

  use the command :SYSTem:IDN on page 376.

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence
A1	Internal leveling mode	Y	Y	[:SOURce]:POWer:ALC:SOURce INTernal
A2	External leveling mode with diode detector	Y	Y	[:SOURce]:POWer:ALC:SOURce DIODe [:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling <val> dB
A3	External leveling mode with power meter	Y	Y	*supported, but has no effect on PSG*
AK0	Amplitude markers off	Y	Y	[:SOURce]:MARKer:AMPLitude OFF|0
AK1	Amplitude markers on	Y	Y	[:SOURce]:MARKer:AMPLitude ON|1
AL0	Alternate sweep mode off	Y	Y	:SYSTem:ALTernate:STATe OFF
AL1	Alternate sweep mode on	Y	Y	:SYSTem:ALTernate:STATe ON :SYSTem:ALTernate n
AM0	Amplitude modulation off	Y	N	[:SOURce]:AM1:STATe OFF|0 [:SOURce]:AM2:STATe OFF|0
AM1	Amplitude modulation on	Y	N	[:SOURce]:AM1:STATe OFF|0 [:SOURce]:AM2:SOURce EXT[1] [:SOURce]:AM2:EXTernal[1]:COUPling DC [:SOURce]:AM2:DEPTh 100 [:SOURce]:AM2:EXTernal[1]:IMPedance 600 [:SOURce]:AM2:STATe ON|1
AS0	Alternate state selection: select current front panel	Y	Y	*supported, but no equivalent SCPI command sequence*
AS1	Alternate state selection: select recalled state	Y	Y	*supported, but no equivalent SCPI command sequence*
AT	Set attenuator	Y	N	[:SOURce]:POWer:ATTenuation <val><unit>
AU	Auto-coupled mode to obtain shortest possible sweep time	Y	N	[:SOURce]:SWEep:TIME:AUTO ON|1
BC	Advance to next frequency bandcrossing	N	N	*not supported*
C1	1 MHz crystal marker frequency	N	Y	*supported, but has no effect on PSG*
C2	10 MHz crystal marker frequency	N	Y	*supported, but has no effect on PSG*
C3	50 MHz crystal marker frequency	N	Y	*supported, but has no effect on PSG*
C4	External crystal marker frequency	N	Y	*supported, but has no effect on PSG*

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence
CA0	Amplitude crystal markers off	N	Y	*supported, but has no effect on PSG*
CA1	Amplitude crystal markers on	N	Y	*supported, but has no effect on PSG*
CF	Center frequency (step sweep)	Y	Y	[:SOURce]:SWEep:MODE AUTO [:SOURce]:FREQuency:MODE SWEep [:SOURce]:FREQuency:CENTer <val><unit>
CL0	Intensity crystal markers off	N	Y	*supported, but has no effect on PSG*
CL1	Intensity crystal markers on	N	Y	*supported, but has no effect on PSG*
CS	Clear both status bytes	Y	Y	*CLS
CW	Set CW frequency	Y	Y	[:SOURce]:SWEep:MODE AUTO [:SOURce]:FREQuency:MODE CW [:SOURce]:FREQuency[:CW] <val><unit>
DB	dB(m) terminator	Y	Y	DB
DF	Delta frequency (step sweep)	Y	Y	[:SOURce]:SWEep:MODE AUTO [:SOURce]:FREQuency:MODE SWEep [:SOURce]:FREQuency:SPAN <val> <unit>
DM	dB(m) terminator	Y	Y	DB
DN	Step down (decrements active function by step value)	Y	Y	*supported, see Table 6-2 on page 234*
DP0	Display blanking off	N	Y	DISPlay[:WINDow][:STATe] ON|1
DP1	Display blanking on	N	Y	DISPlay[:WINDow][:STATe] OFF|0
DU0	Display update off	Y	Y	DISPlay[:WINDow][:STATe] OFF|0
DU1	Display update on	Y	Y	DISPlay[:WINDow][:STATe] ON|1
EF	Entry display off	Y	Y	DISPlay[:WINDow][:STATe] ON|1
EK	Enable knob	N	N	*not supported*
EM0	Extended marker mode off	N	Y	*supported, but* *no equivalent SCPI command sequence*
EM1	Extended marker mode on	N	Y	*supported, but* *no equivalent SCPI command sequence*
F1	20 MHz/V FM sensitivity	N	N	*not supported*
F2	6 MHz/V FM sensitivity	N	N	*not supported*

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

<table>
<tr><th>Cmd</th><th>Description</th><th>8340</th><th>8757</th><th>Equivalent SCPI Command Sequence</th></tr>
<tr><td>FA</td><td>Start frequency (step sweep)</td><td>Y</td><td>Y</td><td>[:SOURce]:SWEep:MODE AUTO
[:SOURce]:FREQuency:MODE SWEep
[:SOURce]:FREQuency:STARt <val><unit></td></tr>
<tr><td>FB</td><td>Stop frequency (step sweep)</td><td>Y</td><td>Y</td><td>[:SOURce]:SWEep:MODE AUTO
[:SOURce]:FREQuency:MODE SWEep
[:SOURce]:FREQuency:STOP <val><unit></td></tr>
<tr><td>FL0</td><td>CW filter off</td><td>N</td><td>Y</td><td>supported, but has no effect on PSG</td></tr>
<tr><td>FL1</td><td>CW filter on</td><td>N</td><td>Y</td><td>supported, but has no effect on PSG</td></tr>
<tr><td>FM0</td><td>Frequency modulation off</td><td>Y</td><td>N</td><td>[:SOURce]:FM1:STATe OFF|0
[:SOURce]:FM2:STATe OFF|0</td></tr>
<tr><td>FM1</td><td>Frequency modulation on</td><td>Y</td><td>N</td><td>[:SOURce]:FM1:STATe OFF|0
[:SOURce]:FM2:SOURce EXT2
[:SOURce]:FM2:EXTernal2:COUPling DC
[:SOURce]:FM2:EXTernal2:IMPedance 50
[:SOURce]:FM2:STATe ON|1</td></tr>
<tr><td>FM1</td><td>Frequency modulation sensitivity</td><td>Y</td><td>N</td><td>[:SOURce]:FM2[:DEViation] <val><unit></td></tr>
<tr><td>FP</td><td>Fast phaselock</td><td>Y</td><td>N</td><td>supported, but has no effect on PSG</td></tr>
<tr><td>GZ</td><td>GHz terminator</td><td>Y</td><td>Y</td><td>GHZ</td></tr>
<tr><td>HZ</td><td>Hz terminator</td><td>Y</td><td>Y</td><td>HZ</td></tr>
<tr><td>IF</td><td>Increment frequency</td><td>Y</td><td>N</td><td>TRIGger[:SEQuence][:IMMediate]
or
[:SOURce]:FREQuency[:CW] UP</td></tr>
<tr><td>IL</td><td>Input learn string</td><td>Y</td><td>Y</td><td>supported, but
no equivalent SCPI command sequence</td></tr>
</table>

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence

IP	Instrument preset	Y	N	`SYSTem:PRESet`
				`[:SOURce]:FREQuency[:CW]:STEP [:INCRement] 1 GHZ`
				
				`[:SOURce]:FREQuency:MULTiplier <saved multiplier>`
				
				`[:SOURce]:SWEep:MODE AUTO`
				`[:SOURce]:FREQuency:MODE SWEep`
				`[:SOURce]:FREQuency:STARt 2 GHz or MIN`
				`[:SOURce]:FREQuency:STOP MAX`
				`[:SOURce]:POWer[:LEVel][:IMMediate] [:AMPLitude] 0 dB`
				
				`OUTput[:STATe] ON|1`

IP	Instrument preset	N	Y	`SYSTem:PRESet`
				`SYSTem:LANGuage "8757"`
				`[:SOURce]:SWEep:MODE AUTO`
				`[:SOURce]:FREQuency:MODE SWEep`
				`[:SOURce]:FREQuency:STARt 2 GHz or MIN`
				`[:SOURce]:FREQuency:STOP MAX`
				`[:SOURce]:POWer[:LEVel][:IMMediate] [:AMPLitude] 0 dB`
				
				`OUTput[:STATe] ON|1`

IX	Input micro learn string	N	Y	*supported, but has no effect on PSG*
KR	Key release	Y	Y	*supported, but no equivalent SCPI command sequence*
KZ	kHz terminator	Y	Y	`KHZ`
M0 MO	Frequency marker off	Y	Y	`[:SOURce]:MARKer[n]:[STATe] OFF|0`
MA	Turn on and set frequency marker 0	Y	Y	`[:SOURce]:MARKer0:[STATe] ON|1` `[:SOURce]:MARKer0:FREQuency <val><unit>`
M1	Turn on and set frequency marker 1	Y	Y	`[:SOURce]:MARKer1:[STATe] ON|1` `[:SOURce]:MARKer1:FREQuency <val><unit>`
M2	Turn on and set frequency marker 2	Y	Y	`[:SOURce]:MARKer2:[STATe] ON|1` `[:SOURce]:MARKer2:FREQuency <val><unit>`

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence
M3	Turn on and set frequency marker 3	Y	Y	`[:SOURce]:MARKer3:[STATe] ON|1` `[:SOURce]:MARKer3:FREQuency <val><unit>`
M4	Turn on and set frequency marker 4	Y	Y	`[:SOURce]:MARKer4:[STATe] ON|1` `[:SOURce]:MARKer4:FREQuency <val><unit>`
M5	Turn on and set frequency marker 5	Y	Y	`[:SOURce]:MARKer5:[STATe] ON|1` `[:SOURce]:MARKer5:FREQuency <val><unit>`
M6	Turn on and set frequency marker 6	Y	Y	`[:SOURce]:MARKer6:[STATe] ON|1` `[:SOURce]:MARKer6:FREQuency <val><unit>`
M7	Turn on and set frequency marker 7	Y	Y	`[:SOURce]:MARKer7:[STATe] ON|1` `[:SOURce]:MARKer7:FREQuency <val><unit>`
M8	Turn on and set frequency marker 8	Y	Y	`[:SOURce]:MARKer8:[STATe] ON|1` `[:SOURce]:MARKer8:FREQuency <val><unit>`
M9	Turn on and set frequency marker 9	Y	Y	`[:SOURce]:MARKer9:[STATe] ON|1` `[:SOURce]:MARKer9:FREQuency <val><unit>`
MC	Active marker to center frequency	Y	Y	*supported, but* *no equivalent SCPI command sequence*
MD	Marker delta	N	N	*not supported*
MP0	Marker 1-2 sweep off	N	N	*not supported*
MP1	Marker 1-2 sweep on	N	N	*not supported*
MS	Milliseconds terminator	Y	Y	`MS`
MZ	MHz terminator	Y	Y	`MHZ`
NA	Network analyzer mode	N	Y	*supported, but* *no equivalent SCPI command sequence*
NT	Network analyzer trigger	N	Y	*supported, but has no effect on PSG*
OA	Output active parameter	Y	Y	*supported, see Table 6-2 on page 234*
OB	Output next bandcross frequency	N	N	*not supported*

OC	Output coupled parameters
	(start frequency, center frequency,
	sweep time)
		Y	Y	`[:SOURce]:FREQuency:STARt?`
				`[:SOURce]:FREQuency:CENTer?`
				`[:SOURce]:SWEep:TIME?`

OD	Output diagnostic values	N	N	*not supported*

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence
OE	Output when executed	N	Y	*supported, but no equivalent SCPI command sequence*
OF	Output fault	Y	N	*supported, but no equivalent SCPI command sequence*
OI	Output identification	Y	Y	`*IDN?`
OK	Output last lock frequency	N	N	*not supported*
OL	Output learn string	Y	Y	*supported, but no equivalent SCPI command sequence*
OM	Output mode string	N	Y	*supported, but no equivalent SCPI command sequence*
OP	Output interrogated parameter	Y	Y	*supported, see Table 6-2 on page 234*
OPA2	Output external detector coupling factor	Y	Y	`[:SOURce]:POWer:ALC:SOURce:EXTernal :COUPling?`
OPAT	Output attenuator	Y	N	`[:SOURce]:POWer:ATTenuation?`
OPCF	Output center frequency	Y	Y	`[:SOURce]:FREQuency:CENTer?`
OPCW	Output CW frequency	Y	Y	`[:SOURce]:FREQuency:CW?`
OPDF	Output delta frequency	Y	Y	`[:SOURce]:FREQuency:SPAN?`
OPFA	Output start frequency	Y	Y	`[:SOURce]:FREQuency:STARt?`
OPFB	Output stop frequency	Y	Y	`[:SOURce]:FREQuency:STOP?`
OPFM1	Output FM sensitivity	Y	N	`[:SOURce]:FM2[:DEViation]?`
OPMA	Output marker 0 frequency	Y	Y	`[:SOURce]:MARKer0:FREQuency?`
OPM1	Output marker 1 frequency	Y	Y	`[:SOURce]:MARKer1:FREQuency?`
OPM2	Output marker 2 frequency	Y	Y	`[:SOURce]:MARKer2:FREQuency?`
OPM3	Output marker 3 frequency	Y	Y	`[:SOURce]:MARKer3:FREQuency?`
OPM4	Output marker 4 frequency	Y	Y	`[:SOURce]:MARKer4:FREQuency?`
OPM5	Output marker 5 frequency	Y	Y	`[:SOURce]:MARKer5:FREQuency?`
OPM6	Output marker 6 frequency	Y	Y	`[:SOURce]:MARKer6:FREQuency?`
OPM7	Output marker 7 frequency	Y	Y	`[:SOURce]:MARKer7:FREQuency?`

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence
OPM8	Output marker 8 frequency	Y	Y	`[:SOURce]:MARKer8:FREQuency?`
OPM9	Output marker 9 frequency	Y	Y	`[:SOURce]:MARKer9:FREQuency?`
OPPL	Output power level	Y	Y	`[:SOURce]:POWer[:LEVel][:IMMediate] [:AMPLitude]?`
OPPS	Output power sweep span	Y	Y	`[:SOURce]:POWer:SPAN?`
OPSB	Output # of sweep buckets	N	N	*supported, but* *no equivalent SCPI command sequence*
OPSF	Output frequency step size	Y	Y	`[:SOURce]:FREQuency[:CW]:STEP [:INCRement]?`
OPSHA1	Output power level	Y	N	`[:SOURce]:POWer[:LEVel][:IMMediate] [:AMPLitude]?`
OPSHA2	Output ALC level	Y	N	`[:SOURce]:POWer:ALC:LEVel?`
OPSHA3	Output ALC level	Y	N	`[:SOURce]:POWer:ALC:LEVel?`
OPSHAZ	Output ALC level	Y	N	`[:SOURce]:POWer:ALC:LEVel?`
OPSHCF	Output frequency step size	Y	N	`[:SOURce]:FREQuency[:CW]:STEP [:INCRement]?`

OPSHCW	Output swept CW frequency	Y	Y	`[:SOURce]:FREQuency:STARt?`
				or
				`[:SOURce]:FREQuency:STOP?`

OPSHFA	Output frequency multiplier	Y	Y	`[:SOURce]:FREQuency:MULTiplier?`
OPSHFB	Output frequency offset	Y	Y	`[:SOURce]:FREQuency:OFFSet?`
OPSHPL	Output power step size	Y	N	`[:SOURce]:POWer[:LEVel][:IMMediate] [:AMPLitude]:STEP[:INCRement]?`
OPSHPS	Output ALC level	Y	Y	`[:SOURce]:POWer:ALC:LEVel?`
OPSHRF	Output power level	Y	N	`[:SOURce]:POWer[:LEVel][:IMMediate] [:AMPLitude]?`
OPSHSL	Output attenuator	Y	N	`[:SOURce]:POWer:ATTenuation?`
OPSHSN	Output sweep step points	N	Y	`[:SOURce]:SWEep:POINts?`
OPSL	Output power slope	Y	Y	`[:SOURce]:POWer:SLOPe?`
OPSM	Output manual frequency	Y	Y	`[:SOURce]:FREQuency:MANual?`

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence
OPSN	Output sweep step points	Y	Y	`[:SOURce]:SWEep:POINts?`
OPSP	Output power step size	Y	Y	`[:SOURce]:POWer[:LEVel][:IMMediate]` `[:AMPLitude]:STEP[:INCRement]?`
OPST	Output sweep time	Y	Y	`[:SOURce]:SWEep:TIME?`
OPTL	Output sweep time limit	Y	Y	`[:SOURce]:SWEep:TIME:LLIMit?`
OR	Output internally measured power level	N	N	*not supported*
OS	Output status bytes	Y	Y	*supported, but* *no equivalent SCPI command sequence*
OX	Output micro learn string	N	Y	*supported, but has no effect on PSG*
PL	Set power level	Y	Y	`[:SOURce]:POWer:ATTenuation:AUTO ON|1` `[:SOURce]:POWer[:LEVel][:IMMediate]` `[:AMPLitude] <val><unit>`
PM0	Pulse modulation off	Y	Y	`[:SOURce]:PULM:STATe OFF|0`
PM1	Pulse modulation on	Y	N	`[:SOURce]:PULM:SOURce EXTernal` `[:SOURce]:PULM:STATe ON|1`
PM1	27.8 KHz square wave pulse modulation on	N	Y	`[:SOURce]:PULM:SOURce SCALar` `[:SOURce]:PULM:STATe ON|1`
PS0	Power sweep off	Y	Y	`[:SOURce]:POWer:MODE FIXed`
PS1	Power sweep on	Y	Y	`[:SOURce]:POWer:MODE SWEep` `[:SOURce]:POWer:SPAN <val> dB`
R2	Extended status byte #2 mask	N	Y	*supported, but has no effect on PSG*
RB	Control knob remotely	Y	Y	*supported, but* *no equivalent SCPI command sequence*
RC	Recall state	Y	Y	`*RCL <reg_num>[,<seq_num>]`
RE	Extended status byte mask	Y	Y	*supported, but* *no equivalent SCPI command sequence*
RF0	RF output off	Y	Y	`OUTPut[:STATe] OFF|0`
RF1	RF output on	Y	Y	`OUTPut[:STATe] ON|1`
RM	Status byte mask	Y	Y	`*SRE <mask>`

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence
RP0	RF peaking off	Y	N	*supported, but has no effect on PSG*
RP0	RF blanking off	N	Y	*supported, but has no effect on PSG*
RP1	RF peaking on	Y	N	*supported, but has no effect on PSG*
RP1	RF blanking on	N	Y	*supported, but has no effect on PSG*
RS	Reset sweep	Y	Y	*supported, but* *no equivalent SCPI command sequence*

S1	Continuous sweep mode	Y	Y	`[:SOURce]:SWEep:MODE AUTO`
				
				`[:SOURce]:SWEep:GENeration ANALog:TRIGger[:SEQuence]:SOURce IMMediate:INITiate:CONTinuous[:ALL] ON`

S2	Single sweep mode	Y	Y	`[:SOURce]:SWEep:MODE AUTO`
				
				`[:SOURce]:SWEep:GENeration ANALog:TRIGger[:SEQuence]:SOURce IMMediate:INITiate:CONTinuous[:ALL] OFF`

S3	Manual frequency sweep mode	Y	Y	`[:SOURce]:SWEep:MODE MANual`
				
				`[:SOURce]:SWEep:GENeration ANALog:TRIGger[:SEQuence]:SOURce IMMediate:INITiate:CONTinuous[:ALL] OFF`

SB	Number of sweep buckets	N	Y	*supported, but* *no equivalent SCPI command sequence*
SC	Seconds terminator	Y	Y	`S`
SF	Frequency step size	Y	Y	`[:SOURce]:FREQuency[:CW]:STEP[:INCRement] <val><unit>`

SG	Single sweep mode	Y	Y	`[:SOURce]:SWEep:MODE AUTO`
				
				`[:SOURce]:SWEep:GENeration ANALog :TRIGger[:SEQuence]:SOURce IMMediate :INITiate:CONTinuous[:ALL] OFF`

SH	Shift prefix	Y	Y	*supported, but* *no equivalent SCPI command sequence*
SH01	Blank display	N	Y	`DISPlay[:WINDow][:STATe] OFF|0`

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

<table>
<tr><th>Cmd</th><th>Description</th><th>8340</th><th>8757</th><th>Equivalent SCPI Command Sequence</th></tr>
<tr><td>SHA1</td><td>Disable ALC and set power level</td><td>Y</td><td>N</td><td>[:SOURce]:POWer:ALC[:STATe] OFF|0
[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] <val><unit></td></tr>
<tr><td>SHA2</td><td>External leveling mode with millimeter head module</td><td>Y</td><td>N</td><td>[:SOURce]:POWer:ALC:SOURce MMHead
[:SOURce]:POWer:ALC:LEVel <val>dB</td></tr>
<tr><td>SHA3</td><td>Directly control linear modulator circuit (bypassing ALC)</td><td>Y</td><td>N</td><td>[:SOURce]:POWer:ATTenuation:AUTO OFF|0
[:SOURce]:POWer:ALC[:STATe] OFF|0
[:SOURce]:POWer:ALC:LEVel <val>dB</td></tr>
<tr><td>SHAK</td><td>Immediate YTF peak</td><td>Y</td><td>N</td><td>supported, but has no effect on PSG</td></tr>
<tr><td>SHAL</td><td>Retain multiplication factor on power on/off and preset</td><td>Y</td><td>Y</td><td>supported, but
no equivalent SCPI command sequence</td></tr>
<tr><td>SHAM</td><td>Pulse modulation enhancement</td><td>Y</td><td>N</td><td>supported, but has no effect on PSG</td></tr>
<tr><td>SHAZ</td><td>External leveling mode with millimeter head module</td><td>Y</td><td>N</td><td>[:SOURce]:POWer:ALC:SOURce MMHead
[:SOURce]:POWer:ALC:LEVel <val>dB</td></tr>
<tr><td>SHCF</td><td>Frequency step size</td><td>Y</td><td>N</td><td>[:SOURce]:FREQuency[:CW]:STEP[:INCRement]
<val><unit></td></tr>
<tr><td>SHCF</td><td>Coarse CW resolution</td><td>N</td><td>Y</td><td>supported, but has no effect on PSG</td></tr>
<tr><td>SHCW</td><td>Swept CW</td><td>N</td><td>Y</td><td>[:SOURce]:SWEep:MODE AUTO
[:SOURce]:FREQuency:MODE SWEep
[:SOURce]:FREQuency:STARt <val><unit>
[:SOURce]:FREQuency:STOP <val><unit></td></tr>
<tr><td>SHDF</td><td>Fine CW resolution</td><td>N</td><td>Y</td><td>supported, but has no effect on PSG</td></tr>
<tr><td>SHEF</td><td>Restore cal. const. access function</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SHFA</td><td>Frequency multiplier</td><td>Y</td><td>Y</td><td>[:SOURce]:FREQuency:MULTiplier <val></td></tr>
<tr><td>SHFB</td><td>Frequency offset</td><td>Y</td><td>Y</td><td>[:SOURce]:FREQuency:OFFSet <val><unit></td></tr>
<tr><td>SHIP</td><td>Reset multiplication factor to 1 and preset instrument</td><td>Y</td><td>Y</td><td>supported, but
no equivalent SCPI command sequence</td></tr>
<tr><td>SHM0</td><td>All frequency markers off</td><td>Y</td><td>Y</td><td>[:SOURce]:MARKer:AOFF</td></tr>
<tr><td>SHM1</td><td>Turn on and set marker delta</td><td>N</td><td>Y</td><td>[:SOURce]:MARKer:MODE DELTa</td></tr>
<tr><td>SHM2</td><td>Enable counter interface</td><td>N</td><td>Y</td><td>supported, but has no effect on PSG</td></tr>
<tr><td>SHM3</td><td>Disable counter interface</td><td>N</td><td>Y</td><td>supported, but has no effect on PSG</td></tr>
</table>

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence
SHM4	Diagnostics: test/display results	N	N	*not supported*
SHMO	All frequency markers off	N	Y	`[:SOURce]:MARKer:AOFF`
SHMP	Set start frequency to marker 1 and set stop frequency to marker 2	Y	Y	`[:SOURce]:SWEep:MARKer:XFER`
SHPL	Power step size	Y	N	`[:SOURce]:POWer[:LEVel][:IMMediate]` `[:AMPLitude]:STEP[:INCRement] <val>`

SHPM	27.8 KHz square wave pulse modulation on	Y	Y	`[:SOURce]:PULM:SOURce SCALar`
				
				`[:SOURce]:PULM:STATe ON|1:OUTPut`
				`:MODulation[:STATe] ON|1`

SHPS	Decouple attenuator and ALC (control ALC independently)	Y	Y	`[:SOURce]:POWer:ATTenuation:AUTO OFF|0`
				`[:SOURce]:POWer:ALC[:STATe] ON|1`
				`[:SOURce]:POWer:ALC:LEVel <val>dB`

SHRC	Unlock save/recall	Y	Y	*supported, but* *no equivalent SCPI command sequence*

SHRF	Disable ALC and set power level	Y	N	`[:SOURce]:POWer:ALC[:STATe] OFF|0`
				
				`[:SOURce]:POWer[:LEVel][:IMMediate]`
				`[:AMPLitude] <val><unit>`

SHRP	Auto track	Y	N	*supported, but has no effect on PSG*
SHS10	Disable display update	Y	N	`DISPlay[:WINDow][:STATe] OFF|0`
SHS11	Re-enable display update	Y	N	`DISPlay[:WINDow][:STATe] ON|1`
SHS3	Display fault diagnostic	N	N	*not supported*
SHSL	Set attenuator from front panel	Y	Y	`[:SOURce]:POWer:ATTenuation <val><unit>`

SHSN	Stepped sweep	N	Y	`[:SOURce]:SWEep:MODE AUTO`
				`[:SOURce]:SWEep:GENeration STEPped`
				`[:SOURce]:LIST:TYPE STEP`
				
				`[:SOURce]:LIST:TRIGger:SOURce IMMediate`
				`:TRIGger[:SEQuence]:SOURce IMMediate`
				`:INITiate:CONTinuous[:ALL] ON`
				
				`[:SOURce]:SWEep:POINts <val>`

SHSS	Reset step sizes to default values	N	Y	*supported, but has no effect on PSG*

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

<table>
<tr><th>Cmd</th><th>Description</th><th>8340</th><th>8757</th><th>Equivalent SCPI Command Sequence</th></tr>
<tr><td>SHST</td><td>Zoom function</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SHSV</td><td>Lock save/recall</td><td>Y</td><td>Y</td><td>supported, but
no equivalent SCPI command sequence</td></tr>
<tr><td>SHT1</td><td>Test displays</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SHT2</td><td>Bandcrossing penlift</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SHT3</td><td>Display unlock indicators</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SHGZ</td><td>IO Channel</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SHMZ</td><td>IO Subchannel</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SHKZ</td><td>Write to IO</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SHHZ</td><td>Read from IO</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SHVR</td><td>Frequency offset</td><td>N</td><td>N</td><td>not supported</td></tr>
<tr><td>SL0</td><td>Power slope off</td><td>Y</td><td>Y</td><td>[:SOURce]:POWer:SLOPe:STATe OFF|0</td></tr>
<tr><td>SL1</td><td>Power slope on</td><td>Y</td><td>N</td><td>[:SOURce]:POWer:SLOPe:STATe ON|1
[:SOURce]:POWer:SLOPe <value> [DB/GHz]</td></tr>
<tr><td>SL1</td><td>Power slope on</td><td>N</td><td>Y</td><td>[:SOURce]:POWer:SLOPe:STATe ON|1
[:SOURce]:POWer:SLOPe <value> [DB/Hz]</td></tr>
<tr><td>SM</td><td>Manual frequency sweep mode</td><td>Y</td><td>Y</td><td>[:SOURce]:SWEep:MODE MANual
[:SOURce]:FREQuency:MANual <val><unit></td></tr>
<tr><td>SN</td><td>Number of points in a stepped sweep</td><td>Y</td><td>Y</td><td>[:SOURce]:SWEep:MODE AUTO
[:SOURce]:SWEep:GENeration STEPped
[:SOURce]:LIST:TYPE STEP

[:SOURce]:LIST:TRIGger:SOURce
BUS:TRIGger[:SEQuence]:SOURce
IMMediate:INITiate:CONTinuous[:ALL] ON

[:SOURce]:SWEep:POINts <val></td></tr>
<tr><td>✻✩</td><td>Power step size</td><td>Y</td><td>Y</td><td>[:SOURce]:POWer[:LEVel][:IMMediate]
[:AMPLitude]:STEP[:INCRement] <val></td></tr>
<tr><td>ST</td><td>Sweep time</td><td>Y</td><td>Y</td><td>[:SOURce]:SWEep:MODE AUTO
[:SOURce]:SWEep:TIME <val> <unit></td></tr>
<tr><td>SV</td><td>Save state</td><td>Y</td><td>Y</td><td>*SAV <reg_num>[,<seq_num>]</td></tr>
</table>

Table 7-1 8340B / 41B Prog. Codes & Equivalent SCPI Sequences (Continued)

Cmd	Description	8340	8757	Equivalent SCPI Command Sequence
SW0	Swap network analyzer channels	Y	Y	*supported, but no equivalent SCPI command sequence*
SW1	Swap network analyzer channels	Y	Y	*supported, but no equivalent SCPI command sequence*
SX	External sweep type	N	Y	*supported, but has no effect on PSG*
T1	Free run sweep trigger mode	Y	Y	`:TRIGger[:SEQuence]:SOURce IMMediate` `:INITiate:CONTinuous[:ALL] ON`
T2	Line sweep trigger mode	N	N	*not supported*
T3	External sweep trigger mode	Y	Y	`:TRIGger[:SEQuence]:SOURce EXTernal` `:INITiate:CONTinuous[:ALL] ON`
T4	Single sweep trigger mode	N	Y	`:INITiate[:IMMediate][:ALL]`
TL	Sweep time limit	Y	Y	`[:SOURce]:SWEep:TIME:LLIMit <val> <unit>`
TS	Take sweep	Y	Y	`:TSWeep`
UP	Step up (increments active function by step value)	Y	Y	*supported, see Table 6-2 on page 234*
VR	CW vernier	N	Y	*supported, but has no effect on PSG*

Table 7-2 8340 and 8757 Code Compatibility

Code	Sets Active Function	Comp. with OA / OP	Comp. with UP / DN	Comp. with RB (Knob)	Equivalent SCPI Commands for OA / OP query and UP / DN command
A2	✓	✓	✓		[:SOURce]:POWer:ALC:SOURce:EXTernal:COUPling? [:SOURce]:POWer:ATTenuation UP [:SOURce]:POWer:ATTenuation DOWN
AT	✓	✓	✓		[:SOURce]:POWer:ATTenuation? [:SOURce]:POWer:ATTenuation UP [:SOURce]:POWer:ATTenuation DOWN
CF	✓	✓			[:SOURce]:FREQuency:CENTer?
CW	✓	✓	✓	✓	[:SOURce]:FREQuency[:CW]? [:SOURce]:FREQuency[:CW] UP [:SOURce]:FREQuency[:CW] DOWN
DF	✓	✓			[:SOURce]:FREQuency:SPAN?
FA	✓	✓			[:SOURce]:FREQuency:STARt?
FB	✓	✓			[:SOURce]:FREQuency:STOP?
FM1	✓	✓			[:SOURce]:FM2[:DEViation]?
MA	✓	✓			[:SOURce]:MARKer0:FREQuency?
M1	✓	✓			[:SOURce]:MARKer1:FREQuency?
M2	✓	✓			[:SOURce]:MARKer2:FREQuency?
M3	✓	✓			[:SOURce]:MARKer3:FREQuency?
M4	✓	✓			[:SOURce]:MARKer4:FREQuency?
M5	✓	✓			[:SOURce]:MARKer5:FREQuency?
M6	✓	✓			[:SOURce]:MARKer6:FREQuency?
M7	✓	✓			[:SOURce]:MARKer7:FREQuency?
M8	✓	✓			[:SOURce]:MARKer8:FREQuency?
M9	✓	✓			[:SOURce]:MARKer9:FREQuency?

Table 7-2 8340 and 8757 Code Compatibility (Continued)

Code	Sets Active Function	Comp. with OA / OP	Comp. with UP / DN	Comp. with RB (Knob)	Equivalent SCPI Commands for OA / OP query and UP / DN command

PL	✓	✓	✓	✓	`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]?`
					`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] UP`
					`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] DOWN`

PS	✓	✓			`[:SOURce]:POWer:SPAN?`
RC	✓				*none*
SB	✓	✓			*supported, but no equivalent SCPI command sequence*
SF	✓	✓		✓	`[:SOURce]:FREQuency[:CW]:STEP[:INCRement]?`

SHA1	✓	✓	✓	✓	`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]?`
					`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] UP`
					`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] DOWN`

SHA2	✓	✓		✓	`[:SOURce]:POWer:ALC:LEVel?`

SHA3	✓	✓	✓	✓	`[:SOURce]:POWer:ALC:LEVel?`
					`[:SOURce]:POWer:ATTenuation UP`
					`[:SOURce]:POWer:ATTenuation DOWN`

SHAZ	✓	✓		✓	`[:SOURce]:POWer:ALC:LEVel?`
SHCF	✓	✓		✓	`[:SOURce]:FREQuency[:CW]:STEP[:INCRement]?`

SHC W	✓	✓			`[:SOURce]:FREQuency:STARt?`
					or
					`[:SOURce]:FREQuency:STOP?`

SHFA	✓	✓		✓	`[:SOURce]:FREQuency:MULTiplier?`
SHFB	✓	✓		✓	`[:SOURce]:FREQuency:OFFSet?`

SHPL	✓	✓	✓	✓	`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]:STEP [:INCRement]?`
					
					`[:SOURce]:POWer:ATTenuation UP`
					`[:SOURce]:POWer:ATTenuation DOWN`

SHPS	✓	✓	✓	✓	`[:SOURce]:POWer:ALC:LEVel?`
					`[:SOURce]:POWer:ATTenuation UP`
					`[:SOURce]:POWer:ATTenuation DOWN`

Table 7-2 8340 and 8757 Code Compatibility (Continued)

Code	Sets Active Function	Comp. with OA / OP	Comp. with UP / DN	Comp. with RB (Knob)	Equivalent SCPI Commands for OA / OP query and UP / DN command

SHRF	✓	✓	✓	✓	`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]?`
					`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] UP`
					`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude] DOWN`

SHSL	✓	✓			`[:SOURce]:POWer:ATTenuation?`
SHSN	✓	✓		✓	`[:SOURce]:SWEep:POINts?`
SL	✓	✓			`[:SOURce]:POWer:SLOPe?`
SM	✓	✓			`[:SOURce]:FREQuency:MANual?`
SN	✓	✓		✓	`[:SOURce]:SWEep:POINts?`
SP	✓	✓		✓	`[:SOURce]:POWer[:LEVel][:IMMediate][:AMPLitude]:STEP [:INCRement]?`
ST	✓	✓			`[:SOURce]:SWEep:TIME?`
SV	✓				*none*
TL	✓	✓			`[:SOURce]:SWEep:TIME:LLIMit?`

836xxB / L Compatible SCPI Commands

Table 7-3 is a comprehensive list of 836xxB/L SCPI commands arranged by subsystem. Commands that are supported by the PSG are identified, in addition to commands that are unsupported. Use the legend within the table to determine command compatibility.

The preset state of the PSG differs from that of the 836xxB/L. The RF output and sweep are turned off in the PSG, while in the 836xxB/L, these parameters are turned on. To optimize the benefit of using 836xxB/L compatible commands with a PSG, set up a user-defined preset state, emulating the preset state of the 836xxB/L.

NOTE Some of the PSG supported commands are a subset of the 836xxB/L commands. When this occurs, the syntax supported by the PSG is shown in addition to the syntax that is not supported.

Table 7-3 836xxB / L SCPI Commands

Y= Supported by PSG **N= Not supported by PSG**	**83620B & 83640B**	**83620L & 83640L**
IEEE Common Commands		
`*CLS`	Y	Y
`*ESE <data>`	Y	Y
`*ESE?`	Y	Y
`*ESR?`	Y	Y
`*IDN?`[a]	Y	Y
`*LRN?`	N	N
`*OPC`	Y	Y
`*OPC?`	Y	Y
`*OPT?`	N	N
`*RCL <reg_num>`	Y	Y

Table 7-3 836xxB / L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
*RST	Y	Y
*SAV <reg_num>	Y	Y
*SRE <data>	Y	Y
*SRE?	Y	Y
*STB?	Y	Y
*TRG	Y	Y
*TST?	Y	Y
*WAI	Y	Y
Abort Subsystem		
:ABORt	Y	Y
Amplitude Modulation Subsystem		
:AM[:DEPTh] <num>[PCT]|MAXimum|MINimum|<num>DB	Y	
:AM[:DEPTh]? [MAXimum|MINimum]	Y	
:AM:INTernal:FREQuency <num>[<freq suffix>]|MAXimum|MINimum	Y	
:AM:INTernal:FREQuency? [MAXimum|MINimum]	Y	
:AM:INTernal:FUNCtion SINusoid|SQUare|TRIangle|RAMP|NOISe	Y	
:AM:INTernal:FUNCtion?	Y	
:AM:SOURce INTernal|EXTernal	Y	
:AM:SOURce?	Y	
:AM:MODE DEEP|NORMal	Y	
:AM:MODE?	Y	

Table 7-3 836xxB/L SCPI Commands

Y= Supported by PSG **N= Not supported by PSG**	**83620B & 83640B**	**83620L & 83640L**
:AM:STATe ON|OFF|1|0	Y	
:AM:STATe?	Y	
:AM:TYPE LINear|EXPonential	Y	
:AM:TYPE?	Y	
Calibration Subsystem		
:CALibration:AM:AUTO ON|OFF|1|0	N	
:CALibration:AM:AUTO?	N	
:CALibration:AM[:EXECute]	N	
:CALibration:PEAKing:AUTO ON|OFF|1|0	N	N
:CALibration:PEAKing:AUTO?	N	N
:CALibration:PEAKing[:EXECute]	N	N
:CALibration:PMETer:DETector:INITiate? IDETector|DIODe	N	N
:CALibration:PMETer:DETector:NEXT? <num>[<lvl suffix>]	N	N
:CALibration:PMETer:FLATness:INITiate? USER|DIODe|PMETER| MMHead	N	N
:CALibration:PMETer:FLATness:NEXT? <value>[<lvl suffix>]	N	N
:CALibration:SPAN:AUTO ON|OFF|1|0	N	N
:CALibration:SPAN:AUTO?	N	N
:CALibration:SPAN[:EXECute]	N	N
:CALibration:TRACk	N	N

Table 7-3 836xxB/L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
Correction Subsystem		
:CORRection:ARRay[i]{<value>[DB]}	N	N
:CORRection:ARRay[i]?	N	N
:CORRection:FLATness {<num>[freq suffix],<num>[DB]}2*801	N	N
:CORRection:FLATness?	Y	Y
:CORRection:SOURce[i] ARRay|FLATness	N	N
:CORRection:SOURce[i]?	N	N
:CORRection:FLATness:POINts? [MAXimum|MINimum]	Y	Y
:CORRection[:STATe] ON|OFF|1|0	Y	Y
:CORRection[:STATe]?	Y	Y
Diagnostics Subsystem		
:DIAGnostics:ABUS? <value>	N	N
:DIAGnostics:ABUS:AVERage <value>	N	N
:DIAGnostics:ABUS:AVERage?	N	N
:DIAGnostics:ABUS:STATus?	N	N
:DIAGnostics:INSTrument:PMETer:ADDRess <value>	N	N
:DIAGnostics:INSTrument:PMETer:ADDRess?	N	N
:DIAGnostics:INSTrument:PRINter:ADDRess <value>	N	N
:DIAGnostics:INSTrument:PRINter:ADDRess?	N	N
:DIAGnostics:IORW <value>,<value>	N	N
:DIAGnostics:IORW? <value>	N	N

Table 7-3 836xxB/L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
:DIAGnostics:OUTPut:FAULt?	N	N
:DIAGnostics:RESult?	N	N
:DIAGnostics:TEST:CONTinue	N	N
:DIAGnostics:TEST:DATA:DESC?	N	N
:DIAGnostics:TEST:DATA:MAXimum?	N	N
:DIAGnostics:TEST:DATA:MINimum?	N	N
:DIAGnostics:TEST:DATA:VALue?	N	N
:DIAGnostics:TEST:DISable {<num>}1*?|ALL	N	N
:DIAGnostics:TEST:ENABle {<num>}1*?|ALL	N	N
:DIAGnostics:TEST[:EXECute] <value>	N	N
:DIAGnostics:TEST:LOG:SOURce ALL|FAIL	N	N
:DIAGnostics:TEST:LOG:SOURce?	N	N
:DIAGnostics:TEST:LOG[:STATe]?	N	N
:DIAGnostics:TEST:LOG[:STATe] ON|OFF|1|0	N	N
:DIAGnostics:TEST:LOOP ON|OFF|1|0	N	N
:DIAGnostics:TEST:LOOP?	N	N
:DIAGnostics:TEST:NAME? [<value>]	N	N
:DIAGnostics:TEST:POINts?	N	N
:DIAGnostics:TEST:RESult? [<value>]	N	N
:DIAGnostics:TINT? <value>	N	N

Table 7-3 836xxB / L SCPI Commands

Y= Supported by PSG **N= Not supported by PSG**	**83620B & 83640B**	**83620L & 83640L**
Display Subsystem		
:DISPlay[:STATe] ON|OFF|1|0	Y	Y
:DISPlay[:STATe]?	Y	Y
Frequency Modulation Subsystem		
:FM:COUPling AC|DC	Y	
:FM:COUPling?	Y	
:FM[:DEViation] <val><unit>|MAXimum|MINimum	Y	
:FM[:DEViation]? [MAXimum|MINimum]	Y	
:FM:FILTer:HPASs <num>[<freq suffix>]|MAXimum|MINimum	N	
:FM:FILTer:HPASs? [MAXimum|MINimum]	N	
:FM:INTernal:FREQuency <num>[<freq suffix>]|MAXimum|MINimum	Y	
:FM:INTernal:FREQuency? [MAXimum|MINimum]	Y	
:FM:INTernal:FUNCtion SINusoid|SQUare|TRIangle|RAMP|NOISe	Y	
:FM:INTernal:FUNCtion?	Y	
:FM:SOURce INTernal|EXTernal	Y	
:FM:SOURce?	Y	
:FM:SENSitivity <val><freq suffix/V>|MAXimum|MINimum	Y	
:FM:SENSitivity? [MAXimum|MINimum]	Y	
:FM:STATe ON|OFF|1|0	Y	
:FM:STATe?	Y	

Table 7-3 836xxB/L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
Frequency Subsystem		
:FREQuency:CENTer <num>[<freq suffix>]|MAXimum|MINimum| UP|DOWN	Y	Y
:FREQuency:CENTer? [MAXimum|MINimum]	Y	Y
:FREQuency[:CW|:FIXed] <num>[<freq suffix>]|MAXimum| MINimum|UP|DOWN	Y	Y
:FREQuency[:CW]? [MAXimum|MINimum]	Y	Y
:FREQuency[:FIXed]? [MAXimum|MINimum]	Y	Y
:FREQuency[:CW]:AUTO ON|OFF|1|0	N	N
:FREQuency[:CW]:AUTO?	N	N
:FREQuency[:FIXed]:AUTO ON|OFF|1|0	N	N
:FREQuency[:FIXed]:AUTO?	N	N
:FREQuency:MANual <num>[freq suffix]|MAXimum|MINimum|UP| DOWN	N	N
:FREQuency:MANual? [MAXimum|MINimum]	N	N
:FREQuency:MODE FIXed|CW|SWEep|LIST	Y	Y
:FREQuency:MODE?	Y	Y
:FREQuency:MULTiplier <num>|MAXimum|MINimum[b]	Y	Y
:FREQuency:MULTiplier? [MAXimum|MINimum]	Y	Y
:FREQuency:MULTiplier:STATe ON|OFF|1|0	N	N
:FREQuency:MULTiplier:STATe?	N	N
:FREQuency:OFFSet <num>|MAXimum|MINimum	Y	Y

Table 7-3 836xxB/L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
:FREQuency:OFFSet? [MAXimum|MINimum]	Y	Y
:FREQuency:OFFSet:STATe ON|OFF|1|0	Y	Y
:FREQuency:OFFSet:STATe?	Y	Y
:FREQuency:SPAN <num>[<freq suffix>]|MAXimum|MINimum|UP| DOWN	Y	Y
:FREQuency:SPAN? [MAXimum|MINimum]	Y	Y
:FREQuency:STARt <num>[<freq suffix>]|MAXimum|MINimum|UP| DOWN	Y	Y
:FREQuency:STARt? [MAXimum|MINimum]	Y	Y
:FREQuency:STEP:AUTO ON|OFF|1|0	Y	Y
:FREQuency:STEP:AUTO?	Y	Y
:FREQuency:STEP[:INCRement] <num>[<freq suffix>]|MAXimum| MINimum	Y	Y
:FREQuency:STEP[:INCRement]?	Y	Y
:FREQuency:STOP <num>[<freq suffix>]|MAXimum|MINimum|UP| DOWN	Y	Y
:FREQuency:STOP? [MAXimum|MINimum]	Y	Y
Initiate Subsystem		
:INITiate:CONTinuous ON|OFF|1|0	Y	Y
:INITiate:CONTinuous?	Y	Y
:INITiate[:IMMediate]	Y	Y

Table 7-3 836xxB / L SCPI Commands

<table>
<tr><th>Y= Supported by PSG
N= Not supported by PSG</th><th>83620B & 83640B</th><th>83620L & 83640L</th></tr>
<tr><td colspan="3">List Subsystem</td></tr>
<tr><td>:LIST:DWELl {<num>[<time suffix>]|MAXimum|MINimum}</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:DWELl? [MAXimum|MINimum]</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:DWELl:POINts? [MAXimum|MINimum]</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:FREQuency {<value>[<freq suffix>]|MAXimum|MINimum}</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:FREQuency?</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:FREQuency:POINts? [MAXimum|MINimum]</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:MANual <num></td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:MANual?</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:MODE AUTO|MANual</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:MODE?</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST[:POWer]:CORRection {<value>[DB]|MAXimum|MINimum}</td><td>N</td><td>N</td></tr>
<tr><td>:LIST[:POWer]:CORRection?</td><td>N</td><td>N</td></tr>
<tr><td>:LIST[:POWer]:CORRection:POINts? [MAXimum|MINimum]</td><td>N</td><td>N</td></tr>
<tr><td>:LIST:TRIGger:SOURce IMMediate|BUS|EXTernal</td><td>Y</td><td>Y</td></tr>
<tr><td>:LIST:TRIGger:SOURce?</td><td>Y</td><td>Y</td></tr>
<tr><td colspan="3">Marker Subsystem</td></tr>
<tr><td>:MARKer[n]:AMPLitude[:STATe] ON|OFF|1|0</td><td>N</td><td>N</td></tr>
<tr><td>:MARKer[n]:AMPLitude[:STATe]?</td><td>N</td><td>N</td></tr>
<tr><td>:MARKer[n]:AMPLitude:VALue <value>[DB]|MAXimum|MINimum</td><td>N</td><td>N</td></tr>
</table>

Table 7-3 **836xxB/L SCPI Commands**

Y= Supported by PSG **N= Not supported by PSG**	**83620B & 83640B**	**83620L & 83640L**
:MARKer[n]:AMPLitude:VALue? [MAXimum|MINimum]	N	N
:MARKer[n]:AOFF	N	N
:MARKer[n]:DELTa? <value>,<value>	N	N
:MARKer[n]:FREQuency <value>[<freq suffix>]|MAXimum|MINimum	N	N
:MARKer[n]:FREQuency? [MAXimum|MINimum]	N	N
:MARKer[n]:MODE FREQuency|DELTa	N	N
:MARKer[n]:MODE?	N	N
:MARKer[n]:REFerence <n>	N	N
:MARKer[n]:REFerence?	N	N
:MARKer[n][:STATe] ON|OFF|1|0	N	N
:MARKer[n][:STATe]?	N	N
Measure Subsystem		
:MEASure:AM?	N	
:MEASure:FM?	N	
Modulation Subsystem		
:MODulation:OUTPut:SOURce AM|FM	N	
:MODulation:OUTPut:SOURce?	N	
:MODulation:OUTPut:STATe ON|OFF|1|0	Y	
:MODulation:OUTPut:STATe?	Y	
:MODulation:STATe?	Y	

Table 7-3 836xxB/L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
Power Subsystem		
:POWer:ALC:BANDwidth|:BWIDth <value>[<freq suffix>]|MAXimum|MINimum	Y	Y
:POWer:ALC:BANDwidth?|:BWIDth? [MAXimum|MINimum]	Y	Y
:POWer:ALC:BANDwidth|:BWIDth:AUTO ON|OFF|1|0	Y	Y
:POWer:ALC:BANDwidth|:BWIDth:AUTO?	Y	Y
:POWer:ALC:CFACtor <value>[DB]|MAXimum|MINimum|UP|DOWN	Y	Y
:POWer:ALC:CFACtor? [MINimum|MAXimum]	Y	Y
:POWer:ALC:SOURce PMETer :POWer:ALC:SOURce INTernal|DIODe|MMHead	N Y	N Y
:POWer:ALC:SOURce?	Y	Y
:POWer:ALC[:STATe] ON|OFF|1|0	Y	Y
:POWer:ALC[:STATe]?	Y	Y
:POWer:AMPLifier:STATE ON|OFF|1|0	N	N
:POWer:AMPLifier:STATE?	N	N
:POWer:AMPLifier:STATE:AUTO ON|OFF|1|0	N	N
:POWer:AMPLifier:STATE:AUTO?	N	N
:POWer:ATTenuation <num>[DB]|MAXimum|MINimum|UP|DOWN	Y	Y
:POWer:ATTenuation? [MAXimum|MINimum]	Y	Y
:POWer:ATTenuation:AUTO ON|OFF|1|0	Y	Y
:POWer:ATTenuation:AUTO?	Y	Y

Table 7-3 836xxB/L SCPI Commands

<table>
<tr><th>Y= Supported by PSG
N= Not supported by PSG</th><th>83620B & 83640B</th><th>83620L & 83640L</th></tr>
<tr><td>:POWer:CENTer <num>[<lvl suffix>]|MAXimum|MINimum|UP|DOWN</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:CENTer? [MAXimum|MINimum]</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer[:LEVel] <num>[<lvl suffix>]|MAXimum|MINimum|UP|DOWN</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer[:LEVel]? [MAXimum|MINimum]</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:MODE FIXed|SWEep</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:MODE?</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:OFFSet <num>[DB]|MAXimum|MINimum|UP|DOWN</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:OFFSet? [MAXimum|MINimum]</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:OFFSet:STATe ON|1[c]
:POWer:OFFSet:STATe OFF|0[d]</td><td>N
Y</td><td>N
Y</td></tr>
<tr><td>:POWer:OFFSet:STATe?</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:RANGe <value>[<lvl suffix>]|MAXimum|MINimum|UP|DOWN</td><td>N</td><td>N</td></tr>
<tr><td>:POWer:RANGe?</td><td>N</td><td>N</td></tr>
<tr><td>:POWer:SEARch ON|OFF|1|0|ONCE</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:SEARch?</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:SLOPe <value>[DB/<freq suffix>]|MIN|MAX|UP|DOWN</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:SLOPe? [MAXimum|MINimum]</td><td>Y</td><td>Y</td></tr>
<tr><td>:POWer:SLOPe:STATe ON|OFF|1|0</td><td>Y</td><td>Y</td></tr>
</table>

Table 7-3 836xxB/L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
:POWer:SLOPe:STATe?	Y	Y
:POWer:SPAN <value>[DB]|MAXimum|MINimum|UP|DOWN	Y	Y
:POWer:SPAN? [MAXimum|MINimum]	Y	Y
:POWer:STARt <val><unit>|MAXimum|MINimum|UP|DOWN	Y	Y
:POWer:STARt? [MAXimum|MINimum]	Y	Y
:POWer:STATe ON|OFF|1|0	Y	Y
:POWer:STATe?	Y	Y
:POWer:STEP:AUTO ON|OFF|1|0	Y	Y
:POWer:STEP:AUTO?	Y	Y
:POWer:STEP[:INCRement] <num>[DB]|MAXimum|MINimum	Y	Y
:POWer:STEP[:INCRement]? [MAXimum|MINimum]	Y	Y
:POWer:STOP <val><unit>|MAXimum|MINimum|UP|DOWN	Y	Y
:POWer:STOP? [MAXimum|MINimum]	Y	Y
Pulse Modulation Subsystem		
:PULM:EXTernal:DELay <value>[<time suffix>]|MAXimum|MINimum	N	
:PULM:EXTernal:DELay? [MAXimum|MINimum]	N	
:PULM:EXTernal:POLarity NORMal|INVerted	Y	
:PULM:EXTernal:POLarity?	Y	
:PULM:INTernal:FREQuency <num>[<freq suffix>]|MAXimum|MINimum	Y	
:PULM:INTernal:FREQuency? [MAXimum|MINimum]	Y	

Table 7-3 836xxB/L SCPI Commands

Y= Supported by PSG **N= Not supported by PSG**	**83620B & 83640B**	**83620L & 83640L**
`:PULM:INTernal:GATE ON|OFF|1|0`	N	
`:PULM:INTernal:GATE?`	N	
`:PULM:INTernal:PERiod <num>[<time suffix>]|MAXimum|MINimum`	Y	
`:PULM:INTernal:PERiod? [MAXimum|MINimum]`	Y	
`:PULM:INTernal:TRIGger:SOURce INTernal|EXTernal`	Y	
`:PULM:INTernal:TRIGger:SOURce? [INTernal|EXTernal]`	Y	
`:PULM:INTernal:WIDTh <num>[<time suffix>]|MAXimum|MINimum`	Y	
`:PULM:INTernal:WIDTh? [MAXimum|MINimum]`	Y	
`:PULM:SLEW <value>[<time suffix>]|MAXimum|MINimum`	N	
`:PULM:SLEW? [MAXimum|MINimum]`	N	
`:PULM:SLEW:AUTO ON|OFF|1|0`	N	
`:PULM:SLEW:AUTO?`	N	
`:PULM:SOURce SCALar` `:PULM:SOURce INTernal|EXTernal`	N Y	
`:PULM:SOURce?`	Y	
`:PULM:STATe ON|OFF|1|0`	Y	
`:PULM:STATe?`	Y	
Pulse Subsystem		
`:PULSe:FREQuency <num>[<freq suffix>]|MAXimum|MINimum`	Y	
`:PULSe:FREQuency? [MAXimum|MINimum]`	Y	
`:PULSe:PERiod <num>[<time suffix>]|MAXimum|MINimum`	Y	

Table 7-3 **836xxB/L SCPI Commands**

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
`:PULSe:PERiod? [MAXimum|MINimum]`	Y	
`:PULSe:WIDTh <num>[<time suffix>]|MAXimum|MINimum`	Y	
`:PULSe:WIDTh? [MAXimum|MINimum]`	Y	
Reference Oscillator Subsystem		
`:ROSCillator:SOURce?`	Y	Y
`:ROSCillator:SOURce:AUTO ON|OFF|1|0`	Y	Y
`:ROSCillator:SOURce:AUTO?`	Y	Y
`:ROSCillator:SOURce INTernal|EXTernal|NONE`	Y	Y
Status Subsystem		
`:STATus:OPERation:CONDition?`	Y	Y
`:STATus:OPERation:ENABle <value>`	Y	Y
`:STATus:OPERation:ENABle?`	Y	Y
`:STATus:OPERation[:EVENt]?`	Y	Y
`:STATus:OPERation:NTRansition <value>`	Y	Y
`:STATus:OPERation:NTRansition?`	Y	Y
`:STATus:OPERation:PTRansition <value>`	Y	Y
`:STATus:OPERation:PTRansition?`	Y	Y
`:STATus:PRESet`	Y	Y
`:STATus:QUEStionable:CONDition?`	Y	Y
`:STATus:QUEStionable:ENABle <value>`	Y	Y
`:STATus:QUEStionable:ENABle?`	Y	Y
`:STATus:QUEStionable[:EVENt]?`	Y	Y

Table 7-3 836xxB / L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
:STATus:QUEStionable:NTRansition <value>	Y	Y
:STATus:QUEStionable:NTRansition?	Y	Y
:STATus:QUEStionable:PTRansition <value>	Y	Y
:STATus:QUEStionable:PTRansition?	Y	Y
Sweep Subsystem		
:SWEep:CONTrol:STATe ON|OFF|1|0	N	N
:SWEep:CONTrol:STATe?	N	N
:SWEep:CONTrol:TYPE MASTer|SLAVe	N	N
:SWEep:CONTrol:TYPE?	N	N
:SWEep:DWELl <num>[<time suffix>]|MAXimum|MINimum	Y	Y
:SWEep:DWELl? [MAXimum|MINimum]	Y	Y
:SWEep:DWELl:AUTO ON|OFF|1|0	N	N
:SWEep:DWELl:AUTO?	N	N
:SWEep:GENeration STEPped|ANALog	N	N
:SWEep:GENeration?	N	N
:SWEep:MANual:POINt <num>|MAXimum|MINimum	Y	Y
:SWEep:MANual:POINt? [MAXimum|MINimum]	Y	Y
:SWEep:MANual[:RELative] <value>	N	N
:SWEep:MANual[:RELative]?	N	N
:SWEep:MARKer:STATe ON|OFF|1|0	N	N
:SWEep:MARKer:STATe?	N	N
:SWEep:MARKer:XFER	N	N

Table 7-3 836xxB / L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
:SWEep:MODE AUTO|MANual	Y	Y
:SWEep:MODE?	Y	Y
:SWEep:POINts <num>|MAXimum|MINimum	Y	Y
:SWEep:POINts? [MAXimum|MINimum]	Y	Y
:SWEep:STEP <value>[<freq suffix>]|MAXimum|MINimum	N	N
:SWEep:STEP? [MAXimum|MINimum]	N	N
:SWEep:TIME <value>[<time suffix>]|MAXimum|MINimum	N	N
:SWEep:TIME? [MAXimum|MINimum]	N	N
:SWEep:TIME:AUTO ON|OFF|1|0	N	N
:SWEep:TIME:AUTO?	N	N
:SWEep:TIME:LLIMit <value>[<time suffix>]|MAXimum|MINimum	N	N
:SWEep:TIME:LLIMit? [MAXimum|MINimum]	N	N
:SWEep:TRIGger:SOURce IMMediate|BUS|EXTernal	Y	Y
:SWEep:TRIGger:SOURce?	Y	Y
System Subsystem		
:SYSTem:ALTernate <value>|MAXimum|MINimum	N	N
:SYSTem:ALTernate? [MAXimum|MINimum]	N	N
:SYSTem:ALTernate:STATe ON|OFF|1|0	N	N
:SYSTem:ALTernate:STATe?	N	N
:SYSTem:COMMunicate:GPIB:ADDRess <number>	Y	Y
:SYSTem:DUMP:PRINter?	N	N

Table 7-3 836xxB / L SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
:SYSTem:ERRor?	Y	Y
:SYSTem:LANGuage CIIL|COMPatible :SYSTem:LANGuage SCPI	N Y	N Y
:SYSTem:MMHead:SELect:AUTO ON|OFF|1|0	Y	Y
:SYSTem:MMHead:SELect:AUTO?	Y	Y
:SYSTem:MMHead:SELect FRONt|REAR|NONE[e]	Y	Y
:SYSTem:MMHead:SELect?	Y	Y
:SYSTem:PRESet[:EXECute]	Y	Y
:SYSTem:PRESet:SAVE	Y	Y
:SYSTem:PRESet:TYPE FACTory|USER	Y	Y
:SYSTem:PRESet:TYPE?	Y	Y
:SYSTem:SECurity:COUNt <value>[fg]	Y	Y
:SYSTem:SECurity:COUNt? [MINimum|MAXimum]	Y	Y
:SYSTem:SECurity[:STATe] ON|OFF|1|0[e]	Y	Y
:SYSTem:SECurity[:STATe]?	Y	Y
:SYSTem:VERSion?	Y	Y
Trigger Subsystem		
:TRIGger[:IMMediate]	Y	Y
:TRIGger:ODELay <value>[time suffix]|MAXimum|MINimum	N	N
:TRIGger:ODELay? [MAXimum|MINimum]	N	N
:TRIGger:SOURce IMMediate|BUS|EXTernal	Y	Y
:TRIGger:SOURce?	Y	Y

Table 7-3 **836xxB / L SCPI Commands**

Y= Supported by PSG N= Not supported by PSG	83620B & 83640B	83620L & 83640L
Tsweep Subsystem		
`:TSWeep`	N	N
Unit Subsystem		
`:UNIT:AM DB|PCT`	N	
`:UNIT:AM?`	N	
`:UNIT:POWer {<lvl suffix>}`	Y	Y
`:UNIT:POWer?`	Y	Y

a. The identification information can be modified for the PSG to reflect the signal generator that is being replaced. Refer to ":SYSTem:IDN" on page 376 for more information.
b. A multiplier of zero is not allowed.
c. The PSG will accept this command, but it has no effect.
d. This command resets the power offset level to 0dBm. It does not turn off or disable the power offset feature.
e. Since the PSG does not have a front panel millimeter head (source module) interface connector, the "FRONT" suffix defaults to the rear connector.
f. Flash memory allows only a limited number of "writes and erasures", excessive use of this command will reduce the memory lifetime.
g. This command can take several hours to execute because the PSG memory size is much larger than the HP 836xx memory.

8373xB and 8371xB Compatible SCPI Commands

Table 7-4 is a comprehensive list of 8373xB and 8371xB SCPI commands arranged by subsystem. Commands that are supported by the PSG are identified, in addition to commands that are unsupported. Use the legend within the table to determine command compatibility.

NOTE Some of the PSG supported commands are subsets of the 8373xB and 8371xB commands. When this occurs, the syntax supported by the PSG is shown in addition to the syntax that is not supported.

Table 7-4 8373xB and 8371xB SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83731B & 83732B	83711B & 83712B
IEEE Common Commands		
`*CLS`	Y	Y
`*DMC`	N	N
`*EMC`	N	N
`*EMC?`	N	N
`*ESE <data>`	Y	Y
`*ESE?`	Y	Y
`*ESR?`	Y	Y
`*GMC?`	N	N
`*IDN?`[a]	Y	Y
`*LMC?`	N	N
`*LRN?`	N	N
`*OPC`	Y	Y
`*OPC?`	Y	Y
`*OPT?`	N	N

Table 7-4 8373xB and 8371xB SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83731B & 83732B	83711B & 83712B
*PMC	N	N
*PSC	Y	Y
*PSC?	Y	Y
*RCL <reg_num>	Y	Y
*RMC	N	N
*RST	Y	Y
*SAV <reg_num>	Y	Y
*SRE <data>	Y	Y
*SRE?	Y	Y
*STB?	Y	Y
*TST?	Y	Y
*WAI	Y	Y
Abort Subsystem		
:ABORt	Y	
Amplitude Modulation Subsystem		
[:SOURce]:AM[:DEPTh] <val><unit>[b]	Y	
[:SOURce]:AM[:DEPTh] <num>[<PCT>]|<num>DB	Y	
[:SOURce]:AM[:DEPTh]:STEP[:INCRement] incr|MINimum| MAXimum|DEFault	Y	
[:SOURce]:AM:INTernal:FREQuency <num>[<freq suffix>] incr| MINimum|MAXimum|DEFault	Y	
[:SOURce]:AM:INTernal:FREQuency:STEP[:INCRement]	Y	
[:SOURce]:AM:INTernal:FUNCtion SINusoid|SQUare|TRIangle| RAMP|NOISe|UNIForm|GAUSsian	Y	
[:SOURce]:AM:SENSitivity <val>|MIN|MAX|DEF	N	

Table 7-4 8373xB and 8371xB SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83731B & 83732B	83711B & 83712B
[:SOURce]:AM:SOURce FEED [:SOURce]:AM:SOURce INTernal|EXTernal	N Y	
[:SOURce]:AM:SOURce?	Y	
[:SOURce]:AM:STATe ON|OFF	Y	
[:SOURce]:AM:STATe?	Y	
[:SOURce]:AM:TYPE LINear|EXPonential	Y	
[:SOURce]:AM:TYPE?	Y	
Display Subsystem		
:DISPlay[:WINDow][:STATe] ON|OFF|1|0	Y	Y
:DISPlay[:WINDow][:STATe]?	Y	Y
Initiate Subsystem		
:INITiate:CONTinuous ON|OFF|1|0	Y	
:INITiate:CONTinuous?	Y	
Correction Subsystem		
[:SOURce]:CORRection:FLATness[:DATA] <freq>,<corr.>,... <freq>,<corr.>	Y	Y
[:SOURce]:CORRection:FLATness:POINts <points>	Y	Y
[:SOURce]:CORRection[:STATe] ON|OFF	Y	Y
[:SOURce]:CORRection[:STATe]?	Y	Y
[:SOURce]:CORRection:CSET[:SELect] tableno	N	N
[:SOURce]:CORRection:CSET[:SELect]?	N	N
[:SOURce]:CORRection:CSET:STATe ON|OFF|1|0	N	N
[:SOURce]:CORRection:CSET:STATe?	N	N
Frequency Modulation Subsystem		
[:SOURce]:FM:COUPling AC|DC	Y	

Table 7-4 8373xB and 8371xB SCPI Commands

<table>
<tr><th>Y= Supported by PSG
N= Not supported by PSG</th><th>83731B & 83732B</th><th>83711B & 83712B</th></tr>
<tr><td>[:SOURce]:FM:COUPling?</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:FM[:DEViation] <val><unit></td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:FM[:DEViation]:STEP[:INCRement] <val>
[<freq suffix>]</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:FM:INTernal:FREQuency <num>[<freq suffix>]</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:FM:INTernal:FREQuency:STEP[:INCRement] incr|
MINimum|MAXimum|DEFault</td><td>N</td><td></td></tr>
<tr><td>[:SOURce]:FM:INTernal:FUNCtion SINusoid|SQUAre|TRIAngle|
RAMP|UNIForm|GAUSsian</td><td>N</td><td></td></tr>
<tr><td>[:SOURce]:FM:SENSitivity?</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:FM:SOURce FEED
[:SOURce]:FM:SOURce INTernal|EXTernal</td><td>N
Y</td><td></td></tr>
<tr><td>[:SOURce]:FM:STATe ON|OFF|1|0</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:FM:STATe?</td><td>Y</td><td></td></tr>
<tr><td colspan="3">Frequency Subsystem</td></tr>
<tr><td>[:SOURce]:FREQuency[:CW|:FIXed] <num>[<freq suffix>]|UP|
DOWN|DEFault</td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:FREQuency[:CW|:FIXed] [MAXimum|MINimum|DEFault]</td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:FREQuency[:CW|:FIXed]:STEP <val><unit></td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:FREQuency[:CW|:FIXed]:STEP?</td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:FREQuency:MULTiplier <val>|UP|DOWN|DEFault[c]</td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:FREQuency:MULTiplier?</td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:FREQuency:MULTiplier:STEP[:INCRement] incr|
MINimum|MAXimum|DEFault</td><td>N</td><td>N</td></tr>
<tr><td>[:SOURce]:FREQuency:MULTiplier:STEP[:INCRement]?</td><td>N</td><td>N</td></tr>
<tr><td colspan="3">Memory Subsystem</td></tr>
</table>

Table 7-4 8373xB and 8371xB SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83731B & 83732B	83711B & 83712B
:MEMory:CATalog[:ALL]?	Y	Y
:MEMory:CATalog:TABLe?	N	N
:MEMory:CATalog:MACRo	N	N
:MEMory:RAM:INITialize	N	N
:MEMory:TABLe:FREQuency freq,...freq|MINimum|MAXimum	N	N
:MEMory:TABLe:FREQuency? MINimum|MAXimum	N	N
:MEMory:TABLe:FREQuency:POINts?	N	N
:MEMory:TABLe:LOSS[:MAGNitude] cf,...cf|MINimum|MAXimum	N	N
:MEMory:TABLe:LOSS[:MAGNitude]?	N	N
:MEMory:TABLe:LOSS[:MAGNitude]:POINts?	N	N
:MEMory:TABLe:SELect tableno	N	N
:MEMory:TABLe:SELect?	N	N
Modulation Subsystem		
[:SOURce]:MODulation:AOFF	Y	
[:SOURce]:MODulation:STATe ON|OFF	N	
[:SOURce]:MODulation:STATe?	Y	
Output Subsystem		
:OUTPut:IMPedance?	N	N
:OUTPut:PROTection[:STATe] ON|OFF	Y	Y
:OUTPut:PROTection[:STATe]?	Y	Y
:OUTPut[:STATe] ON|OFF|1|0	Y	Y
:OUTPut[:STATe]?	Y	Y
Phase Modulation Subsystem		
[:SOURce]:PM:COUPling AC|DC	Y	

Table 7-4 8373xB and 8371xB SCPI Commands

<table>
<tr><th>Y= Supported by PSG
N= Not supported by PSG</th><th>83731B & 83732B</th><th>83711B & 83712B</th></tr>
<tr><td>[:SOURce]:PM[:DEViation] <val><unit></td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:PM[:DEViation]:STEP[:INCRement]</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:PM:INTernal:FREQuency <val><unit></td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:PM:INTernal:FREQuency:STEP[:INCRement]</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:PM:INTernal:FUNCtion SINusoid|SQUare|TRIAngle|
RAMP|UNIForm|GAUSsian</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:PM:RANGe AUTO|LOW|HIGH</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:PM:SENSitivity sens|MINimum|MAXimum|DEFault</td><td>N</td><td></td></tr>
<tr><td>[:SOURce]:PM:SOURce INTernal|FEED|EXTernal[d]</td><td>Y</td><td></td></tr>
<tr><td>[:SOURce]:PM:STATe ON|OFF|1|0</td><td>Y</td><td></td></tr>
<tr><td colspan="3">Power Subsystem</td></tr>
<tr><td>[:SOURce]:POWer:ALC:PMETer pmeter|MINimum|MAXimum|DEFault</td><td>N</td><td>N</td></tr>
<tr><td>[:SOURce]:POWer:ALC:PMETer?</td><td>N</td><td>N</td></tr>
<tr><td>[:SOURce]:POWer:ALC:PMETer:STEP incr|MINimum|MAXimum|
DEFault</td><td>N</td><td>N</td></tr>
<tr><td>[:SOURce]:POWer:ALC:PMETer:STEP?</td><td>N</td><td>N</td></tr>
<tr><td>[:SOURce]:POWer:ALC:SOURce PMETer
[:SOURce]:POWer:ALC:SOURce INTernal|DIODe</td><td>N
Y</td><td>N
Y</td></tr>
<tr><td>[:SOURce]:POWer:ALC:SOURce?</td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:POWer:ATTenuation:AUTO ONCE
[:SOURce]:POWer:ATTenuation:AUTO ON|OFF</td><td>N
Y</td><td>N
Y</td></tr>
<tr><td>[:SOURce]:POWer:ATTenuation:AUTO?</td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:POWer[:LEVel] ampl|MINimum|MAXimum|UP|DOWN|
DEFault</td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:POWer[:LEVel]?</td><td>Y</td><td>Y</td></tr>
<tr><td>[:SOURce]:POWer[:LEVel]:STEP incr|MINimum|MAXimum|DEFault</td><td>Y</td><td>Y</td></tr>
</table>

Table 7-4 8373xB and 8371xB SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83731B & 83732B	83711B & 83712B
[:SOURce]:POWer[:LEVel]:STEP?	Y	Y
[:SOURce]:POWer:PROTection:STATe ON|OFF	Y	Y
[:SOURce]:POWer:PROTection:STATe?	Y	Y
Pulse Modulation Subsystem		
[:SOURce]:PULM:EXTernal:POLarity NORMal|INVerted	Y	
[:SOURce]:PULM:EXTernal:POLarity?	Y	
[:SOURce]:PULM:SOURce INTernal|EXTernal	Y	
[:SOURce]:PULM:SOURce?	Y	
[:SOURce]:PULM:STATe ON|OFF|1|0	Y	
[:SOURce]:PULM:STATe?	Y	
Pulse Subsystem		
[:SOURce]:PULSe:DELay delay|MINimum|MAXimum|UP|DOWN| DEFault	Y	
[:SOURce]:PULSe:DELay?	Y	
[:SOURce]:PULSe:DELay:STEP <num>[<time suffix>][DEFault]	Y	
[:SOURce]:PULSe:DELay:STEP? [DEFault]	Y	
[:SOURce]:PULSe:DOUBle[:STATE] ON|OFF	N	
[:SOURce]:PULSe:DOUBle[:STATE]?	N	
[:SOURce]:PULSe:FREQuency freq|MINimum|MAXimum|UP|DOWN| DEFault	Y	
[:SOURce]:PULSe:FREQuency?	Y	
[:SOURce]:PULSe:FREQuency:STEP freq|DEFault	Y	
[:SOURce]:PULSe:FREQuency:STEP? [MIN|MAX|DEF]	Y	
[:SOURce]:PULSe:PERiod <num>[<time suffix>]|UP|DOWN	Y	
[:SOURce]:PULSe:PERiod?	Y	

Table 7-4 8373xB and 8371xB SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83731B & 83732B	83711B & 83712B
[:SOURce]:PULSe:PERiod:STEP <num>[<time suffix>]	Y	
[:SOURce]:PULSe:PERiod:STEP?	Y	
[:SOURce]:PULse:TRANSition[:LEADing] SLOW|MEDium|FAST	N	
[:SOURce]:PULSe:TRANsition[:LEADing]?	N	
[:SOURce]:PULSe:TRANsition:STATe ON|OFF	N	
[:SOURce]:PULSe:TRANsition:STATe?	N	
[:SOURce]:PULSe:WIDTh MAXimum|MINimum|UP|DOWN|DEFault	Y	
[:SOURce]:PULSe:WIDTh? [MAXimum|MINimum|DEFault]	Y	
[:SOURce]:PULSe:WIDTh:STEP <num>[<time suffix>]|DEFault	Y	
[:SOURce]:PULSe:WIDTh:STEP? [MINimum|MAXimum|DEFault]	Y	
Reference Oscillator Subsystem		
[:SOURce]:ROSCillator:SOURce?	Y	Y
Status Subsystem		
:STATus:OPERation:CONDition?	Y	Y
:STATus:OPERation:ENABle <value>	Y	Y
:STATus:OPERation:ENABle?	Y	Y
:STATus:OPERation[:EVENt]?	Y	Y
:STATus:OPERation:NTRansition <value>	Y	Y
:STATus:OPERation:NTRansition?	Y	Y
:STATus:OPERation:PTRansition <value>	Y	Y
:STATus:OPERation:PTRansition?	Y	Y
:STATus:PRESet	Y	Y
:STATus:QUEStionable:CONDition?	Y	Y
:STATus:QUEStionable:ENABle <value>	Y	Y

Table 7-4 8373xB and 8371xB SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83731B & 83732B	83711B & 83712B
:STATus:QUEStionable:ENABle?	Y	Y
:STATus:QUEStionable[:EVENt]?	Y	Y
:STATus:QUEStionable:NTRansition <value>	Y	Y
:STATus:QUEStionable:NTRansition?	Y	Y
:STATus:QUEStionable:PTRansition <value>	Y	Y
:STATus:QUEStionable:PTRansition?	Y	Y
System Subsystem		
:SYSTem:COMMunicate:GPIB:ADDRess <number>	Y	Y
:SYSTem:COMMunicate:GPIB:ADDRess?	Y	Y
:SYSTem:COMMunicate:PMETer:ADDRess	Y	Y
:SYSTem:COMMunicate:PMETer:ADDRess?	Y	Y
:SYSTem:ERRor?	Y	Y
:SYSTem:KEY keycode|MINimum|MAXimum	N	N
:SYSTem:KEY?	N	N
:SYSTem:LANGuage "COMP=8673"|"COMPatibility=8673" :SYSTem:LANGuage "SCPI"	N Y	N Y
:SYSTem:LANGuage?	Y	Y
:SYSTem:PRESet	Y	Y
:SYSTem:VERSion?	Y	Y
Trigger Subsystem		
:TRIGger[:SEQuence|:STARt]:SOURce IMMediate|EXTernal	N	
:TRIGger[:SEQuence|:STARt]:SOURce?	N	
:TRIGger:SEQuence2:STOP:SOURce IMMediate|EXTernal	N	
:TRIGger:SEQuence2:STOP:SOURce?	N	

Table 7-4 8373xB and 8371xB SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83731B & 83732B	83711B & 83712B
:TRIGger:SEQuence2:SLOPe	N	
Unit Subsystem		
:UNIT:FREQuency {<freq suffix>}	Y	Y
:UNIT:FREQuency?	Y	Y
:UNIT:POWer {<lvl suffix>}	Y	Y
:UNIT:POWer?	Y	Y
:UNIT:TIME	N	N
:UNIT:TIME?	N	N
:UNIT:VOLTage {<lvl suffix>}	N	N
:UNIT:VOLTage?	N	N

a. The identification information can be modified for the PSG to reflect the signal generator that is being replaced. Refer to ":SYSTem:IDN" on page 376 for more information.
b. In linear mode, % cannot be used to select percent as the unit. Use PCT to specify percent as the unit.
c. A multiplier of zero is not allowed.
d. If FEED is selected, the query returns INT. FEED and INTernal are synonymous.

8375xB Compatible SCPI Commands

Table 7-5 is a comprehensive list of 83751B and 83752B SCPI commands, arranged by subsystem. Commands that are supported by the PSG are identified, in addition to commands that are unsupported. Use the legend within the table to determine command compatibility.

NOTE Some supported commands require the installation of hardware or firmware options.

Table 7-5 8375xB SCPI Commands

Y= Supported by PSG N= Not supported by PSG	83751B & 83752B
IEEE Common Commands	
`*CLS`	Y
`*DMC`	N
`*EMC`	N
`*EMC?`	N
`*ESE <value>`	Y
`*ESE?`	Y
`*ESR?`	Y
`*GMC? <label>`	N
`*IDN?`	Y
`*LMC?`	N
`*LRN?`	N
`*OPC`	Y
`*OPC?`	Y
`*OPT?`	N

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG N= Not supported by PSG	83751B & 83752B
`*PMC`	N
`*PSC ON|OFF|1|0`	Y
`*PSC?`	Y
`*RCL <reg_num>`	Y
`*RMC <label>`	N
`*RST`	Y
`*SAV <reg_num>`	Y
`*SRE <value>`	Y
`*SRE?`	Y
`*STB?`	Y
`*TRG`	Y
`*TST?`	Y
`*WAI`	Y
Abort Subsystem	
`:ABORt`	Y
Amplitude Modulation Subsystem	
`:AM:SOURce1 INTernal|EXTernal` `:AM:SOURce INTernal|EXTernal`	N Y
`:AM:SOURce1?` `:AM:SOURce?`	N Y
`:AM:STATe ON|OFF|1|0`	Y
`:AM:STATe?`	Y
Calibration Subsystem	

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG **N= Not supported by PSG**	**83751B & 83752B**
:CALibration:PEAKing[:EXECute]	N
:CALibration:PEAKing[:EXECute]? <dac_va>	N
:CALibration:PMETer:FLATness:INITiate? USER	N
:CALibration:PMETer:FLATness:NEXT? <value>[<lvlsuffix>]	N
:CALibration:SECurity:CODE <old> <new>	N
:CALibration:SECurity:PASSword <passwd>	N
:CALibration:TRACk	N
Correction Subsystem	
:CORRection:FLATness:AMPL <value>[DB],<value>[DB]...	N
:CORRection:FLATness:AMPL?	N
:CORRection:FLATness:FREQ <value>[<freqsuffix>],<value>[<freqsuffix>]...	N
:CORRection:FLATness:FREQ?	N
:CORRection:FLATness:POINts? MAXimum|MINimum	N
:CORRection:VOLTs:OFFSet	N
:CORRection:VOLTs:OFFSet?	N
:CORRection:VOLTs:SCALe	N
:CORRection:VOLTs:SCALe?	N
:CORRection[:STATe] ON|OFF|1|0	Y
:CORRection[:STATe]?	Y
Diagnostics Subsystem	
:DIAG:LRNS?	N
:DIAGnostic:TEST:FULLtest:REPort?	N

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG N= Not supported by PSG	83751B & 83752B
:DIAGnostic:TEST:FULLtest?	N

Display Subsystem	
:DISPlay[:STATe] ON|OFF|1|0	Y
:DISPlay[:STATe]?	Y
Frequency Modulation Subsystem	
:FM:COUPling AC|DC	Y
:FM:COUPling?	Y
:FM:SENSitivity <value><freqsuffix/V>	Y
:FM:SENSitivity?	Y
:FM:SOURce1 EXTernal :FM:SOURce EXTernal	N
:FM:SOURce1? :FM:SOURce?	N Y
:FM:STATe ON|OFF|1|0	Y
:FM:STATe?	Y
Frequency Subsystem	
:FREQuency:CENTer <value>[<freqsuffix>]|UP|DOWN	Y
:FREQuency:CENTer?	Y
:FREQuency:MANual <value><unit>|UP|DOWN	N
[:SOURce[1]]:FREQuency:MANual? [:SOURce]:FREQuency:MANual?	N Y
:FREQuency:MODE FIXed|CW|SWEep|SWCW	N
:FREQuency:MODE?	Y

Table 7-5 8375xB SCPI Commands (Continued)

<table>
<tr><th>Y= Supported by PSG
N= Not supported by PSG</th><th>83751B & 83752B</th></tr>
<tr><td>:FREQuency:MULTiplier <value></td><td>Y</td></tr>
<tr><td>:FREQuency:MULTiplier:STATe ON|OFF|1|0</td><td>N</td></tr>
<tr><td>:FREQuency:MULTiplier:STATe?</td><td>N</td></tr>
<tr><td>:FREQuency:MULTiplier?</td><td>Y</td></tr>
<tr><td>:FREQuency:OFFSet <value></td><td>Y</td></tr>
<tr><td>:FREQuency:OFFSet:STATe ON|OFF|1|0</td><td>Y</td></tr>
<tr><td>:FREQuency:OFFSet:STATe?</td><td>Y</td></tr>
<tr><td>:FREQuency:OFFSet?</td><td>Y</td></tr>
<tr><td>:FREQuency:SPAN <value>[<freqsuffix>]|UP|DOWN</td><td>Y</td></tr>
<tr><td>:FREQuency:SPAN?</td><td>Y</td></tr>
<tr><td>:FREQuency:STARt <value>[<freqsuffix>]|UP|DOWN</td><td>Y</td></tr>
<tr><td>:FREQuency:STARt?</td><td>Y</td></tr>
<tr><td>:FREQuency:STEP[:INCRement] <value>[<freqsuffix>]</td><td>Y</td></tr>
<tr><td>:FREQuency:STEP[:INCRement]?</td><td>Y</td></tr>
<tr><td>:FREQuency:STOP <value>[<freqsuffix>]|UP|DOWN</td><td>Y</td></tr>
<tr><td>:FREQuency:STOP?</td><td>Y</td></tr>
<tr><td>:FREQuency[:CW|:FIXed] <value>[<freqsuffix>]|UP|DOWN</td><td>Y</td></tr>
<tr><td>:FREQuency[:CW|:FIXed]:AUTO ON|OFF|1|0</td><td>N</td></tr>
<tr><td>:FREQuency[:CW|:FIXed]:AUTO?</td><td>N</td></tr>
<tr><td>:FREQuency[:CW|:FIXed]?</td><td>Y</td></tr>
<tr><td colspan="2">Initiate Subsystem</td></tr>
<tr><td>:INITiate:CONTinuous ON|OFF|1|0</td><td>Y</td></tr>
</table>

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG **N= Not supported by PSG**	**83751B & 83752B**
:INITiate:CONTinuous?	Y
:INITiate[:IMMediate]	Y
Marker Subsystem	
[:SOURce[1]]:MARKer[n]:AMPLitude[:STATe] ON|OFF|1|0 [:SOURce]:MARKer[n]:AMPLitude[:STATe] ON|OFF|1|0	N Y
[:SOURce[1]]:MARKer[n]:AMPLitude[:STATe]? [:SOURce]:MARKer[n]:AMPLitude[:STATe]?	N Y
:MARKer[n]:AOFF	Y
:MARKer[n]:FREQuency <value><unit>	Y
:MARKer[n]:FREQuency?	N
:MARKer[n]:MODE FREQuency|DELTa	Y
:MARKer[n]:MODE?	Y
:MARKer[n]:REFerence <n>	Y
:MARKer[n]:REFerence?	Y
:MARKer[n][:STATe] ON|OFF|1|0	N
:MARKer[n][:STATe]?	N
Memory Subsystem	
:MEMory:RAM:INITialize[:ALL]	N
Output Subsystem	
:OUTPut:IMPedance?	N
:OUTPut[:STATe] ON|OFF|1|0	Y
:OUTPut[:STATe]?	Y

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG N= Not supported by PSG	83751B & 83752B
Power Subsystem	
:POWer:ALC:CFACtor <value>[DB]|UP|DOWN	Y
:POWer:ALC:CFACtor?	Y
:POWer:ALC:SOURce1 INTernal|DIODe|PMETer|MMHead :POWer:ALC:SOURce INTernal|DIODe|PMETer|MMHead	N
:POWer:ALC:SOURce1? :POWer:ALC:SOURce?	N Y
:POWer:ALC[:STATe] ON|OFF|1|0	Y
:POWer:ALC[:STATe]?	Y
:POWer:ATTenuation <value>[DB]|UP|DOWN	Y
:POWer:ATTenuation:AUTO ON|OFF|1|0	Y
:POWer:ATTenuation:AUTO?	Y
:POWer:ATTenuation?	Y
:POWer:CENTer <value>[<lvlsuffix>]|UP|DOWN	Y
:POWer:CENTer?	Y
:POWer:MODE FIXed|SWEep	Y
:POWer:MODE?	Y
:POWer:OFFSet <value>[DB]|UP|DOWN	Y
:POWer:OFFSet:STATe ON|OFF|1|0	Y
:POWer:OFFSet:STATe?	Y
:POWer:OFFSet?	Y

Table 7-5 8375xB SCPI Commands (Continued)

<table>
<tr><th>Y= Supported by PSG
N= Not supported by PSG</th><th>83751B & 83752B</th></tr>
<tr><td>:POWer:SLOPe <value>[DB/freqsuffix]|UP|DOWN</td><td>N</td></tr>
<tr><td>:POWer:SLOPe:STATe ON|OFF|1|0</td><td>N</td></tr>
<tr><td>:POWer:SLOPe:STATe?</td><td>N</td></tr>
<tr><td>:POWer:SLOPe?</td><td>Y</td></tr>
<tr><td>:POWer:SPAN <value>[DB]|UP|DOWN</td><td>Y</td></tr>
<tr><td>:POWer:SPAN?</td><td>Y</td></tr>
<tr><td>:POWer:STARt <value>[<lvlsuffix>]|UP|DOWN</td><td>Y</td></tr>
<tr><td>:POWer:STARt?</td><td>Y</td></tr>
<tr><td>:POWer:STATe ON|OFF|1|0</td><td>Y</td></tr>
<tr><td>:POWer:STATe?</td><td>Y</td></tr>
<tr><td>:POWer:STEP[:INCRement] <value>[DB]</td><td>Y</td></tr>
<tr><td>:POWer:STEP[:INCRement]?</td><td>Y</td></tr>
<tr><td>:POWer:STOP <value>[<lvlsuffix>]|UP|DOWN</td><td>Y</td></tr>
<tr><td>:POWer:STOP?</td><td>Y</td></tr>
<tr><td>:POWer[:LEVel] <value>[<lvlsuffix>]|UP|DOWN</td><td>Y</td></tr>
<tr><td>:POWer[:LEVel]?</td><td>Y</td></tr>
<tr><td colspan="2">Pulse Modulation Subsystem</td></tr>
<tr><td>:PULM:SOURce1 INTernal|EXTernal|SCALar|SQ1K
:PULM:SOURce INTernal|EXTernal|SCALar|SQ1K</td><td>N</td></tr>
<tr><td>:PULM:SOURce1?
:PULM:SOURce?</td><td>N
Y</td></tr>
<tr><td>:PULM:STATe ON|OFF|1|0</td><td>Y</td></tr>
<tr><td>:PULM:STATe?</td><td>Y</td></tr>
</table>

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG N= Not supported by PSG	83751B & 83752B
Pulse Subsystem	
:PULSe:FREQuency <value>[<freqsuffix>]	Y
:PULSe:FREQuency?	Y
:PULSe:PERiod <value>[<timesuffix>]	Y
:PULSe:PERiod?	Y
:PULSe:WIDTh <value>[<timesuffix>]	Y
:PULSe:WIDTh?	Y
Reference Oscillator Subsystem	
:ROSCillator:SOURce1 INTernal|EXTernal|NONE :ROSCillator:SOURce INTernal|EXTernal|NONE	N Y
:ROSCillator:SOURce1:AUTO ON|OFF|1|0 :ROSCillator:SOURce:AUTO ON|OFF|1|0	N Y
:ROSCillator:SOURce1:AUTO? :ROSCillator:SOURce:AUTO?	N Y
:ROSCillator:SOURce1? :ROSCillator:SOURce?	N Y
Status Subsystem	
:STATus:OPERation:CONDition?	Y
:STATus:OPERation:ENABle <value>	Y
:STATus:OPERation:ENABle?	Y
:STATus:OPERation:NTRansition <value>	Y
:STATus:OPERation:NTRansition?	Y

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG **N= Not supported by PSG**	**83751B & 83752B**
:STATus:OPERation:PTRansition <value>	Y
:STATus:OPERation:PTRansition?	Y
:STATus:OPERation[:EVENt]?	Y
:STATus:PRESet	Y
:STATus:QUEStionable:CONDition?	Y
:STATus:QUEStionable:ENABle <value>	Y
:STATus:QUEStionable:ENABle?	Y
:STATus:QUEStionable:NTRansition <value>	Y
:STATus:QUEStionable:NTRansition?	Y
:STATus:QUEStionable:PTRansition <value>	Y
:STATus:QUEStionable:PTRansition?	Y
:STATus:QUEStionable[:EVENt]?	Y
Sweep Subsystem	
:SWEep:CONTrol:TYPE MASTer|SLAVe	Y
:SWEep:CONTrol:TYPE?	Y
:SWEep:DWELl <value>[<timesuffix>]	Y
:SWEep:DWELl:AUTO ON|OFF|1|0	N
:SWEep:DWELl:AUTO?	N
:SWEep:DWELl?	Y
:SWEep:GENeration ANALog|STEPped	Y
:SWEep:GENeration?	Y
:SWEep:MANual:POINt <value>	Y

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG N= Not supported by PSG	83751B & 83752B
:SWEep:MANual:POINt?	Y
:SWEep:MANual[:RELative] <value>	N
:SWEep:MANual[:RELative]?	N
:SWEep:MARKer:STATe ON|OFF|1|0	N
:SWEep:MARKer:STATe?	N
:SWEep:MARKer:XFER	N
:SWEep:MODE AUTO|MANual	Y
:SWEep:MODE?	Y
:SWEep:POINts <value>	Y
:SWEep:POINts?	Y
:SWEep:POWer:STEP <value>[<lvlsuffix>]|UP|DOWN	N
:SWEep:POWer:STEP?	N
:SWEep:TIME <value>[<timesuffix>]	N
:SWEep:TIME:AUTO ON|OFF|1|0	N
:SWEep:TIME:AUTO?	Y
:SWEep:TIME:LLIMit <value>[<timesuffix>]	Y
:SWEep:TIME:LLIMit?	Y
:SWEep:TIME?	Y
:SWEep[:FREQuency]:STEP <value>[<freqsuffix>]|UP|DOWN	N
:SWEep[:FREQuency]:STEP?	N
:SWEep[:POINts]:TRIGger:SOURce IMMediate|BUS|EXTernal :SWEep[:POINts]:TRIGger:SOURce IMMediate|BUS|EXTernal	N

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG **N= Not supported by PSG**	**83751B & 83752B**
:SWEep[:POINts]:TRIGger:SOURce? :SWEep[:POINts]:TRIGger:SOURce?	N
:SWEep[:POINts]:TRIGger[:IMMediate]	N
System Subsystem	
:SYSTem:ALTernate <reg num>	Y
:SYSTem:ALTernate:STATe ON|OFF|1|0	Y
:SYSTem:ALTernate:STATe?	Y
:SYSTem:ALTernate?	Y
:SYSTem:COMMunicate:GPIB:ADDRess <value>	Y
:SYSTem:COMMunicate:PMETer:ADDRess <value>	Y
:SYSTem:COMMunicate:PMETer:ADDRess?	Y
:SYSTem:COMMunicate:PMETer:TYPE SCPI|70100A|437B|438A	N
:SYSTem:COMMunicate:PMETer:TYPE?	N
:SYSTem:ERRor?	Y
:SYSTem:KEY:DISable SAVE	N
:SYSTem:KEY:DISable? SAVE	N
:SYSTem:KEY:ENABle SAVE	N
:SYSTem:KEY:ENABle? SAVE	N
:SYSTem:KEY[:CODE] <value>	N
:SYSTem:KEY[:CODE]?	N
:SYSTem:LANGuage "SCPI"|"TMSL"|"COMP"	N
:SYSTem:LANGuage?	Y
:SYSTem:PRESet:TYPE FACTory|USER	Y

Table 7-5 8375xB SCPI Commands (Continued)

Y= Supported by PSG **N= Not supported by PSG**	**83751B & 83752B**
:SYSTem:PRESet:TYPE?	Y
:SYSTem:PRESet[:EXECute]	Y
:SYSTem:PRESet[:USER]:SAVE	Y
:SYSTem:SECurity:CLEar	N
:SYSTem:SECurity:COUNt <value>	Y
:SYSTem:SECurity:KLOCk ON|OFF|0|1	N
:SYSTem:SECurity:ZERO ON|OFF|0|1	N
:SYSTem:VERSion?	Y
Trigger Subsystem	
:TRIGger:SOURce1 IMMediate|BUS|EXTernal|HOLD :TRIGger:SOURce IMMediate|BUS|EXTernal|HOLD	N
:TRIGger:SOURce1? :TRIGger:SOURce?	N Y
:TRIGger[:IMMediate]	Y
Tsweep Subsystem	
:TSWeep	Y

8662A / 63A Compatible Commands

The tables in this section provide the following:

Table 7-6 on page 438: a comprehensive list of 8662A/63A programming commands, listed in alphabetical order. The equivalent SCPI command sequence for each supported code is provided. Codes that have no equivalent SCPI command sequence are indicated in the command column, as are codes that are *not* supported by the PSG family.

Table 7-7 on page 447: a list of the implemented 8662A/63A programming commands that set the active function. This table also indicates which codes are compatible with the increment (up), and the decrement (down) SCPI commands.

NOTE Compatibility is provided for GPIB only; RS-232 and LAN are *not* supported.

Device Clear does not preset the instrument.

To reproduce the sweep functionality, use the PSG List Sweep features.

Table 7-6 8662A / 63A Commands & Equivalent SCPI Sequences

Command	Description	8662	8663	Equivalent SCPI Command Sequence
@1	Write require service mask	N	N	*not supported*
@2	Deferred execution mode	N	N	*not supported*
@3	Immediate execution mode	N	N	*not supported*
+D	+dBm	Y	Y	`DBM`

AM	AM modulation
	See also: Table 7-7 on page 447
		Y		`AM:DEPTh <val> <units>`
				`AM:TRAC ON`
				`FM:STAT OFF`
				`AM:STAT ON`

			Y	`AM:DEPTh <val> <units>`
				`AM:TRAC ON`
				`AM:STAT ON`

AO	Amplitude off	Y	Y	`OUTPut:STATe OFF`

AP	Amplitude	Y	Y	`POW:REF:STATe OFF`
				`POWer:AMPL <val> <units>`
				`OUTPut:STATe ON`
				See also: Table 7-7 on page 447

Table 7-6 8662A/63A Commands & Equivalent SCPI Sequences (Continued)

Command	Description	8662	8663	Equivalent SCPI Command Sequence
AS|BLSQ	Auto sequence	N	N	*not supported*
BP	BPSK modulation		N	*not supported*
CT	Configure trigger	Y	Y	*no equivalent SCPI command sequence*
–D	–dBm Negates the power value.	Y	Y	DBM
DB	dB	Y	Y	DB
DG	Degree	Y		DEG
DM	dBm	Y	Y	DBM
DN	Decrement Passes DOWN as parameter of active function command.	Y	Y	*See* Table 7-7 on page 447
FA	Start frequency	Y	Y	*See W2, W3, W4,* *and* Table 7-7 on page 447
FB	Stop frequency	Y	Y	*See W2, W3, W4,* *and* Table 7-7 on page 447
FM	FM modulation *See also*: Table 7-7 on page 447	Y		FM:DEV <val> <units> AM:STAT OFF FM:STAT ON
			Y	FM:DEV <val> <units> FM:STAT ON
FR	Center frequency	Y	Y	FREQuency:CW <val> <units> *See also*: *W2, W3, and W4,* *and* Table 7-7 on page 447
FS	Span frequency	Y	Y	*See W2, W3, W4,* *and* Table 7-7 on page 447
GZ	GHz	Y	Y	GHZ
HZ	Hz	Y	Y	HZ
IS	Set increment Adds STEP:INCR to active function command.	Y	Y	*no equivalent SCPI command sequence*
KZ	kHz	Y	Y	KHZ
L1	Learn front panel	N	N	*not supported*
L2	Fast learn	N	N	*not supported*

Table 7-6 8662A / 63A Commands & Equivalent SCPI Sequences (Continued)

<table>
<tr><th>Command</th><th>Description</th><th>8662</th><th>8663</th><th>Equivalent SCPI Command Sequence</th></tr>
<tr><td>MO|M0</td><td>Modulation off</td><td>Y</td><td>Y</td><td>AM:STATe OFF
FM:STATe OFF
PULM:STATe OFF
PM:STATe OFF</td></tr>
<tr><td rowspan="2">M1</td><td rowspan="2">For 8662A: <mod> = FM or AM, depending on which is on.

Modulation source internal 400 Hz

For 8663A: Executes MF with <freq> = 400 Hz</td><td>Y</td><td></td><td><mod>:SOURce INT1
<mod>:INT1:FREQ 400Hz</td></tr>
<tr><td></td><td>Y</td><td>AM:INT1:FREQ 400 MHz
FM:INT2:FREQ 400 MHz
PM:INT2:FREQ 400 MHz
PULM:INT:FREQ 400 MHz</td></tr>
<tr><td rowspan="2">M2</td><td rowspan="2">For 8662A: <mod> = FM or AM, depending on which is on.

Modulation source internal 1 kHz

For 8663A: Executes MF with <freq> = 1 kHz</td><td>Y</td><td></td><td><mod>:SOURce INT1
<mod>:INT1:FREQ 1kHz</td></tr>
<tr><td></td><td>Y</td><td>AM:INT1:FREQ 1
kHzFM:INT2:FREQ 1
kHzPM:INT2:FREQ 1
kHzPULM:INT:FREQ 1 kHz</td></tr>
<tr><td rowspan="2">M3</td><td rowspan="2">For 8662A: <mod> = FM or AM, depending on which is on.

Modulation source external AC

For 8663A:
<mod> = AM, FM, or PM, depending on which is on.
<n> = 1 for AM, 2 for FM or PM
NOTE: For PM, the impedance value is set using the SP71/SP70 commands</td><td>Y</td><td></td><td><mod>:SOURce EXT
<mod>:EXT:COUPling AC
<mod>:EXT:IMP 600</td></tr>
<tr><td></td><td>Y</td><td><mod>:SOURce EXT<n>
<mod>:EXT<n>:COUPling AC
<mod>:EXT<n>:IMP 600</td></tr>
<tr><td rowspan="2">M4</td><td rowspan="2">For 8662A: <mod> = FM or AM, depending on which is on.

Modulation source external DC

For 8663A:
<mod> = AM, FM, or PM, depending on which is on.
<n> = 1 for AM, 2 for FM or PM
NOTE: For PM, the impedance value is set using the SP71/SP70 commands</td><td>Y</td><td></td><td><mod>:SOURce EXT
<mod>:EXT:COUPling DC
<mod>:EXT:IMP 600</td></tr>
<tr><td></td><td>Y</td><td><mod>:SOURce EXT<n>
<mod>:EXT<n>:COUPling DC
<mod>:EXT<n>:IMP 600</td></tr>
</table>

Table 7-6 8662A / 63A Commands & Equivalent SCPI Sequences (Continued)

Command	Description	8662	8663	Equivalent SCPI Command Sequence

MF	Modulation frequency
	<mod> = FM, or PM, depending on which is on.
	Also see: *M1, M2, and* Table 7-7 on page 447
			Y	**AM:**
				`AM:SOUR: INT1`
				`AM:SOUR:INT1:FREQ <freq>`
				FM or PM:
				`<mod>:SOUR: INT2`
				`<mod>:SOUR:INT2:FREQ <freq>`
				Pulse:
				`PULM:SOUR: INT`
				`PULM:INT:FREQ <freq>`
				`PULM:SOUR:INT SQUARE`

MS	Read status key message Returns status string.	Y	Y	*no equivalent SCPI command sequence*
MV	mV	Y	Y	`MV`
MZ	MHz	Y	Y	`MHZ`
N1	Linear 100 steps	Y	Y	*See W2, W3, and W4*
N2	Linear 1000 steps	Y	Y	*See W2, W3, and W4*
N3	Step size	Y	Y	*See W2, W3,W4,* *and* Table 7-7 on page 447
N4	Log 10% steps	Y	Y	*See W2, W3, and W4*
N5	Log 1% steps	Y	Y	*See W2, W3, and W4*
PC	%	Y	Y	`PCT`
PL	Pulse modulation Must have an instrument with pulse capability.		Y	`PULM:STAT ON`
PM	Phase modulation Not compatible with any FM modulation.		Y	`PM:STAT ON` *See also*: Table 7-7 on page 447
R1	Knob resolution x10	N	N	*not supported*
R2	Knob resolution /10	N	N	*not supported*
R3	Knob off	N	N	*not supported*
R4|BLR1	Knob hold	N	N	*not supported*
R5|BLR2	Knob increment	N	N	*not supported*
RC	Recall	Y	Y	`*RCL`
RD	Knob down Only for manual sweep	Y	Y	`LIST:MANual DOWN`

Table 7-6 8662A / 63A Commands & Equivalent SCPI Sequences (Continued)

Command	Description	8662	8663	Equivalent SCPI Command Sequence
RM	Read require service mask	N	N	*not supported*
RU	Knob up Only for manual sweep	Y	Y	LIST:MANual UP
SP00	System preset Presets the instrument, including the compatibility language.	Y	Y	SYSTem:PRESet
SP10	Frequency offset off	Y	Y	FREQ:OFFS:STAT OFF
SP11	Positive frequency offset The 8662 modifies the output, but does not change the displayed frequency; the PSG modifies the displayed frequency, but does *not* change the output. Because of this, you must first set the offset, then reapply the frequency to change the output.	Y	Y	FREQ:OFFS -<value> FREQ:OFFS:STAT ON FREQ:CW <displayed value>
SP12	Negative frequency offset The 8662 modifies the output, but does not change the displayed frequency; the PSG modifies the displayed frequency, but does *not* change the output. Because of this, you must first set the offset, then reapply the frequency to change the output.	Y	Y	FREQ:OFFS <value> FREQ:OFFS:STAT ON FREQ:CW <displayed value>
SP20	ALC bandwidth normal		Y	POWer:ALC:BANDwidth:AUTO ON
SP21	ALC bandwidth < 1 kHz		Y	POWer:ALC:BANDwidth:AUTO OFFPOWer:ALC:BANDwidth 1KHZ
SP30	Amplitude reference off	Y	Y	POW:REF:STATe OFF
SP31	Amplitude reference	Y	Y	POW:REF <val> <val> = current amplitude setting POW:REF:STATe ON
SP32	Amplitude reference relative to 1 µV		Y	POW:REF 106.99DBM POW:REF:STATe ON POW 1UV
SP40	External AM off	Y		AM:STAT OFF
	Modulation frequency sweep mode off		N	*not supported*

Table 7-6 8662A / 63A Commands & Equivalent SCPI Sequences (Continued)

Command	Description	8662	8663	Equivalent SCPI Command Sequence

SP41	Internal FM + external AM (AC)	Y		`FM:SOUR INT1`
				`FM:INT1:FREQ 400 HZ`
				`FM:STAT ON`
				`AM:SOUR EXT1`
				`AM:EXT1:IMP 600`
				`AM:DEPTH 95 PCT`
				`AM:EXT1:COUP AC`
				`AM:STAT ON`

	Modulation frequency sweep mode on		N	*not supported*

SP42	Internal FM + external AM (DC)	Y		`FM:SOUR INT1`
				`FM:INT1:FREQ 400 HZ`
				`FM:STAT ON`
				`AM:SOUR EXT1`
				`AM:EXT1:IMP 600`
				`AM:DEPTH 95 PCT`
				`AM:EXT1:COUP DC`
				`AM:STAT ON`

SP50	AUX FM off	Y	Y	`FM2:STAT OFF`

SP51	AUX FM on
	
	RF (MHz) FM Deviation (kHz)
	0.01–120 25
	120–160 6.25
	160–320 12.5
	320–640 25
	640–1280 50
	1280–2560 100
	
	<dev> is dependant on output frequency, and mimics the 8662 hardware settings.
	
	NOTE: The deviation for this command cannot be greater than the deviation of the `FM1` path.
		Y	Y	`FM2:SOUR EXT2`
				`FM2:EXT2:COUP DC`
				`FM2:EXT2:IMP 600`
				`FM2:DEV <dev> kHz`
				`FM2:STAT ON`

SP60	Parameter shift keying off	N	N	*not supported*
SP61	Parameter shift keying up/down (two-key)	N	N	*not supported*
SP62	Parameter shift keying up/down (one-key)	N	N	*not supported*
SP70	External PM input impedance 50Ω Effects the behavior of `M3` and `M4`.		Y	*no equivalent SCPI command sequence*
SP71	External PM input impedance 600Ω Effects the behavior of `M3` and `M4`.		Y	*no equivalent SCPI command sequence*

SP80	Special functions 10-62 off	Y	Y	`FM2:STAT OFF`
				`AM:STAT OFF`
				`FREQ:OFFS:STAT OFF`

Table 7-6 8662A / 63A Commands & Equivalent SCPI Sequences (Continued)

Command	Description	8662	8663	Equivalent SCPI Command Sequence
SP81	Amplitude conversion (V-dBm)	N	N	*not supported*
SP82	Display GPIB address	N	N	*not supported*
SP83	ROM test	N	N	*not supported*
SP84	RAM test	N	N	*not supported*
SP85	Amplitude correction off	Y	Y	`POWer:ALC:STATe OFF`
SP86	Amplitude correction on PSG ALC ON always works with sweep.	Y	Y	`POWer:ALC:STATe ON`
SP87	Amplitude correction on (includes Sweep)		Y	`POWer:ALC:STATe ON`
SP87	GPIB operator request response	N		*not supported*
SP88	Auto sequence	N	N	*not supported*
SP89	GPIB operator request response		N	*not supported*
SP90	Set auto sequence step delay		N	*not supported*
SP91	Enable frequency hopping mode		N	*not supported*
SP92	Knob (restore normal operation)		N	*not supported*
SP93	Manual amplitude level control		N	*not supported*
SP94	Knob, 120 increments per revolution		N	*not supported*
SP95	Knob, 120 increments per revolution, reconfigure AUX con.		N	*not supported*
SP96	Modulation oscillator off when modulation is off		N	*not supported*
SP97	Modulation oscillator on		N	*not supported*
SP98	Turn display on		Y	`DISP ON`
SP99	Turn display off		Y	`DISP OFF`
SP2.0	Power up preset off		N	*not supported*
SP2.1	Power up preset on		N	*not supported*
SQ	Sequence	N	N	*not supported*
SS|BLST	Set sequence	N	N	*not supported*
ST	Store Saves/recalls register to sequence 0.	Y	Y	`*SAV`
T1	0.5 ms per step	Y	Y	`SWEEP:DWELL 0.5ms` *Beyond PSG range limit; is set to 1ms.*

Table 7-6 8662A / 63A Commands & Equivalent SCPI Sequences (Continued)

Command	Description	8662	8663	Equivalent SCPI Command Sequence
T2	1 ms per step	Y	Y	`SWEEP:DWELL 1ms`
T3	2 ms per step	Y	Y	`SWEEP:DWELL 2ms`
T4	10 ms per step	Y	Y	`SWEEP:DWELL 10ms`
T5	100 ms per step	Y	Y	`SWEEP:DWELL 100ms`
TR	Trigger Performs command code setup with `CT` command.	Y	Y	*no equivalent SCPI command sequence*
UP	Increment Passes `UP` as a parameter of the active function command.	Y	Y	*See Table 7-7 on page 447*
UV	µV	Y	Y	`UV`
W1	Sweep off	Y	Y	`FREQ:MODE CW` `LIST:TRIG:SOUR IMM`

W2	Auto sweep mode on
	
	Generates a sweep list based on stored parameters from FA, FB, FR, FS, N1, N2, N3, N4, and N5
	Default values:
	FR = 100 MHz,
	FS = 10 MHz,
	N1, T2 FA = 1 MHz,
	FB = 1279 MHz
		Y	Y	`INIT:CONT ON`
				`SWEEP:MODE AUTO`
				`LIST:TRIG:SOUR IMM`
				`LIST:DWELl:TYPE STEP`
				`LIST:TYPE LIST`
				`FREQ:MODE LIST`

W3	Manual sweep mode on
	
	Generates a sweep list based on stored parameters from FA, FB, FR, FS, N1, N2, N3, N4, and N5
	Default values:
	FR = 100 MHz,
	FS = 10 MHz,
	N1, T2 FA = 1 MHz,
	FB = 1279 MHz
		Y	Y	`INIT:CONT ON`
				`SWEEP:MODE MANual`
				`LIST:TRIG:SOUR IMM`
				`LIST:DWELl:TYPE STEP`
				`LIST:TYPE LIST`
				`FREQ:MODE LIST`

W4	Single sweep mode on
	
	Generates a sweep list based on stored parameters from FA, FB, FR, FS, N1, N2, N3, N4, and N5
	Default values:
	FR = 100 MHz,
	FS = 10 MHz,
	N1, T2 FA = 1 MHz,
	FB = 1279 MHz
		Y	Y	`INIT:CONT OFF`
				`SWEEP:MODE AUTO`
				`LIST:TRIG:SOUR IMM`
				`LIST:DWELl:TYPE STEP`
				`LIST:TYPE LIST`
				`FREQ:MODE LIST`
				`INIT`

X1	Marker 1	N	N	*not supported*

Table 7-6 8662A / 63A Commands & Equivalent SCPI Sequences (Continued)

Command	Description	8662	8663	Equivalent SCPI Command Sequence
X2	Marker 2	N	N	*not supported*
X3	Marker 3	N	N	*not supported*
X4	Marker 4	N	N	*not supported*
X5	Marker 5	N	N	*not supported*
X6	Marker off	N	N	*not supported*
X7|BLX6	All markers off	N	N	*not supported*
Y0	Remote stepped sweep off	Y	Y	FREQ:MODE CW LIST:TRIG:SOUR IMM
Y1|Y2	Remote stepped sweep on	Y	Y	INIT:CONT ON SWEEP:MODE AUTO LIST:DWELl:TYPE STEP LIST:TYPE LIST FREQ:MODE LIST LIST:TRIG:SOUR BUS
Y3	Execute remote stepped sweep	Y	Y	*TRG

Table 7-7 8662 / 63B Command Compatibility

Command	Description	Sets Active Function	Compatible with UP / DN	8662	8663	Equivalent SCPI Commands for UP / DN and Increment

AM	AM modulation	Y	Y	Y	Y	`AM:DEPTH UP`
						`AM:DEPTH DOWN`
						`AM:DEPTH:STEP:INCR`

AP	Amplitude	Y	Y	Y	Y	`POW:AMPL UP`
						`POW:AMPL DOWN`
						`POW:AMPL:STEP:INCR`

FA	Start frequency	Y	Y	Y	Y	`FREQ:CW:STEP:INCR`
FB	Stop frequency	Y	Y	Y	Y	`FREQ:CW:STEP:INCR`

FM	FM modulation	Y	Y	Y	Y	`FM:DEV UP`
						`FM:DEV DOWN`
						`FM:DEV:STEP:INCR`

FR	Center frequency	Y	Y	Y	Y	`FREQ:CW UP`
						`FREQ:CW DOWN`
						`FREQ:CW:STEP:INCR`

FS	Span frequency	Y	Y	Y	Y	`FREQ:CW:STEP:INCR`
MF	Modulation frequency	Y	Y		Y	<mod>:INT:FREQ UP <mod>:INT:FREQ DOWN <mod>:INT:FREQ:STEP:INCR <mod> = AM|FM|PM|PULM
N3	Step size	Y	Y	Y	Y	*no equivalent SCPI commands*

PM	Phase modulation
	Not compatible with any FM modulation.
		Y	Y		Y	`PM:DEV UP`
						`PM:DEV DOWN`
						`PM:DEV:STEP:INCR`

Symbols

Numerics

A

Index

G

T