Programmer's Guide

Agilent Technologies 8712ET/ES and 8714ET/ES RF Network Analyzers

Agilent Technologies

Part No. 08714-90015
Printed in USA
Print Date: June 2000
Supersedes: October 1999

Notice

The information contained in this document is subject to change without notice.

Agilent Technologies makes no warranty of any kind with regard to this material, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose.
Agilent Technologies shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Key Conventions

This manual uses the following conventions:

(FRONT PANEL KEY): This represents a key physically located on the analyzer (a "hardkey").

Softkey : This indicates a "softkey"-- a key whose label is determined by the instrument's firmware, and is displayed on the right side of the instrument's screen next to the eight unlabeled keys.

Firmware Revision

This manual documents analyzers with firmware revisions E.06.00 and above.

GPIB Programming

This document is an introduction to programming your analyzer over the general purpose interface bus (GPIB). Its purpose is to provide concise information about the operation of the instrument under GPIB control. It provides some background information on the GPIB and some short programming examples to demonstrate the remote operation of the analyzer.

Example programs can be run on the analyzer's internal controller or on an external controller. These programs can be found in the following three locations:

- *Example Programs Disk* (included with the analyzer)— DOS Format : part number 08714-10003.

  A LIF version of the *Example Programs Disk* is available, but is not shipped with your analyzer:

  ExamplePrograms Disk – LIF Format part number 08714-10004.

  Contact the nearest Agilent Technologies sales office for ordering information. A list of Agilent Technologies sales and service offices can be found in the "Specifications" chapter of the *User's Guide*.
- *Example Programs Guide* (included with the analyzer): part number 08714-90016. (This document may not include all of the example programs found on the disk or on the Web site.)
- Web site **http://www.agilent.com**. Use the search function to find Web pages related to 8712 example programs.

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

- Information on making measurements with the analyzer is available in the analyzer's *User's Guide.*
- Information on HP Instrument BASIC is available in the *HP Instrument BASIC User's Handbook.*

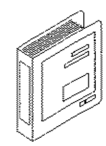
- Information on HP BASIC programming is available in the manual set for the BASIC revision being used. For example: *BASIC 7.0 Programming Techniques* and *BASIC 7.0 Language Reference*.
- Example programs are described in *Example Programs Guide*.
- Information on using the GPIB is available in the *Tutorial Description of the Hewlett-Packard Interface Bus* (HP literature no. 5021-1927).
- Information on using the analyzer to make automated measurements is available in *Automated Measurements User's Guide Supplement*.
- Information on using the analyzer with a Local Area Network (LAN) is available in *The LAN Interface User's Guide*.

Contact the nearest Agilent Technologies sales office for ordering information. A list of sales and service offices can be found in the "Specifications" chapter of the *User's Guide*.

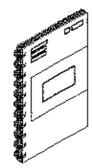
Agilent Technologies 8712ET/ES and 8714ET/ES Network Analyzer Documentation Map

The **CDROM** provides the contents of all of the documents listed below.

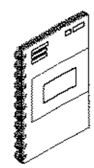
The **User's Guide** shows how to make measurements, explains commonly-used features, and tells you how to get the most performance from the analyzer.



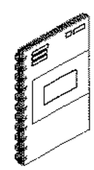
The **LAN Interface User's Guide Supplement** shows how to use a local area network (LAN) for programming and remote operation of the analyzer.

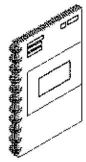


The **Automating Measurements User's Guide Supplement** provides information on how to configure and control test systems for automation of test processes.

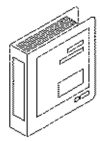


The **Programmer's Guide** provides programming information including GPIB and SCPI command references, as well as short programming examples.

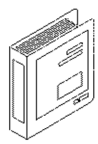




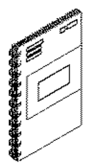
The **Example Programs Guide** provides a tutorial introduction using BASIC programming examples to demonstrate the remote operation of the analyzer.



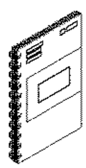
The **Service Guide** provides the information needed to adjust, troubleshoot, repair, and verify analyzer conformance to published specifications.



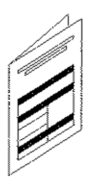
The **HP Instrument BASIC User's Handbook** describes programming and interfacing techniques using HP Instrument BASIC, and includes a language reference.



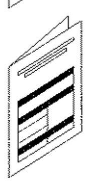
The **HP Instrument BASIC User's Handbook Supplement** shows how to use HP Instrument BASIC to program the analyzer.



The **Option 100 Fault Location and Structural Return Loss Measurements User's Guide Supplement** provides theory and measurement examples for making fault location and SRL measurements. (Shipped only with Option 100 analyzers.)



The **CATV Quick Start Guide** provides abbreviated instructions for testing the quality of coaxial cables. (Shipped only with Option 100 analyzers.)



The **Cellular Antenna Quick Start Guide** provides abbreviated instructions for verifying the performance of cellular antenna systems. (Shipped only with Option 100 analyzers.)

1 Introduction to GPIB Programming

Introduction to GPIB Programming

GPIB—the general purpopse interface bus—is a high-performance bus that connects individual instruments and computers together to make integrated test systems. The bus and its associated interface operations are defined by the IEEE 488.1 standard. The IEEE 488.2 standard defines the interface capabilities of instruments and controllers in a measurement system, including some frequently used commands.

GPIB cables provide the physical link between devices on the bus. There are eight data lines on each cable that are used to send data from one device to another. Devices that send data over these lines are called **Talkers**. **Listeners** are devices that receive data over the same lines. There are also five control lines on each cable that are used to manage traffic on the data lines and to control other interface operations. **Controllers** are devices that use these control lines to specify the talker and listener in a data exchange. When a GPIB system contains more that one device with controller capabilities, only one of the devices is allowed to control data exchanges at any given time. The device currently controlling data exchanges is called the **Active Controller**. Also, only one of the controller-capable devices can be designated as the **System Controller**, the one device that can take control of the bus even if it is not the active controller. The network analyzer can act as a talker, listener, active controller or system controller at different times.

GPIB addresses provide a way to identify devices on the bus. Each device on the bus must have a unique address. The active controller uses GPIB addresses to specify which device talks and which device listens during a data exchange. Device addresses are set on each device using either a front-panel key sequence or a rear-panel switch.

To set the GPIB address on the analyzer, use the softkeys located in the (SYSTEM OPTIONS) **GPIB** menu. The factory default address for the analyzer is 16.

NOTE

Throughout this manual, the following conventions are used:

- **Square brackets** ([ ]) are used to enclose a keyword that is optional or implied when programming the command; that is, the instrument will process the command to have the same effect whether the option node is omitted or not.
- **Parameter types** (< >) are distinguished by enclosing the type name in angle brackets.
- **A vertical bar** (|) can be read as "or" and is used to separate alternative parameter options.
- A (HARDKEY) is a labeled button on the instrument front panel.
- A **Softkey** is one of the eight unlabeled buttons along the right side of the instrument display. The function of each **Softkey** is indicated next to the **Softkey** on the instrument display.

Bus Structure

Data Bus

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

Handshake Lines

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

Control Lines

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

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

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

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

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

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

Sending Commands

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

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

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

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

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

This sends the command `CALCULATE:MARKER:MAXIMUM` to the GPIB device at address 716. If the device is an analyzer, the command instructs the analyzer to set a marker to the maximum point on the data trace.

GPIB Requirements

Number of Interconnected Devices:
15 maximum

Interconnection Path/Maximum Cable Length:
20 meters maximum or 2 meters per device, whichever is less.

Message Transfer Scheme:
Byte serial/bit parallel asynchronous data transfer using a 3-line handshake system.

Data Rate:
Maximum of 1 megabyte per second over limited distances with tri-state drivers. The actual data rate is the transfer rate of the slowest device involved.

Address Capability:
Primary addresses: 31 talk, 31 listen. A maximum of 1 talker and 14 listeners at one time.

Multiple Controller Capability:
In systems with more than one controller (like the analyzer system), only one can be active at a time. The active controller can pass control to another controller, but only the system controller can assume unconditional control. Only one system controller is allowed. The system controller is hard-wired to assume bus control after a power failure.

Interface Capabilities

The analyzer has the following interface capabilities, defined by the IEEE 488.1 standard:

Table 1-1 Analyzer Interface Capabilities (IEEE 488.1)

SH1	full Source handshake capability
AH1	full Acceptor handshake capability
T6	basic Talker, Serial Poll, no Talk Only, unaddress if MLA
TE0	no Extended Talker capability
L4	basic Listener, no Listen Only, unaddress if MTA
LE0	no Extended Listener capability
SR1	full Service Request capability
RL1	full Remote/Local capability
DC1	full Device Clear capability
C1	System Controller capability
C2	send IFC and take charge Controller capability
C3	send REN Controller capability
C4[1]	respond to SRQ
C8[1]	send IFC, receive control, pass control, pass control to self
C12[2]	send IF messages, receive control, pass control
E2	tri-state drivers
DT1	full device trigger capability
PP0	no parallel poll capability

1. only when an HP Instrument BASIC program is running
2. only when an HP Instrument BASIC program is not running

Programming Fundamentals

This section includes specific information for programming your network analyzer. It includes how the analyzer interacts with a controller, how data is transferred between the analyzer and a controller, and how to use the analyzer's status register structure to generate service requests.

Controller Capabilities

The analyzer can be configured as a GPIB system controller or as a talker/listener on the bus. To configure the analyzer, select either the **System Controller** or the **Talker Listener** softkey in the (SYSTEM OPTIONS) **GPIB** menu.

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

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

Response to Bus Management Commands

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

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

Device Clear (DCL)

When the analyzer receives this command, it does the following:

- clears its input and output queues
- resets its command parser (so it is ready to receive a new program message)
- cancels any pending `*OPC` command or query

The command does not affect the following:

- front panel operation
- any analyzer operations in progress (other than those already mentioned)
- any instrument settings or registers (although clearing the output queue may indirectly affect the status byte's Message Available (MAV) bit)

Go To Local (GTL)

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

Interface Clear (IFC)

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

Local Lockout (LLO)

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

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

Parallel Poll

The analyzer ignores all of the following parallel poll commands:

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

Remote Enable (REN)

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

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

Selected Device Clear (SDC)

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

When the analyzer receives this command it does the following:

- clears its input and output queues
- resets its command parser (so it is ready to receive a new program message)
- cancels any pending `*OPC` command or query

The command does not affect the following:

- front-panel operation
- any analyzer operations in progress (other than those already mentioned)
- any analyzer settings or registers (although clearing the output queue may indirectly affect the status byte's MAV bit) passed

Serial Poll

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

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

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

Take Control Talker (TCT)

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

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

Message Exchange

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

Response messages are used to return data from the analyzer. The syntax for both kinds of messages is discussed in Chapter 9, "Introduction to SCPI."

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

- The analyzer only talks after it receives a terminated query (see "Query Response Generation" on page 1-16).
- Once it receives a terminated query, the analyzer expects to talk before it is told to do something else.

GPIB Queues

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

- an input queue
- an error queue
- an output queue

Input Queue

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

- device commands and queries
- the GPIB END message (EOI asserted while the last data byte is on the bus)

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

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

Error Queue

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

The error queue is cleared when the following actions occur:

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

Output Queue

The output queue temporarily stores a single response message until it is read by a controller. It is cleared when the following actions occur:

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

Command Parser

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

One of the parser's most important functions is to determine the position of a program message in the analyzer's command tree (described in Chapter 9). When the command parser is reset, the next command it receives is expected to arise from the base of the analyzer's command tree.

The parser is reset when the following actions occur:

- the analyzer is turned on
- The Device Clear (DCL) or Selected Device Clear (SDC) command is received.
- a colon immediately follows a semicolon in a program message. (For more information see "Sending Multiple Commands" on page 9-7.)
- A program message terminator is received. A program message terminator can be an ASCII carriage return ($^{C}{}_{R}$) or newline character or the GPIB END message (EOI set true).

Query Response Generation

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

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

2 Synchronizing the Analyzer and a Controller

Synchronizing the Analyzer and a Controller

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

Device commands can be divided into two broad classes:

- Sequential commands
- Overlapped commands

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

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

Overlapped Commands

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

The analyzer has the following overlapped commands:

```
ABORt
CALibration:SELF: ALL
CALibration:SELF: <ON|OFF|ONCE>
CALibration:SELF:METHod:<ONEPort|TWOPort>
CALibration:ZERO:AUTO
CONFigure[1|2]
DIAGnostic:CCONstants:LOAD
DIAGnostic:CCONstants:STORe:DISK
DIAGnostic:CCONstants:STORe:EEPRom
DIAGnostic:DITHer
DIAGnostic:SPUR:AVOid
HCOPy[:IMMediate]
INITiate[1|2]:CONTinuous
INITiate[1|2][:IMMediate]
MMEMory:LOAD:STATe
OUTPut[:STATe]
POWer[1|2]:MODE
PROGram[:SELected]:EXECute
ROUTe[1|2]:PATH:DEFine:PORT?
ROUTe[1|2]:PATH:DEFine:PORT <num1>, <num2>
ROUTe[1|2]:REFLection:DEFine:PORT <num>
```

ROUTe[1|2]:TRANsmission:DEFine:PORT <num>

SENSe[1|2]:AVERage:CLEar

SENSe[1|2]:AVERage:COUNt

SENSe[1|2]:AVERage[:STATe]

SENSe[1|2]:BWIDth[:RESolution]

SENSe[1|2]:CORRection:CLASs[:SELect]?

SENSe[1|2]:CORRection:COLLect[:ACQuire]

SENSe[1|2]:CORRection:COLLect[:ACQuire] STANdard1-7

SENSe[1|2]:CORRection:COLLect:CKIT:PORT[1|12]
[:SELECT]

SENSe[1|2]:CORRection:COLLect:IState[:AUTO]

SENSe[1|2]:CORRection:COLLect:METHod

SENSe[1|2]:CORRection:COLLect:METHod TWOPort

SENSe[1|2]:CORRection:COLLect:SAVE

SENSe[1|2]:CORRection:CSET[:SELect]

SENSe[1|2]:CORRection[:STATe]

SENSe[1|2]:CORRection:ONEPort:REFLection[:IMMediate]

SENSe[1|2]:CORRection:ONEPort:TRANSmission
[:IMMediate]

SENSe[1|2]:CORRection:TWOPort[:IMMediate]

SENSe:COUPle

SENSe[1|2]:DETector[:FUNCtion]

SENSe[1|2]:DISTance:STARt (Option 100 only)

SENSe[1|2]:DISTance:STOP (Option 100 only)

SENSe[1|2]:FREQuency:CENTer

SENSe[1|2]:FREQuency:MODE (Option 100 only)

SENSe[1|2]:FREQuency:SPAN

SENSe[1|2]:FREQuency:SPAN:MAXimum

SENSe[1|2]:FREQuency:STARt

```
SENSe[1|2]:FREQuency:STOP
SENSe[1|2]:FUNCtion
SENSe[1|2]:FUNCtion 'FLOC . . .
SENSe[1|2]:FUNCtion 'SRL . . .
SENSe[1|2]:FUNCtion 'XFR:GDEL:RAT . . .
SENSe[1|2]:FUNCtion 'XFR:POW . . .
SENSe[1|2]:FUNCtion 'XFR:POW:RAT . . .
SENSe[1|2]:FUNCtion 'XFR:S . . .
SENSe[1|2]:FUNCtion:SRL:SCAN[:IMMediate] (Option 100
only)
SENSe:ROSCillator:SOURce
SENSe[1|2]:STATe
SENSe[1|2]:SWEep:POINts
SENSe[1|2]:SWEep:TIME
SENSe[1|2]:SWEep:TIME:AUTO
SENSe:SWEep:TRIGger:SOURce
SOURce[1|2]:POWer[:LEVel][:IMMediate][:AMPLitude]
SYSTem:PRESet
TRACe[:DATA]
TRACe:CORRection:SIMulate:SAVE-?<TRANsmission1|...>
TRIGger[:SEQuence]:SOURce
```

Controlling Execution of Overlapped Commands

Each overlapped command is executed in two stages: initiation and completion. When both stages are complete for a given command, the command has "completed execution."

`*WAI`	Holds off the processing of subsequent commands until the initiation stage of all preceding commands is finished. If used after each overlapped command, this command ensures that commands in the analyzer's input queue complete initiation in the order received. Use of the *WAI command is explained later in this section and is demonstrated in the `SETUP` example program.
`*OPC?`	Places a 1 in the analyzer's output queue when all preceding commands have completed execution. If the program reads the output queue before it continues, this effectively pauses the controller until all executing overlapped commands are completed. This command is generally preferred to *WAI for control of command execution. Use of the `*OPC?` command is explained later in this chapter and is demonstrated in the `TRANCAL` and `REFLCAL` example programs.
`*OPC`	Sets bit 0 of the Standard Event Status event register to 1 when all preceding commands have completed execution. The analyzer's status registers can then be used to generate a service request when all overlapped commands are completed. This synchronizes the controller to the completion of an overlapped command, but also leaves the controller free to perform other tasks while the command is executing within the analyzer.

NOTE

*OPC only informs you when all currently executing commands have completed execution. It does not hold off the processing of subsequent commands. No commands should be sent to the analyzer between sending the *OPC command and receiving the service request. Any commands sent will be executed and may affect how the instrument responds to the previously sent *OPC.

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

NOTE

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

CAUTION

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

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

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

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

Refer to "Taking Sweeps" in the *Example Programs Guide* for more information.

Using *WAI and *OPC?

*WAI

The following example describes the use of the *WAI command. For this discussion, remember that a sequential command holds off the processing of subsequent commands until it has been completely processed. An overlapped command does not.

```
10 OUTPUT @Rfna;"command1"
20 OUTPUT @Rfna;"command2;*WAI"
30 OUTPUT @Rfna;"command3;"
40 OUTPUT @Rfna;"command4"
50 END
```

In the example above, commands are sent and completed in the following order:

- Commands 1 through 4 are sent to the analyzer as fast as the GPIB bus traffic will allow. The program sending the commands may very well end before any command has been completed.
- Command 1 begins execution first.
- If both commands 1 and 2 are overlapped types, the order in which they finish initiation depends on the commands. The order of completion is unknown.
- Commands 3 and 4 will not be started until both commands 1 and 2 have finished initiation.
- Command 3 will begin execution before command 4.
- If all four commands are overlapped types, the order in which they complete execution is unknown.

NOTE Because *WAI only controls the order of the initiation stage of commands, rather than the order of completion, it is strongly recommended that *OPC? be used whenever sequential operation of overlapping commands is required.

*OPC

The following example describes the use of the `*OPC?` command. For this discussion, remember that a sequential command holds off the processing of subsequent commands until it has been completely processed. An overlapped command does not.

```
10 OUTPUT @Rfna;"command1"
20 OUTPUT @Rfna;"command2;*OPC?"
30 ENTER @Rfna;Opc_done
40 OUTPUT @Rfna;"command3;"
50 OUTPUT @Rfna;"command4;*OPC?"
60 ENTER @Rfna;Opc_done
70 END
```

In the example above, commands are sent and completed in the following order:

- Commands 1 and 2 are sent to the analyzer as fast as the GPIB bus traffic will allow.
- Command 1 will begin execution before command 2.
- If both commands 1 and 2 are overlapped commands, the order of command completion is unknown.
- When both commands 1 and 2 have completed execution, commands 3 and 4 will be sent to the analyzer as fast as the GPIB bus traffic will allow.
- Command 3 will begin execution before command 4.
- If both commands 3 and 4 are overlapped commands, the order of command completion is unknown.
- This program will not end until the Opc_done, located in line 60, is returned indicating that both commands have completed execution.

Use `*OPC?` to ensure commands complete before proceeding. This can be done by calling a subroutine that issues the `*OPC?` command, and reads the analyzer response with `ENTER`:

```
100 Command_done !Example of subroutine using *OPC?
110 OUTPUT @Rfna;"*OPC?"
120 ENTER @Rfna;Opc_done
130 RETURN
```

Call the Command_done subroutine after each overlapped command to ensure the desired order of command execution.

3 Passing Control

Passing Control

When an external controller is connected to the analyzer with a GPIB cable, passing control may be needed to control devices such as printers and plotters that are also connected on the GPIB. For some operations the active controller must pass control to the analyzer. When the analyzer completes the operation, it automatically passes control of the bus back to the external controller.

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

NOTE Pass Control is not needed to control peripherals connected to the serial, parallel, or LAN ports.

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

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

The following is a procedure for passing control:

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

NOTE For this procedure to work properly, only the command that requires control of the bus should be pending. Other overlapped commands should not. For more information on overlapped commands, see Chapter 2, "Synchronizing the Analyzer and a Controller."

4 Data Types and Encoding

Data Types and Encoding

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

- the different data types used during data transfers
- data encoding used during transfers of numeric block data

Data Types

The analyzer uses a number of different data types during data transfers. Data transfer occurs in response to a query. The data type used is determined by the parameter being queried. Data types described in this section are:

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

Numeric Data

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

NR1 data Integers (such as `+1`, `0`, `-1`, `123`, `-12345`). This is the response type for boolean parameters as well as some numeric parameters.

NR2 data Floating point numbers with an explicit decimal point (such as `12.3`, `+1.234`, `-0.12345`).

NR3 data Floating point numbers in scientific notation (such as `+1.23E+5`, `+123.4E-3`, `-456.789E+6`).

Character Data

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

String Data

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

Expression Data

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

Block Data

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

Definite Block Length

The general form for a definite block length transfer is:

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

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

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

NOTE

This analyzer will send an additional <$^{C}{}_{R}$> with EOI asserted for definite block length transfers. The definite length block form for your analyzer is: #<num_digits><num_bytes><data_bytes><$^{C}{}_{R}$><EOI>.

<num_bytes> is the number of <data_bytes> without counting <$^{C}{}_{R}$><EOI>.

Indefinite Block Length

The general form for an indefinite block length transfer is:
#0<data_bytes><$^{C}{}_{R}$><EOI>

After the last data byte is sent, the indefinite length block must be terminated by sending a carriage return or newline with EOI asserted. This forces the termination of the program message. An example IBASIC (or HP BASIC) statement to send ABC+XYZ as an indefinite block length parameter is shown; note that END is used to properly terminate the message.

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

NOTE

Files are transferred as indefinite length blocks.

Data Encoding for Large Data Transfers

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

`REAL` — specifies the block data type. Either the definite or indefinite length syntax can be used. The block is transferred as a series of binary-encoded floating-point numbers. Data transfers of the `REAL,64` data type are demonstrated in the `REALDATA` example program.

`INTeger` — specifies the block data type. Either the definite or indefinite length syntax can be used. The block is transferred as an array of binary-encoded data with each point represented by a set of four 16-bit integers. This is the instrument's internal format — it should only be used for data that will be returned to the instrument for later use. Data transfers of the `INTeger 16` data type are demonstrated in the `INTDATA` and `LOADCALS` example programs.

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

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

```
PROGram[:SELected]:DEFine

SYSTem:SET
```

CAUTION INTeger 16 data for the Agilent 8711/12/13/14/ A-, B-, and C-series instruments is represented by sets of three 16-bit integers. The Agilent 8712ET/ES and 8714ET/ES instruments use sets of four 16-bit integers.

ASCII Encoding

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

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

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

Binary Encoding

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

`FORMat:DATA REAL,32`	selects the IEEE 32-bit format (not supported by IBASIC or HP BASIC)
`FORMat:DATA REAL,64`	selects the IEEE 64-bit format.
`FORMat:DATA INTeger,16`	selects the 16-bit integer format.

Byte Swapping

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

`FORMat:BORDer SWAPped`	selects the byte-swapped format
`FORMat:BORDer NORMal`	selects the standard format

5 Using Status Registers

Using Status Registers

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

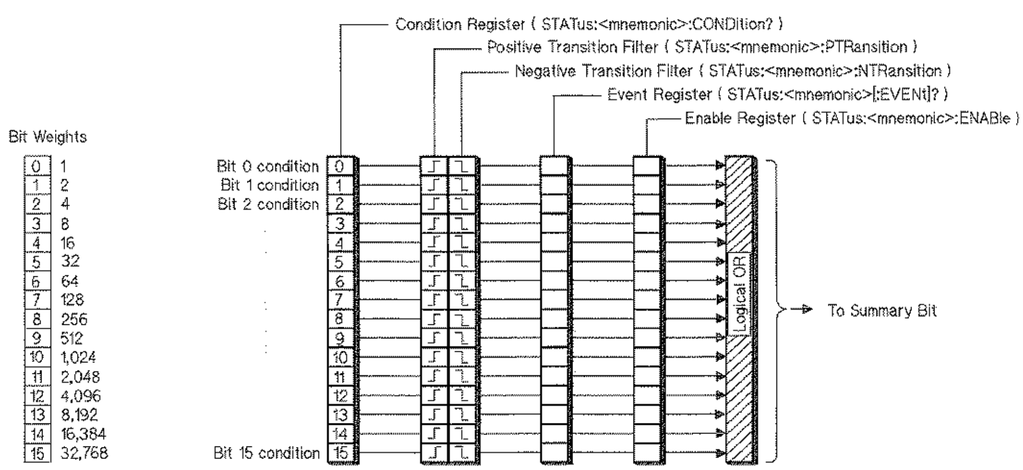
Example programs using the status registers are included in the *Example Programs Guide*. These programs include `SRQ` and `GRAPHICS` which use service request interrupt routines, `PASSCTRL` which uses the status byte to request control of the GPIB, and `LIMITEST` which uses the Limit Fail condition register.

General Status Register Model

The analyzer's status system is based on the general status register model shown in Figure 5-1. Most of the analyzer's register sets include all of the registers shown in the model, although commands are not always available for reading or writing a particular register. The information flow within a register set starts at the condition register and ends at the register summary bit (see Figure 5-2 on page 5-5 for actual connections between the registers). This flow is controlled by setting bits in the transition and enable registers.

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

Figure 5-1 **General Status Register Model**



Condition Register

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

Transition Registers

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

Event Register

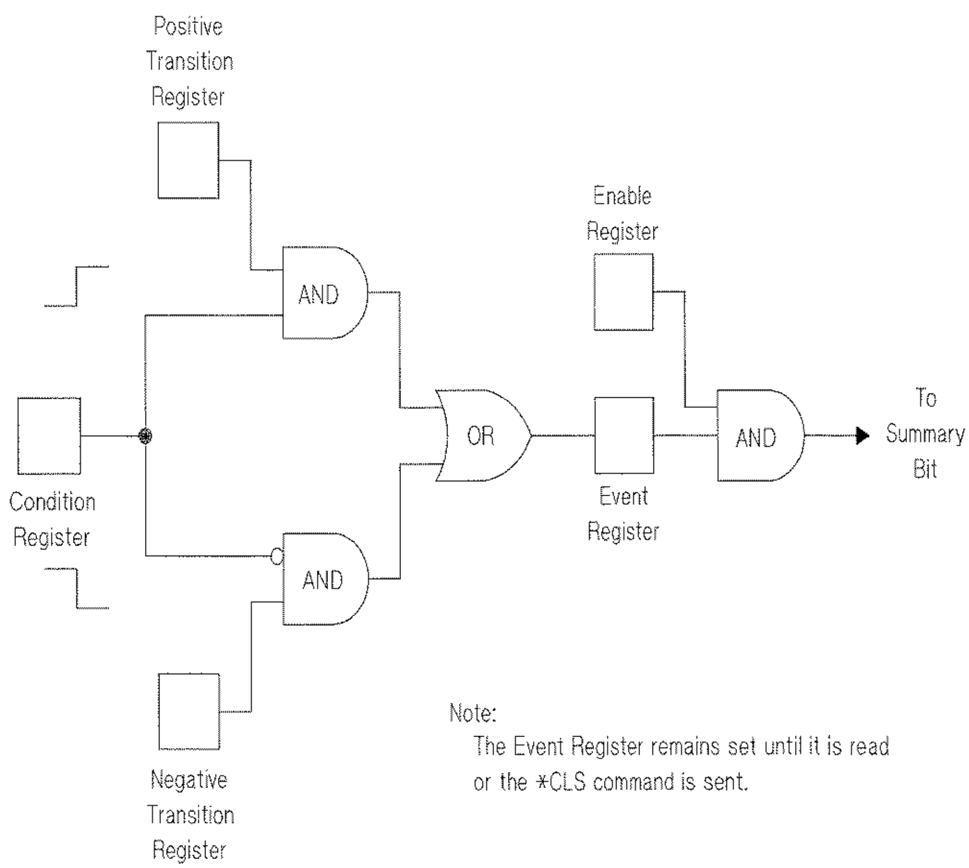
Event registers latch any reported condition changes. When a transition bit allows a condition change to be reported, the corresponding event bit is set to 1. Once set, an event bit is no longer affected by condition changes. It remains set until the event register is cleared. Event registers are read-only.

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

Enable Register

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

Figure 5-2 **Flow of Information Within a Register Set**



How to Use Registers

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

- the direct-read method
- the service request (SRQ) method

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

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

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

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

The Service Request Process

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

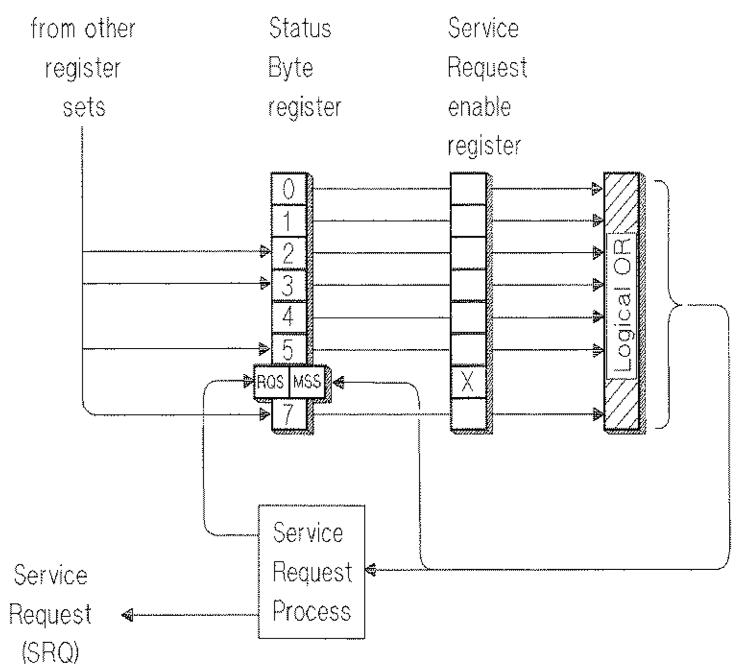
1. Determine which bit monitors the condition.
2. Determine how that bit reports to the request service (RQS) bit of the Status Byte.
3. Send GPIB commands to enable the bit that monitors the condition and to enable the summary bits that report the condition to the RQS bit.
4. Enable the controller to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

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

Generating a Service Request

A service request is generated using the Status Byte. As shown in Figure 5-3, the analyzer's other register sets report to the Status Byte. Some of them report directly while others report indirectly through other register sets.

Figure 5-3 **Generating a Service Request**



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

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

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

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

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

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

As implied in Figure 5-3, bit 6 of the Status Byte register serves two functions: the request service function (RQS) and the master summary status function (MSS). Two different methods for reading the register allow you to access the two functions. Reading the register with a serial poll allows you to access the bit's RQS function. Reading the register with `*STB` allows you to access the bit's MSS function.

The Analyzer's Status Register Sets

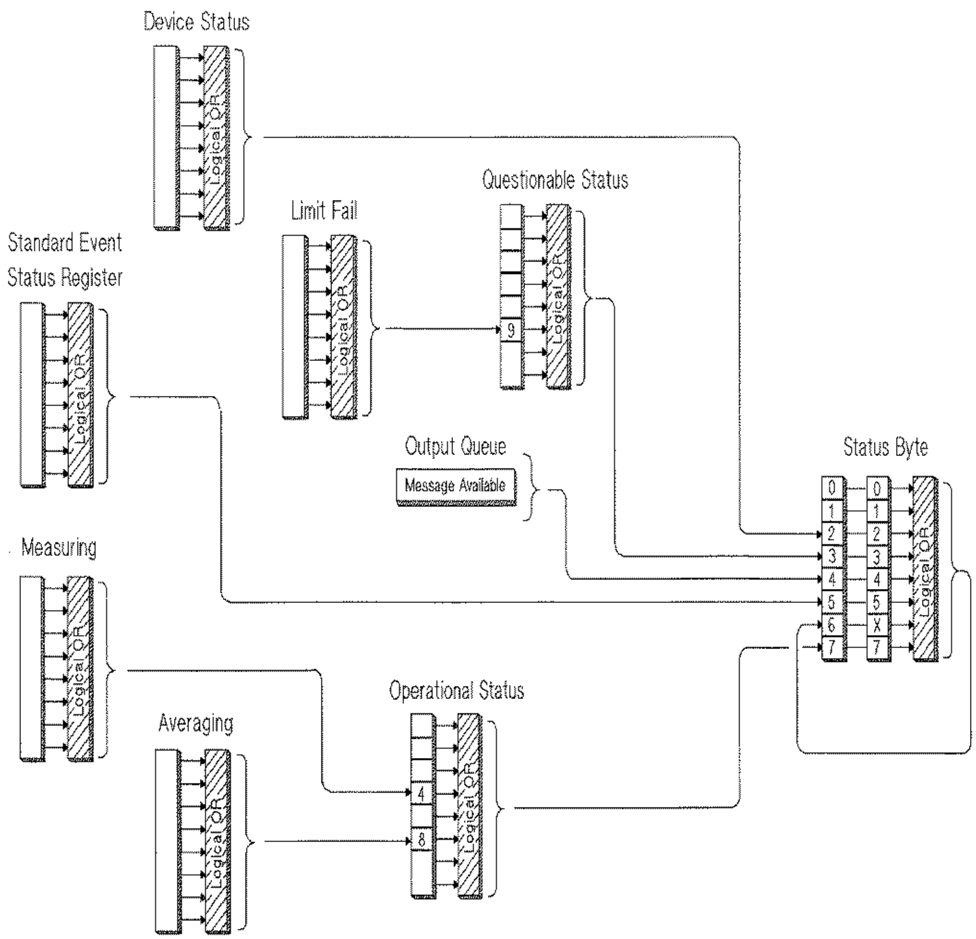
The analyzer uses eight register sets to keep track of instrument status:

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

Their reporting structure is summarized in Figure 5-4. They are described in greater detail in the following section.

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

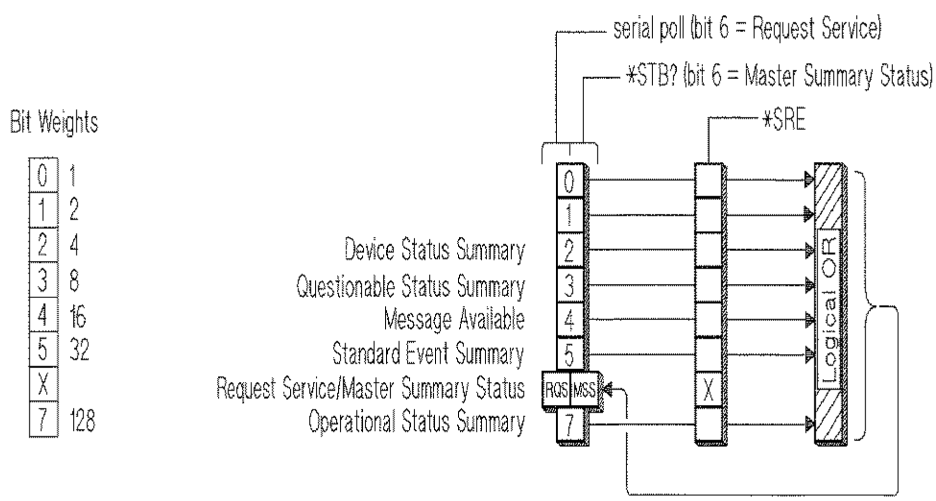
Figure 5-4 **Analyzer Register Sets**



Status Byte

The Status Byte register set summarizes the states of the other register sets and monitors the analyzer's output queue. It is also responsible for generating service requests see "Generating a Service Request" on page 5-8. See Figure 5-5.

Figure 5-5 **The Status Byte Register Set**



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

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

Device Status Summary
(bit 2) is set to 1 when one or more enabled bits in the Device Status event register are set to 1.

Questionable Status Summary
(bit 3) is set to 1 when one or more enabled bits in the Questionable Status event register are set to 1.

Message Available
(bit 4) is set to 1 when the output queue contains a response message.

Standard Event Status Summary
(bit 5) is set to 1 when one or more enabled bits in the Standard Event Status event register are set to 1.

Master Summary Status
(bit 6, when read by `*STB`) is set to 1 when one or more enabled bits in the Status Byte register are set to 1.

Request Service
(bit 6, when read by serial poll) is set to 1 by the service request process (see "Generating a Service Request" on page 5-8).

Operational Status Summary
(bit 7) is set to 1 when one or more enabled bits in the Operational Status event register are set to 1.

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

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

Device Status Register Set

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

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

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

Any Softkey Pressed
(bit 1) is set to 1 when one of the analyzer's softkeys has been pressed.

Any External Keyboard Key Pressed
(bit 2) is set to 1 when a key has been pressed on an external keyboard connected to the DIN KEYBOARD connector on the rear panel of the analyzer.

Front Panel Knob Turned
(bit 3) is set to 1 when the analyzer's front panel knob is turned.

Limit Fail Register Set

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

The inputs for the bits in the Limit Fail condition register are latched. (See Figure 5-6.) The two bits for measurement channel 1 are latched when the Limit Test is OFF for channel 1 or when MEAS 1 is OFF. The two bits for measurement channel 2 are latched when Limt Test is OFF for channel 2 or when MEAS 2 is OFF.

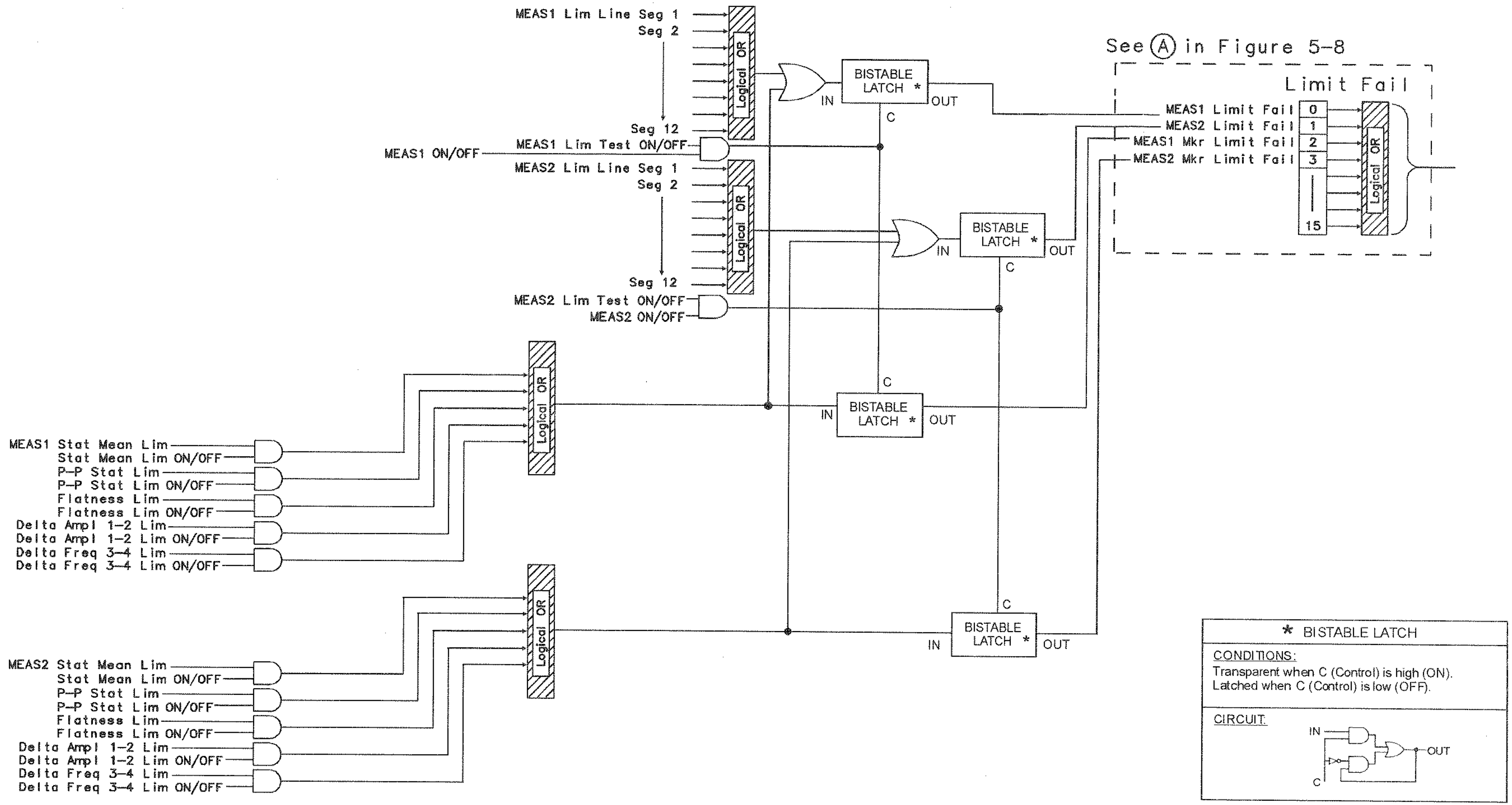
The following conditions determine the state for each of the bits when the corresponding Limit Test is ON.

Measurement Channel 1 Limit Failed
(bit 0) is set to 1 when limit testing is enabled and any point on measurement channel 1 fails the limit test, *or* when any enabled marker limit on measurement channel 1 has failed.

Measurement Channel 2 Limit Failed
(bit 1) is set to 1 when limit testing is enabled and any point on measurement channel 2 fails the limit test, *or* when any enabled marker limit on measurement channel 2 has failed.

Measurement Channel 1 Marker Limit Failed
(bit 2) is set to 1 when any enabled marker limit on measurement channel 1 has failed.

Measurement Channel 2 Marker Limit Failed
(bit 3) is set to 1 when any enabled marker limit on measurement channel 2 has failed.


MEAS1 Lim Line Seg 1
Seg 2
Seg 12
MEAS1 ON/OFF
MEAS1 Lim Test ON/OFF
MEAS2 Lim Line Seg 1
Seg 2
Seg 12
MEAS2 Lim Test ON/OFF
MEAS2 ON/OFF
Logical OR
BISTABLE LATCH *
IN
OUT
C
See (A) in Figure 5-8
Limit Fail
MEAS1 Limit Fail
MEAS2 Limit Fail
MEAS1 Mkr Limit Fail
MEAS2 Mkr Limit Fail
0
1
2
3
15
MEAS1 Stat Mean Lim
Stat Mean Lim ON/OFF
P-P Stat Lim
P-P Stat Lim ON/OFF
Flatness Lim
Flatness Lim ON/OFF
Delta Ampl 1-2 Lim
Delta Ampl 1-2 Lim ON/OFF
Delta Freq 3-4 Lim
Delta Freq 3-4 Lim ON/OFF
MEAS2 Stat Mean Lim
Stat Mean Lim ON/OFF
P-P Stat Lim
P-P Stat Lim ON/OFF
Flatness Lim
Flatness Lim ON/OFF
Delta Ampl 1-2 Lim
Delta Ampl 1-2 Lim ON/OFF
Delta Freq 3-4 Lim
Delta Freq 3-4 Lim ON/OFF
* BISTABLE LATCH
CONDITIONS:
Transparent when C (Control) is high (ON).
Latched when C (Control) is low (OFF).
CIRCUIT:
IN
C
OUT

cw61e

Fig. 5-6

Questionable Status Register Set

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

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

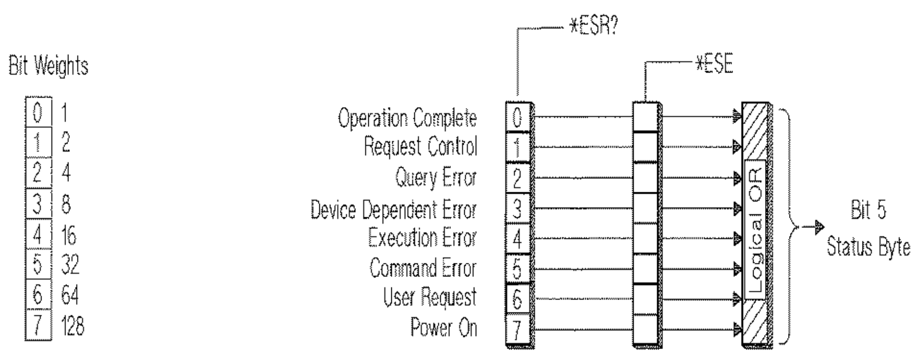
Limit Fail
(bit 9) is set to 1 when one or more enabled bits in the Limit Fail event register are set to 1.

Data Questionable
(bit 10) is set to 1 when a change in the analyzer's configuration requires that new measurement data be taken.

Standard Event Status Register Set

The Standard Event Status register set monitors GPIB errors and synchronization conditions. See Figure 5-7.

Figure 5-7 **The Standard Event Status Register Set**



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

Operation Complete
(bit 0) is set to one when the following two events occur (in the order listed):

1. The *OPC command is sent to the analyzer.
2. The analyzer completes all pending overlapped commands.

Request Control
(bit 1) is set to 1 when both of the following conditions are true:

- The analyzer is configured as a talker/listener for GPIB operation.
- The analyzer is instructed to do something (such as plotting or printing) that requires it to take control of the bus.

Query Error
(bit 2) is set when the command parser detects a query error. A query error indicates that one or both of the following actions occurred:

- an attempt to read data from the Output Queue when no data was present.
- that data in the Output Queue was lost. An example of this would be queue overflow.

Device Dependent Error
(bit 3) is set to 1 when the command parser detects a device-dependent error. A device-dependent error is any analyzer operation that did not execute properly due to some internal condition such as overrange. This bit indicates that the error was not a command, query, or an execution error.

Execution Error

(bit 4) is set to 1 when the command parser detects an execution error. Execution errors occur when the following conditions occur:

- a <PROGRAM DATA> element received in a command was outside the legal range for the analyzer, or inconsistent with the operation of the analyzer.
- the analyzer could not execute a valid command due to some analyzer condition.

Command Error

(bit 5) is set to 1 when the command parser detects a command error. The following events cause a command error:

- An IEEE 488.2 syntax error occurred. This means that the analyzer received a message that did not follow the syntax defined by the 488.2 standard.
- A semantic error occurred. For example, the analyzer received an incorrectly spelled command. Another example would be that the analyzer received an optional 488.2 command that it does not implement.

User Request

(bit 6) is not implemented. For keypress related functions, see "Device Status Register Set" on page 5-15.

Power On

(bit 7) is set to 1 when you turn on the analyzer.

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

`*ESR?`	reads the value of the standard event status register.
`*ESE <num>`	sets bits in the standard event status enable register. The current setting of the standard event statue enable register is stored in non-volatile memory. If `*PSC` has been set, it will be saved at power on.
`*ESE?`	reads the current state of the standard event status enable register.

Measuring Status Register Set

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

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

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

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

Averaging Status Register Set

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

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

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

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

Operational Status Register Set

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

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

Calibrating (bit 0) is set to 1 while the instrument is zeroing the broadband diode detectors.

Settling (bit 1) is set to 1 while the measurement hardware is settling.

Measuring (bit 4) is set to 1 when one or more enabled bits in the Measuring Status event register are set to 1.

Correcting (bit 7) is set to 1 while the analyzer is performing a calibration function.

Averaging (bit 8) is set to 1 when one or more enabled bits in the Averaging Status event register are set to 1.

Hardcopy Running (bit 9) is set to 1 while the analyzer is performing a hardcopy (print or plot) function.

Test Running (bit 10) is set to 1 when one of the analyzer's internal service tests is being run.

Program Running (bit 14) is set to 1 while an HP Instrument BASIC program is running on the analyzer's internal controller.

Settings for STATus:PRESet

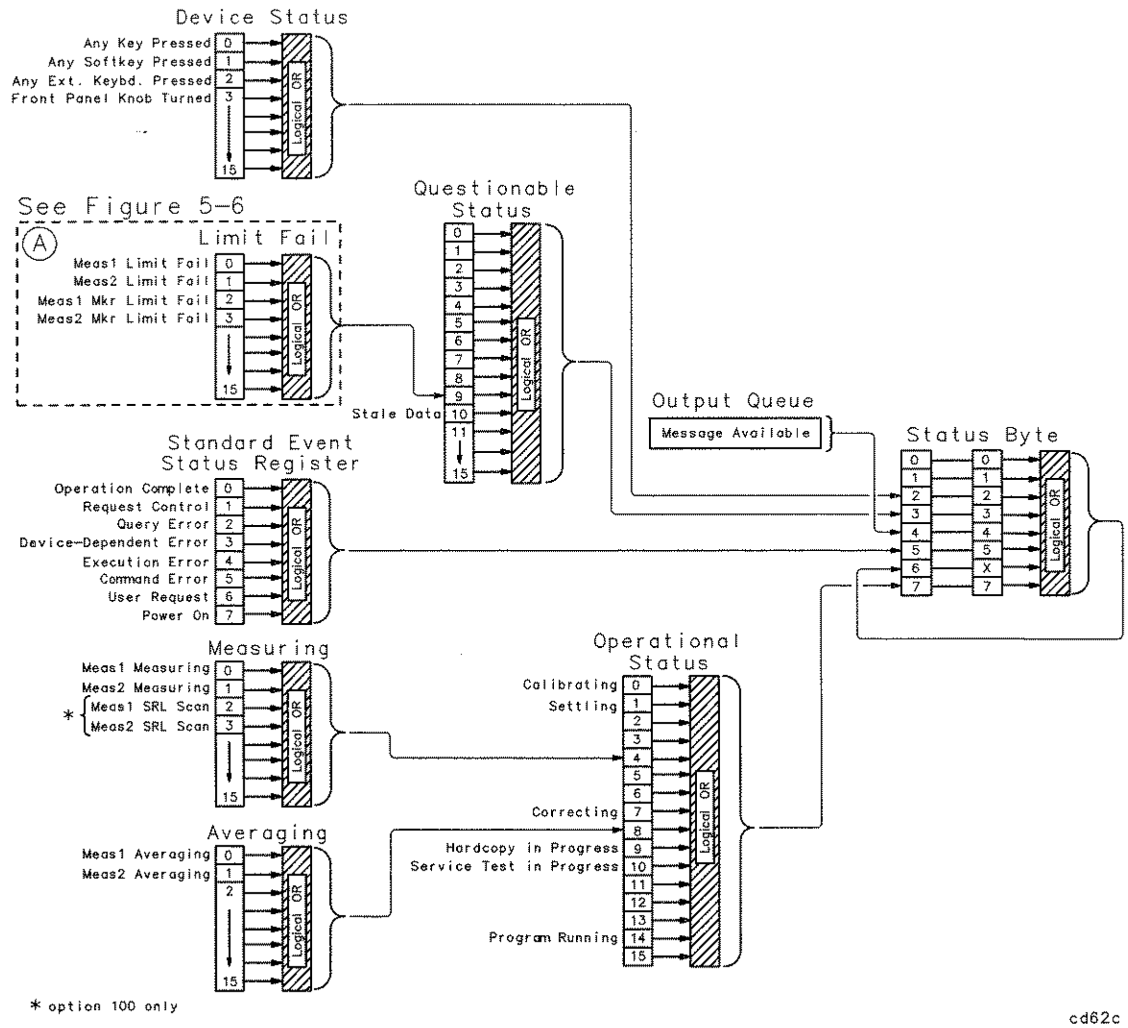
Executing the STATus:PRESet command changes the settings in the enable (ENAB), positive transition (PTR), and negative transition (NTR) registers. The table below shows the settings after the command is executed.

Table 5-1 Status Register States After PRESet Command

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

Analyzer Register Set Summary

Figure 5-8 **Register Set Summary**



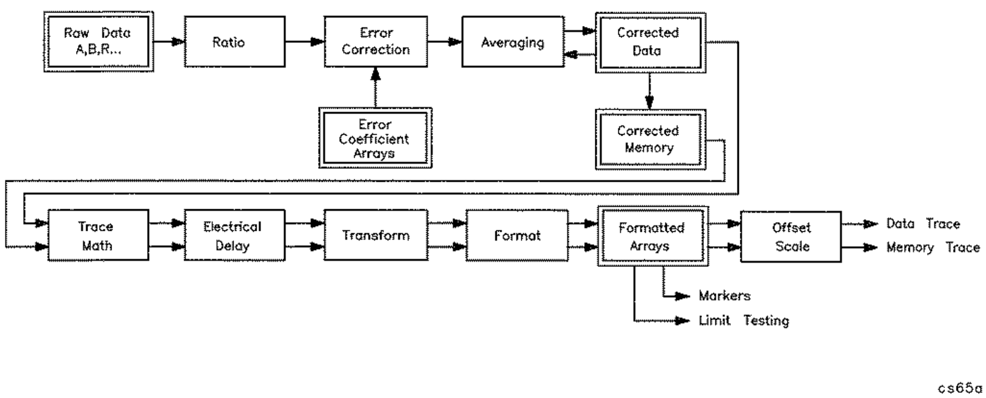
6 Trace Data Transfers

Trace Data Transfers

This chapter explains how to read (query) the measurement data trace from the analyzer into your program. It also describes how to send data from your program to the analyzer's measurement arrays. Accessing the measurement arrays is done using SCPI commands. If you are using IBASIC, you can also access the measurement arrays using high-speed subroutines. Refer to the *HP Instrument BASIC User's Handbook* for more details.

Figure 6-1 is a data processing flow diagram that represents the flow of numerical data. The data passes through several math operations, denoted in the figure by single-line boxes. Most of these operations can be selected and controlled with the front panel CONFIGURE block menus. The data is stored in arrays along the way, denoted by double-line boxes. These arrays are places in the flow path where data is accessible via GPIB. While only a single flow path is shown, two identical paths are available, corresponding to measurement channels 1 and 2.

Figure 6-1 **Numeric Data Flow Through the Network Analyzer**



Querying the Measurement Trace Using BASIC

After making a measurement, you can read the resultant measurement trace out of the analyzer using the SCPI query:

```
"TRACE:DATA?  CH1FDATA"
```

The BASIC program segment below shows how to read the trace from the analyzer into an array in your program.

```
10    REAL Trace(1:201)
20    ASSIGN @Hp8711 TO 716
30    ! Take sweep here
40    OUTPUT @Hp8711;"FORM:DATA ASCII,5"
50    OUTPUT @Hp8711;"TRACE:DATA? CH1FDATA"
60    ENTER @Hp8711;Trace(*)
70    DISP Trace(1),Trace(2),Trace(3),". . . ."
```

In this program, the `TRACE:DATA?` query returns all of the measurement points as a single block. The analyzer computes the value for each point using the measurement format selected by the [FORMAT] menu (`CALC:FORM SCPI` command), and returns a block of data called the formatted data array. The values of each point correspond to the values displayed on the screen, or those shown in the marker readouts. The frequency stimulus value (X-axis) of each point is not returned by the `TRACE:DATA?` query; only the measurement response (Y-axis) values are returned.

When transferring the block of trace data, you may select either binary or ASCII data encoding. This is explained in Chapter 4, “Data Types and Encoding,” in the section titled “Data Encoding for Large Data Transfers” on page 4-6. Notice that the terms "encoding format" and "measurement format" are not the same. The encoding format determines how the numbers are represented as bytes, while the measurement format corresponds to the meaning of the value of the numbers.

The easiest way to transfer a measurement data trace is to use ASCII data encoding.

In the previous BASIC program segment, the array Trace(1:201) contains 201 real (floating point) numbers. The SCPI command `"FORM:DATA ASCII,5"` specifies ASCII data encoding, with 5 significant digits. The command `"TRACE:DATA? CH1FDATA"` instructs the analyzer to send the measurement trace. The `ENTER` statement reads the measurement data sent by the analyzer into the Trace(1:201) array.

It is important to make sure that the Trace array declared in your program is the same size as the measurement trace on the analyzer, or an error will occur. The ENTER statement attempts to read data from the analyzer until it completely fills the Trace array, at which point it expects to receive an end-of-data terminator from the analyzer. To be safe, your program should use the `"SENS:SWE:POIN"` SCPI command to set the number of measurement data points to the desired value.

Refer to the example program ASCDATA in the *Example Programs Guide* for a complete example.

Smith Chart and Polar Formats

Each measurement point is represented by a single floating point number. This is the case for all of the analyzer's measurement formats except Smith Chart and Polar. When Smith Chart or Polar format is selected, each point is represented by two numbers, the first one being the real portion and the second being the imaginary portion of the complex measurement value.

Below is a modified example program that will work when using Smith Chart or Polar formats.

```
10   REAL Trace(1:201,1:2)
20   ASSIGN @Hp8711 TO 716
30   ! Take sweep here
40   OUTPUT @Hp8711;"FORM:DATA ASCII,5"
50   OUTPUT @Hp8711;"TRACE:DATA? CH1FDATA"
60   ENTER @Hp8711;Trace(*)
70   DISP Trace(1,1),Trace(1,2),". . . .",Trace(201,1),Trace(201,
2)
```

Querying the Measurement Trace Using SICL

This section includes a complete SICL C program that shows how to read the measurement trace from the analyzer.

```
/***************************************************************************
 * This program takes a sweep, reads the trace, and prints it.
 * It uses SICL (Standard Instrument Control Library) to talk
 * to the analyzer over HP-IB.
 *
 * On HP-UX, compile using:    cc -Aa -o query_trace  query_trace.c -lsicl
 **************************************************************************/
#include <sicl.h>           /* For iopen(), iprintf(), iscanf(), INST, ... */
#include <stdio.h>          /* For printf() */

int main(void) {
    INST analyzer;              /* Handle used to talk to analyzer */
    float data_buf[1601];       /* measurement trace.  32-bit floats */
    int num_trace_bytes;
    int pt;
    num_trace_bytes = sizeof(data_buf);    /* Set to max allowable bytes */

    /* Open the network analyzer at address 16 */
    analyzer = iopen("hpib,16");

    /* Clear the bus */
    iclear(analyzer);

    /* Abort current sweep and put analyzer sweep in hold */
    iprintf(analyzer, "ABORT\n");
    iprintf(analyzer, "INIT:CONT OFF\n");

    /* Take one sweep, wait until done */
    iprintf(analyzer, "INIT1\n");
    iprintf(analyzer, "*OPC?\n");
    iscanf(analyzer, "%*s");

    /* Request the trace data in 32-bit floating point format */
    iprintf(analyzer, "FORM:BORD NORM\n");
    iprintf(analyzer, "FORM:DATA REAL,32\n");

    /* Query the trace, read into data_buf[]. */
    iprintf(analyzer, "TRAC? CH1FDATA\n");
    iscanf(analyzer, "%#b%*c", &num_trace_bytes, &data_buf[0]);

    /* Print the trace values. */
    for (pt = 0; pt < num_trace_bytes/sizeof(float); pt++) {
        printf("%4d     %g\n", pt, data_buf[pt]);
    }
    /* Close analyzer and exit. */
    iclose(analyzer);
    return 0;
}
```

Using Binary Data Encoding

The previous section describes how to query the measurement trace, and transfer it into your program using ASCII encoding. Binary encoding can be used for faster data transfers, as shown in the table below:

Table 6-1 **Trace Transfer Times (typical)**

Number of Trace Points	Transfer Times (ms)	
	Binary Transfer	ASCII Transfer
51	21	47
201	23	164
401	30	314
1601	82	1200

When using binary data transfers, the entire trace is sent from the analyzer to your program in a block called a definite length block. The details of block data are described in detail in Chapter 4, "Data Types and Encoding." The definite length block contains a header and a data section. The header indicates how many bytes are in the data section.

In order to read the definite length block, your program must first read the header, and then read the data section. Refer to the example program REALDATA in the *Example Programs Guide* for an example of how to do this.

In the REALDATA program, you will notice the following lines which read the definite block header:

```
180    ENTER @Hp8711 USING "%,A,D";A$,Digits
190    ENTER @Hp8711 USING "%,"&VAL$(Digits)&"D";Bytes
```

and these lines which read the data section:

```
200    ASSIGN @Hp8711;FORMAT OFF
210    ENTER @Hp8711;Data1(*)
```

Each measurement point in the data section is represented as 4 or 8 bytes (32 or 64 bits), depending on whether single precision or double precision numbers are requested. When using HP BASIC or IBASIC, you must select double precision numbers to match BASIC's "REAL" data type. Do this using the SCPI command `"FORM:DATA REAL,64"`. If you are using another language that supports single precision data types, you can select single precision using the SCPI command `"FORM:DATA REAL,32"`. Languages such as QuickBASIC and C have support for both single and double precision floating point numbers.

When transferring data using binary encoding, you may need to reverse the order of the bytes in each measurement point, since PCs frequently store IEEE floating point numbers with the byte order reversed. To instruct the analyzer to reverse the byte order of the data, send the command `"FORMAT:BORDer SWAPped"` before querying the trace data.

Trace Data Transfer Sizes

The following table shows how many bytes are transmitted during trace data transfers. The left column shows the format of the data, which you can specify using the SCPI command `Format:DATA`. As you can see, the size of the measurement point data and trace data varies as you change format.

Table 6-2 Trace Data Transfer Size Using TRACE:DATA Command

Format Type (FORMat:DATA)	Type of Data	Size of Single Measurement Point (bytes)		Size of 201 Point Trace (bytes)	
		Real	Complex	Real	Complex
REAL,32	IEEE 32-bit Floating Point	4	8	809	1614
REAL,64	IEEE 64-bit Floating Point	8	16	1614	3222
ASCII,5	ASCII numbers	13	26	2613	5226
ASCII,3	ASCII numbers	11	22	2211	4422
INT,16	Internal Binary	—	8	—	1614

When transmitting data in "REAL" or "INT" format, a header is sent before the data block. The header indicates the size of the data block. The header size varies in length from 3 to 11 bytes. Refer to Chapter 4, "Data Types and Encoding," for details on the header.

Transmitting ASCII data requires no header. The ASCII values are separated by commas, and a linefeed is sent after the last value. The sizes shown in the table include the size of the comma(s) and terminating linefeed. Typical data in ASCII,5 format:

```
-1.2254E+000,+5.0035E-001,+4.5226E-001,...
```

The analyzer stores its internal data with approximately 5 significant digits of resolution. Using REAL,32 or ASCII,5 format provides sufficient precision for data transfers. However, REAL,64 may be necessary when using a programming language which does not support IEEE 32-bit floating point.

Transferring Data with IBASIC

If you are using IBASIC, your IBASIC program can avoid the overhead of using OUTPUT and ENTER to transfer trace data, and instead use the analyzer's built-in high-speed subprograms. These built-in subroutines let you quickly move data between the analyzer's measurement arrays and your program's data arrays. For example, to read the analyzer's formatted data array, use the following:

```
10 DIM Fmt(1:201)
20 INTEGER Chan
30 LOADSUB Read_fdata FROM "XFER:MEM 0,0"
40 Chan=1
50 Read_fdata(Chan,Fmt(*))
```

Refer to the *HP Instrument BASIC User's Handbook* for more details.

The table below compares the speed of IBASIC using high-speed transfer subroutines with that of a fast external controller using the SCPI TRACE:DATA? CH1FDATA query.

Table 6-3 High-Speed Trace Transfer Times

Number of Trace Points	Controller Using Binary TRACE:DATA? (ms)	IBASIC Using Read_fdata (ms)
51	21	7
201	23	10
401	30	13
1601	82	32

Taking Sweeps

When making measurements and querying traces, your program should perform the following steps:

1. Place the analyzer's sweep in hold.
2. Initiate a single sweep.
3. Wait for the sweep to complete.
4. Query the measurement trace.

Use the following program lines to perform these steps:

```
10 OUTPUT @Hp8711;"ABORT;:INIT1:CONT OFF"
20 OUTPUT @Hp8711;"INIT1"
30 OUTPUT @Hp8711;"*OPC?"
35 ENTER @Hp8711;Opc
40 OUTPUT @Hp8711;"TRACE:DATA? CH1FDATA"
45 ENTER @Hp8711;Fmt(*)
```

If you query the measurement trace while the analyzer is in continuous sweep, the query will still work, but the data may not be correct. Using `INIT` and `*OPC?` ensures that a complete sweep has finished before you query the measurement data. In many cases, you can also use the command "`*WAI`" in place of the "`*OPC?`" query, replacing lines 30 and 35 above with:

```
30 OUTPUT @Hp8711;"*WAI"
```

However, there are cases where "`*WAI`" will produce incorrect results. One case is when using IBASIC's high-speed subprograms to query the trace data. "`*WAI`" only ensures that the SCPI commands following the "`*WAI`" are not executed until the commands before the "`*WAI`" are complete. Since IBASIC subprograms don't use SCPI commands to access the trace data, "`*WAI`" is ineffective, and "`*OPC?`" should be used.

When using "`*OPC?`", the `ENTER` statement following the "`*OPC?`" will wait until the previous SCPI commands are complete, preventing your program from executing beyond the `ENTER` statement. When using "`*WAI`", your program can continue to run and send SCPI commands, and the analyzer will buffer them and act upon them in order.

Chapter 2, "Synchronizing the Analyzer and a Controller," provides additional details.

CALC:DATA? versus TRACE:DATA?

The SCPI command "`CALC1:DATA?`" is functionally equivalent to the command "`TRACE:DATA? CH1FDATA`". The two can be used interchangeably for trace queries of the formatted measurement data. The "`TRACE:DATA`" command is more flexible, allowing you to query other measurement arrays and to download data to measurement arrays.

Querying Single Data Points Using Markers

If you only need to query a single data point, you can use a marker query instead of a trace query. The program segment below shows how to do this using the SCPI command CALC:MARK.

```
10    ASSIGN @Hp8711 TO 716
20    ! Take sweep here
30    OUTPUT @Hp8711;"CALC1:MARK ON"    ! turn on marker
40    OUTPUT @Hp8711;"CALC1:MARK1:X 177 MHz"    ! set frequency
50    OUTPUT @Hp8711;"CALC1:MARK1:Y?"           ! read marker
60    ENTER @Hp8711;Marker_y
70    DISP Marker_y
```

You can also use the CALC:MARK:FUNC:RES? query to return the results of a bandwidth search. The following program steps accomplish these tasks:

```
10    ! Select -3 dB bandwidth
20    OUTPUT @Hp8711;"CALC:MARK:BWID -3"
30    ! Get result of bandwidth search
40    OUTPUT @Hp8711;"CALC:MARK:FUNC:RES?"
50    ENTER @Hp8711;Bwidth,Center_freq,Q,Loss
```

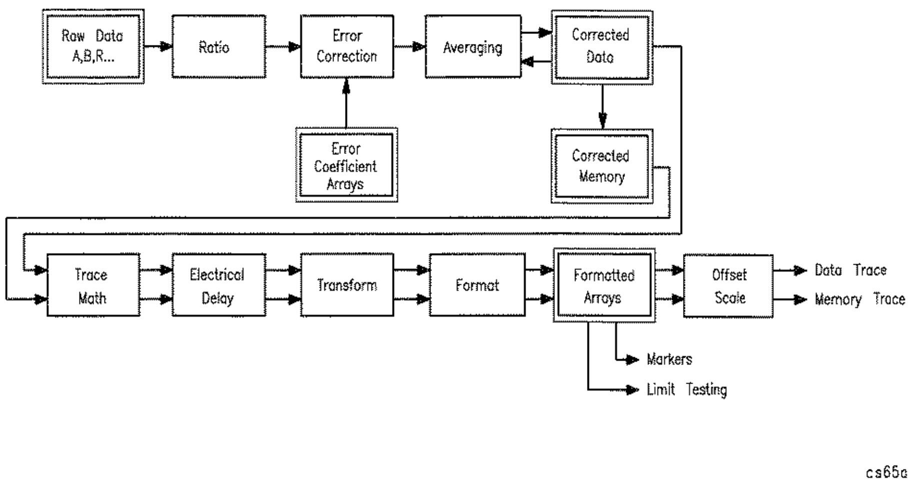
For more information on using markers, refer to the *Example Programs Guide*.

Accessing Other Measurement Arrays

The preceding sections describe how to query the formatted data array using the `TRACE:DATA?` query with the argument CH1FDATA. The formatted array is the last array in the analyzer's data processing chain, and is generally of most interest.

The analyzer also allows you to query other measurement arrays which are earlier in its data processing chain. Figure 6-2 shows the data processing chain.

Figure 6-2 **Numeric Data Flow Through the Network Analyzer**



The first array is the Raw Data Array, which contains each of the separate input components (A, B, R, B*, R*, X, Y, AUX) immediately after they are measured. These arrays can be queried and set, but doing so is of limited use, since the data values contained in the arrays are uncorrected, and are not directly correlated to any meaningful reference, such as 0 dBm.

The Error Coefficient arrays contain default correction values or values created during a measurement calibration. These arrays can be queried and set, but care should be exercised in setting them since incorrect measurements may result. If you wish to apply your own corrections in addition to the analyzer's current correction, the best technique is to use the Corrected Memory array and the Data/Memory feature, explained below.

The Corrected Data array contains the results of the currently selected measurement (Transmission, Reflection, etc.) after error correction and averaging have been applied. The measurement data in these arrays is represented as complex number pairs. When measuring the transmission response of a through cable, the magnitude of the complex numbers will be very close to 1.0. When measuring an open circuit, the magnitude of the complex numbers will be very close to 0.0. When measuring an amplifier, the magnitude of the complex numbers will be greater than 1.0.

The Corrected Memory array is filled with a copy of the Corrected Data array when the Data –> Memory operation is performed. It can be used to apply a gain correction to the measured data. This is described in the following section.

The Formatted Data array contains the measurement data after it has been formatted using the format selected by the [FORMAT] menu. Querying the Formatted Data array is described in detail at the beginning of this chapter. You can also download data to this array, and the analyzer will display the data using the current Scale and Reference values.

Applying Gain Correction Using the Memory Trace

The Corrected Memory array is filled with a copy of the Corrected Data array when the Data -> Memory operation is performed. By setting the analyzer to perform Data/Memory trace math, you can apply your own correction factor to the measurement data trace by filling the Corrected Memory array with the appropriate complex numbers.

In general, you should use the analyzer's calibration feature to correct for errors in your system. However, there may be cases where you wish to simulate the effect of adding a cable in series with your DUT, and observe how this imaginary cable will attenuate the measured response versus frequency. Or you may wish to apply an absolute offset to simulate the effect of adding or removing a pad from the measurement. These simulations are easily accomplished using the Corrected Memory array and the Data/Memory feature.

The Corrected Data and Memory arrays contain complex linear data, as opposed to logged data. When displaying the traces using Lin Mag format, the result of the Data divided by Memory operation (Data/Mem) will be to divide each point of the data trace by each point of the memory trace. When displaying data in Log Mag format, the result of Data/Memory will be equivalent to subtracting the Log Mag value of the Memory trace from that of the Data trace.

The following example BASIC code segment shows how to download a complex array from your program to the analyzer's Memory trace. The program's "Mem" array is initialized with the proper values such that when the analyzer computes Data divided by Memory, the desired increasing gain will be applied.

```
100    REAL Mem(1:201,1:2)
110    ASSIGN @Hp8711 TO 716
120    ! Fill memory array (denominator in Data/Mem)
130    ! with values that will result in an
140    ! upward sloping gain factor vs. frequency.
150    ! Used to compensate for cable loss vs. frequency
160    ! Adds 0 dB of gain at start freq; 3 dB at stop freq
170    FOR Pt=1 TO 201
180       Gain_factor_db=3.0*(Pt - 1)/200  ! 0..3 dB Power
190       Gain_factor_lin=10^(Gain_factor_db/20)
200       Mem(Pt,1)=1.0/Gain_factor_lin  ! real
210       Mem(Pt,2)=0.0                  ! imag
220    NEXT Pt
230    ! Download to the memory trace
240    OUTPUT @Hp8711;"FORM:DATA ASCII"
250    OUTPUT @Hp8711;"TRACE:DATA CH1SMEM";   ! Note the ";"
260    FOR Pt=1 TO 201
270       FOR I=1 TO 2
280          OUTPUT @Hp8711;",";Mem(Pt,I);    ! Note the ";"
290       NEXT I
300    NEXT Pt
310    OUTPUT @Hp8711;""     ! Send linefeed
320    OUTPUT @Hp8711;"CALC1:MATH (IMPL/CH1SMEM)"  ! Data/Mem
```

The example above downloads data to the corrected memory array. The data is sent by the program to the analyzer using ASCII encoding. The data is sent as ASCII characters, separated by commas. The analyzer accepts the comma separated list of numbers until it receives a linefeed to terminate the command. The program uses semicolons at the end of some OUTPUT statements to avoid sending a linefeed until all of the data has been sent. After the last number is sent, the program sends a linefeed, and the analyzer accepts the data.

Remember, for faster transfers, use binary data encoding instead of ASCII.

Performing Your Own Data Processing

After the analyzer has made a measurement, you can read the measurement trace and perform your own post-processing on it, and display the result on the screen. This is done using these steps:

1. Initiate a sweep.
2. Wait for the sweep to finish.
3. Read the measurement data into an array in your program.
4. Perform your post-processing on the measurement data.
5. Write (download) the post-processed data to the analyzer's memory trace.

You may want to instruct the analyzer to display only the memory trace and not the data trace, so that only your post-processed data is seen.

The program below demonstrates how to perform data post-processing. It takes the measurement data and reverses it, such that the low frequency data is displayed on the right end of the trace, and the high frequency data is displayed on the left.

```
100    ! Display the measurement data backwards
110    REAL Fmt(1:201)
120    ASSIGN @Hp8711 TO 716
130    !
140    OUTPUT @Hp8711;"FORM:DATA ASCII"
150    OUTPUT @Hp8711;"ABOR;INIT:CONT OFF;*WAI"
160    OUTPUT @Hp8711;"DISP:WIND:TRAC1 OFF;TRAC2 ON"
170    LOOP
180       ! Take sweep
190         OUTPUT @Hp8711;"INIT1;*WAI"
200         ! Read the trace from the formatted data array
210         OUTPUT @Hp8711;"TRACE:DATA? CH1FDATA"
220         ENTER @Hp8711;Fmt(*)
230         ! Download the trace, backwards,
235         ! to the formatted memory array
240         OUTPUT @Hp8711;"TRACE:DATA CH1FMEM";    ! Note the ";"
250         FOR Pt=1 TO 201
260             OUTPUT @Hp8711;",";Fmt(202-Pt);     ! Note the ";"
270         NEXT Pt
280         OUTPUT @Hp8711;""      ! Send linefeed
290    END LOOP
```

This example program uses ASCII trace data transfers. Higher speed can be achieved using binary data transfers. If using IBASIC, high-speed subroutines can be used for even greater performance. Refer to the IBASIC Handbook for details.

Downloading Trace Data Using Binary Encoding

Data traces can be downloaded to the analyzer using binary encoding. Using binary encoding is faster than using ASCII encoding. As mentioned in "Using Binary Data Encoding" on page 6-6, the binary encoded trace is transferred as a block; the block contains a header and a data section. There are two different types of blocks that can be used: a definite length block, and an indefinite length block.

To send trace data using a definite length block, your program must calculate the number of bytes in the data segment of the block, and create a block header which tells the analyzer how many bytes are in the data segment.

For example, if you are sending a trace with 201 data points and using 64-bit floating point numbers for each data point (`FORM:DATA REAL, 64`), the block's data segment will contain 1608 bytes (201 points * 8 bytes/point). The header characters for a 1608 byte block are: "#41608". The first digit after the "#", "4" tells how many following digits are used to specify the size. In this case, 4 digits follow, and those digits are "1608", meaning that the block contains 1608 bytes.

When you send a definite length block to the analyzer, the analyzer will read the data segment bytes, stopping when it receives the number specified in the block header.

To send trace data using an indefinite length block, your program sends a block header of "#0", followed by the data segment. After sending the data segment, your program must terminate the data block by sending an EOI. The analyzer will read the data segment bytes, stopping when it receives an EOI. To send an EOI using BASIC, you can use the statement:

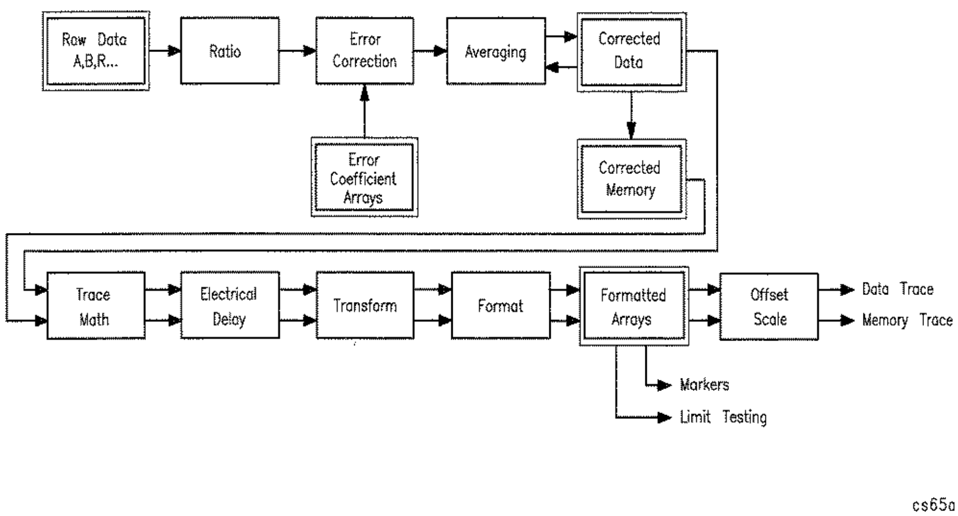
```
OUTPUT @Hp8711;END
```

Internal Measurement Arrays

The following sections describe the sequence of math operations and the resulting data arrays as the measurement information flows from the raw data arrays to the display. This information explains the measurement arrays accessible via GPIB.

Figure 6-3 is a data processing flow diagram that represents the flow of numerical data. The data passes through several math operations, denoted in the figure by single-line boxes. Most of these operations can be selected and controlled with the front panel CONFIGURE block menus. The data is stored in arrays along the way, denoted by double-line boxes. These arrays are places in the flow path where data is accessible via GPIB. While only a single flow path is shown, two identical paths are available, corresponding to measurement channels 1 and 2.

Figure 6-3 **Numeric Data Flow Through the Network Analyzer**



Raw Data Arrays

These arrays are linear measurements of the inputs used in the selected measurement. Note that these are pairs of complex numbers. The arrays are directly accessible via GPIB and are referenced as `CH[1|2]AFWD`, CH[1|2]BFWD and CH[1|2]RFWD. Raw data for AUX INPUT is not available via GPIB. Use the corrected data array to access AUX INPUT data.

Table 6-4 **Raw Data Arrays**

<table>
<tr><th>Selected Measurement</th><th>Raw Arrays</th></tr>
<tr><td>Transmission (B/R)</td><td>B = CH[1|2]BFWD, R=CH[1|2]RFWD</td></tr>
<tr><td>Reflection (A/R)</td><td>A = CH[1|2]AFWD, R= CH[1|2]RFWD</td></tr>
<tr><td>A</td><td>A = CH[1|2]AFWD</td></tr>
<tr><td>B</td><td>B = CH[1|2]BFWD</td></tr>
<tr><td>R</td><td>R = CH[1|2]RFWD</td></tr>
<tr><td>Power (B*)</td><td>B*= CH[1|2]BFWD</td></tr>
<tr><td>Conversion Loss (B*/R*)</td><td>B*= CH[1|2]BFWD, R*= CH[1|2]RFWD</td></tr>
<tr><td>R*</td><td>R*= CH[1|2]RFWD</td></tr>
<tr><td>AM Delay (Y/X)</td><td>Y = CH[1|2]BFWD, X = CH[1|2]RFWD</td></tr>
<tr><td>X</td><td>X = CH[1|2]RFWD</td></tr>
<tr><td>Y</td><td>Y = CH[1|2]BFWD</td></tr>
<tr><td>Y/R*</td><td>Y = CH[1|2]BFWD, R* = CH[1|2]RFWD</td></tr>
<tr><td>Y/X, X/Y</td><td>Y = CH[1|2]BFWD, X = CH[1|2]RFWD</td></tr>
</table>

Ratio Calculations

These are performed if the selected measurement is a ratio (e.g. A/R or B/R). This is simply a complex divide operation. If the selected measurement is absolute (e.g. A or B), no operation is performed.

Error Correction

Error correction is performed next if correction is turned on. Error correction removes repeatable systematic errors (stored in the error coefficient arrays) from the raw arrays. The operations performed depend on the selected measurement type.

Error Coefficient Arrays

The error coefficient arrays are either default values or are created during a measurement calibration. These are used whenever correction is on. They contain complex number pairs, are accessible via GPIB, and are referenced as `CH[1|2]SCORR1`, `CH[1|2]SCORR2`, `CH[1|2]SCORR3` and `CH[1|2]SCORR4`.

Table 6-5 Error Coefficient Arrays

Selected Measurement	Error Coefficient Arrays
Transmission (B/R) Response	CH[1|2]SCORR1 Tracking
Transmission (B/R) Response & Isolation	CH[1|2]SCORR1 Tracking CH[1|2]SCORR2 Isolation Term
Transmission (B/R) Enhanced Response	CH[1|2]SCORR1 Directivity CH[1|2]SCORR2 Source Match CH[1|2]SCORR3 Reflection Tracking CH[1|2]SCORR4 Transmission Tracking
Reflection (A/R)	CH[1|2]SCORR1 Directivity CH[1|2]SCORR2 Source Match CH[1|2]SCORR3 Tracking
Broadband Internal	CH[1|2]SCORR1 R* Response

NOTE These arrays do not apply to Broadband External measurements.

Table 6-6 2-Port Error Coefficient Arrays

<table>
<tr><th>Direction</th><th>Error Coefficient Arrays</th></tr>
<tr><td rowspan="6">Forward</td><td>CH[1|2]SCORR1 Directivity</td></tr>
<tr><td>CH[1|2]SCORR2 Source match</td></tr>
<tr><td>CH[1|2]SCORR3 Reflection tracking</td></tr>
<tr><td>CH[1|2]SCORR4 Transmission tracking</td></tr>
<tr><td>CH[1|2]SCORR5 Load match</td></tr>
<tr><td>CH[1|2]SCORR6 Isolation</td></tr>
<tr><td rowspan="6">Reverse</td><td>CH[1|2]SCORR7 Directivity</td></tr>
<tr><td>CH[1|2]SCORR8 Source match</td></tr>
<tr><td>CH[1|2]SCORR9 Reflection tracking</td></tr>
<tr><td>CH[1|2]SCORR10 Transmission tracking</td></tr>
<tr><td>CH[1|2]SCORR11 Load match</td></tr>
<tr><td>CH[1|2]SCORR12 Isolation</td></tr>
</table>

Averaging

Averaging is a noise reduction technique. This calculation involves taking the complex exponential average of several consecutive sweeps. This averaging calculation is different than the System Bandwidth setting. System Bandwidth uses digital filtering, applying noise reduction to the measured data before it is stored into the Raw Data Arrays.

Corrected Data Arrays

The combined results of the ratio, error correction and averaging operations are stored in the corrected data arrays as complex number pairs. These arrays are accessible via GPIB and referenced as `CH[1|2]SDATA`.

Corrected Memory Arrays

If the Data–>Mem or Normalize operations are performed, the corrected data arrays are copied into the corrected memory arrays. These arrays are accessible via GPIB and referenced as `CH[1|2]SMEM`.

Trace Math Operation

This selects either the corrected data array, or the corrected memory array, or both to continue flowing through the data processing path. In addition, the complex ratio of the two (Data/Memory) can also be selected. If memory is displayed, the data from the memory arrays goes through exactly the same data processing flow path as the data from the data arrays.

Electrical Delay

This block adds or subtracts phase, based on the settings of Phase Offset, Electrical Delay, and Port Extension. The Electrical Delay and Port Extension features add or subtract phase in proportion to frequency. This is equivalent to "line stretching" or artificially moving the measurement reference plane. (See your analyzer's *User Guide* for more details on these features.)

Transform (Option 100 only)

This block converts frequency domain data into distance domain, or into an SRL impedance value when measuring fault location or SRL. The transform employs an inverse fast Fourier transform (FFT) algorithm to accomplish the conversion.

Formatting

This converts the complex number pairs into a scalar representation for display, according to the selected format (e.g. Log Mag, SWR, etc). These formats are often easier to interpret than the complex number representation. Note that after formatting, it is impossible to recover the complex data.

Formatted Arrays

The results so far are stored in the formatted data and formatted memory arrays. It is important to note that marker values and marker functions are all derived from the formatted arrays. Limit testing is also performed on the formatted arrays. These arrays are accessible via GPIB and referenced as `CH[1|2]FDATA` and `CH[1|2]FMEM`.

Offset and Scale

These operations prepare the formatted arrays for display. This is where the reference position, reference value, and scale calculations are performed, as appropriate for the format.

7 Using Graphics

Introduction

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

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

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

`WINDow1`	draws the graphics to the channel 1 measurement screen. (This is the default if no window is specified in the mnemonic.)
`WINDow2`	draws the graphics to the channel 2 measurement screen.
`WINDow10`	draws the graphics to an IBASIC display partition. The graphics can only be output in red on a PCL printer.

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

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

```
DISPlay:WINDow[1|2|10]:GRAPhics:CIRCle <y_radius>
```

```
DISPlay:WINDow[1|2|10]:GRAPhics:CLEar
```

```
DISPlay:WINDow[1|2|10]:GRAPhics:COLor <pen>
```

- color choices are: 0 for erase, 1 for bright, 2 for dim

```
DISPlay:WINDow[1|2|10]:GRAPhics[:DRAW] <new_x>,<new_y>
```

```
DISPlay:WINDow[1|2|10]:GRAPhics:LABel <string>
```

```
DISPlay:WINDow[1|2|10]:GRAPhics:LABel:FONT <font>
```

- font choices are: SMALl, HSMall, NORMal, HNORmal, BOLD, HBOLd, SLANt, HSLant (H as the first letter of the font name indicates highlighted text – inverse video).

```
DISPlay:WINDow[1|2|10]:GRAPhics:MOVE <new_x>,<new_y>
```

```
DISPlay:WINDow[1|2|10]:GRAPhics:RECTangle
<width>,<height>
```

```
DISPlay:WINDow[1|2|10]:GRAPhics:SCALe
<xmin>,<xmax>,<ymin>,<ymax>
```

```
DISPlay:WINDow[1|2|10]:GRAPhics:STATe?
```

Window Geometry

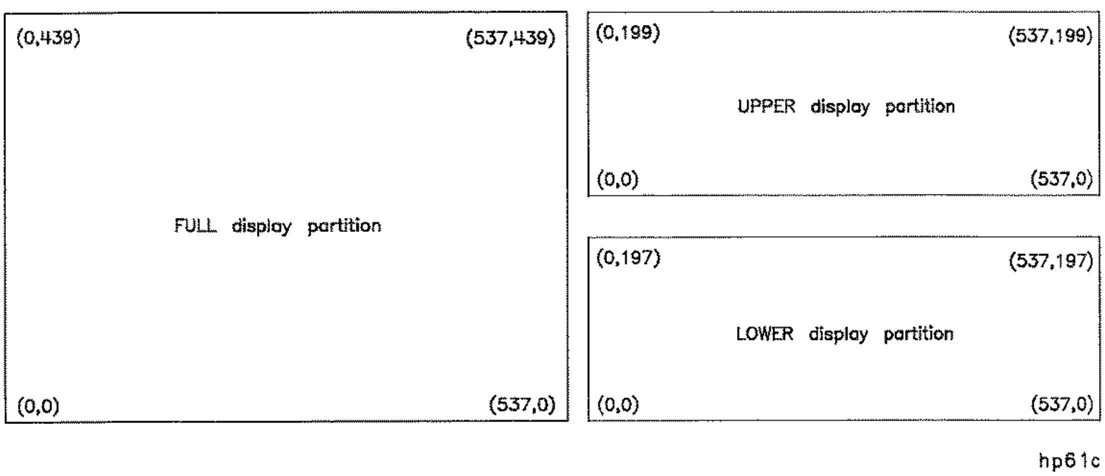
Even though there are only three graphics windows, these windows can have different sizes and locations.

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

The sizes of the different graphics windows are listed below. Figure 7-1 shows the display partitions.

- Measurement channel 1 or 2 full screen measurement:
- Measurement channel 1 or 2 split screen measurement:
- IBASIC full screen display:
- IBASIC upper display:
- IBASIC lower display:

Figure 7-1 **Pixel Dimensions with Available Display Partitions**



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

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

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

NOTE The origin of *every* graphics window is its lower left corner. The locations returned in response to the `LLEFt` and `URIGht` are relative to the *absolute* origin of the entire display, *not* to the graphics window.

The Graphics Buffer

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

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

The graphics buffer will hold up to:

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

Use the following command to clear the graphics buffer and user-graphics display.

```
DISPlay:WINDow:GRAPhics:CLEar
```

NOTE Only graphics that can be refreshed will be printed or plotted. If you intend to print or plot your graphics, make sure they will fit within the graphics buffer.

8 Front Panel Keycodes

Front Panel Keycodes

Your program can control or monitor the analyzer's front panel with the use of the SCPI `SYSTem:KEY` commands.

Controlling the Front Panel

The front panel can be controlled by sending commands to execute the function of specific keys. The SCPI command `SYSTem:KEY <char>` sends a key name to the analyzer which executes the same function as the corresponding front panel key. For example, `SYSTem:KEY FREQ` will execute the function of the FREQ hardkey.

Every hardkey and softkey has a unique key name. Refer to the last table in this chapter for a list of all key names.

Monitoring the Front Panel

The front panel can be monitored to determine when a key has been pressed or when the knob (RPG — rotary pulse generator) has been turned. Key presses from an attached PC keyboard can also be captured.

When keys are pressed or when the knob is turned, the analyzer detects this event, sets bit 0 of the Device Status Register (see Chapter 5, "Using Status Registers"), and stores the associated information in a key queue. Your program can use the SCPI `SYSTem:KEY` commands to read the contents of the key queue.

The SCPI query `SYSTem:KEY:TYPE?` returns a string indicating the type of key press event:

Table 8-1 **Key Press Return Values**

Return Value	Meaning
NONE	No key has been pressed.
KEY	A front panel key has been pressed.
RPG	The analyzer's knob has been turned.
ASC	A key on the ASCII PC DIN keyboard has been pressed.

The SCPI query `SYSTem:KEY[:VALue]?` returns a number describing the type of key press. The meaning of the number depends on the key type returned by the `SYSTem:KEY:TYPE?` query:

Table 8-2 **Key Press Types**

SYST:KEY:TYPE	SYST:KEY:VALUE Meaning
NONE	No meaning. Returns –1.
KEY	A number from 0 to 56 representing the "key code" of the front panel key. See following table for list.
RPG	The number of knob "ticks." Positive values indicate a clock-wise turn; negative numbers indicate counter-clockwise. Larger numbers indicate the knob has been turned faster or further.
ASC	The ASCII value of the pressed key.

Key Queue

The `SYSTem:KEY[:VALue]?` query removes the key from the key queue, so that you can read the next key. For this reason, you must perform the `SYSTem:KEY:TYPE?` query before performing the `SYSTem:KEY[:VALue]?`.

The Key Queue stores up to 32 key press events. After 32 key presses, the queue is full, and no more key press events can be stored without reading from the queue (using `SYSTem:KEY[:VALue]?`). Subsequent key presses or knob ticks will be ignored when the queue is full.

You can query the queue length using this command:

```
SYSTem:KEY:QUEue:MAXimum?
```

You can clear the queue using this command:

```
SYSTem:KEY:QUEue:CLEar
```

You can check how many key presses or knob tick events have occurred using this command:

```
SYSTem:KEY:QUEUE:COUNt?
```

You can turn the key queue on or off using this command:

```
SYSTem:KEY:QUEUE[:STATe] <ON|OFF>
```

When the queue is turned off, your program must read each key before a following key is pressed, or information will be lost. It is generally best to leave the queue enabled.

Example Program

For a complete example of how to read the front panel keys and knob, refer to the KEYCODE example program in the *Example Programs Guide*.

Table 8-3 **Key Codes**

Key Label	Key Code	GPIB Key Name
Softkey 1	0	SOFTkey1
Softkey 2	1	SOFTkey2
Softkey 3	2	SOFTkey3
Softkey 4	3	SOFTkey4
Softkey 5	4	SOFTkey5
Softkey 6	5	SOFTkey6
Softkey 7	6	SOFTkey7
Softkey 8	7	SOFTkey8
0	10	ZERO
1	11	ONE
2	12	TWO
3	13	THRee
4	14	FOUR
5	15	FIVE
6	16	SIX
7	17	SEVen
8	18	EIGHt
9	19	NINE
ENTER	20	ENTer

Key Label	Key Code	GPIB Key Name
.	21	POINt
–	22	MINus
⇧	23	UP
⇩	24	DOWN
BEGIN	40	BEGin
MEAS 1	41	MEAS1
MEAS 2	42	MEAS2
POWER	43	POWer
MENU	44	MENU
FREQ	45	FREQ
SWEEP	46	SWEep
CAL	47	CAL
DISPLAY	48	DISPLAY
SCALE	49	SCALe
AVG	50	AVG
FORMAT	51	FORMat
MARKER	52	MARKer
SAVE RECALL	53	SAVE
SYSTEM OPTIONS	54	SYSTem
HARD COPY	55	HARDcopy
PRESET	56	PRESet

9 Introduction to SCPI

Introduction to SCPI

This chapter is a guide to GPIB control of the analyzer. Its purpose is to provide concise information about the operation of the analyzer under GPIB control. The reader should already be familiar with making measurements with the analyzer and with the general operation of GPIB.

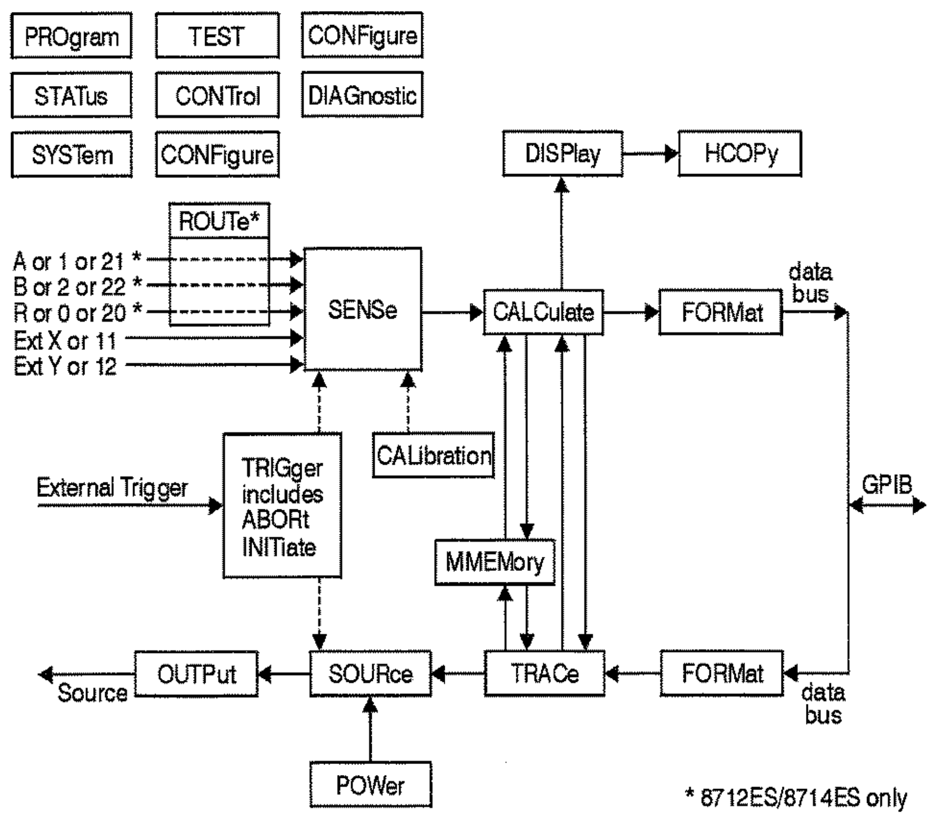
Standard Commands for Programmable Instruments (SCPI) is a programming language designed specifically for controlling instruments by Agilent Technologies and other industry leaders. SCPI provides commands that are common from one instrument to another. This elimination of "device specific" commands for common functions allows programs to be used on different instruments with very little modification.

SCPI was developed to conform to the IEEE 488.2 standard (replacing IEEE 728-1982). The IEEE 488.2 standard defines the syntax and data formats used to send data between devices, the structure of status registers, and the commands used for common tasks. For more information, refer to the IEEE standard itself. SCPI defines the commands used to control device-specific functions, the parameters accepted by these functions, and the values they return.

The Command Tree

The SCPI standard organizes related instrument functions by grouping them together on a common branch of a command tree (see Figure 9-2 on page 9-6 for an example command tree). Each branch is assigned a mnemonic to indicate the nature of the related functions. The analyzer has 16 major SCPI branches or **subsystems**. See Figure 9-1 for a model of how these subsystems are organized to manage the measurement and data flow for the analyzer.

Figure 9-1 **Measurement and Data Flow of the Analyzer**



cp61es

The analyzer's major SCPI subsystems and their functions are described below.

`ABORt`	Aborts any sweep in progress.
`CALCulate`	Configures post-measurement processing of the measured data (such as marker and limit testing functions).
`CALibration`	Controls zeroing the broadband diode detectors.
`CONFigure`	Configures the analyzer to measure a specific device type, including amplifiers, mixers, filters, and cables.
`CONTrol`	Configures the analyzer for use with a multiport test set. Used with the 87075C multiport test set only.
`DIAGnostic`	Performs a number of diagnostic and I/O functions. Includes LAN diagnostics, port reads and writes, correction constants utilities, and other miscellaneous functions.
`DISPlay`	Controls the display of measurement data, annotation and user graphics.
`FORMat`	Controls the format of data transfers over the GPIB. For more information about GPIB data transfers, refer to Chapter 4, "Data Types and Encoding."
`HCOPy`	Controls hardcopy (printer and plotter) output.
`INITiate`	Controls the triggering of sweeps.
`MMEMory`	Controls mass storage of instrument states and data (disk and internal memory interface functions).
`OUTPut`	Turns on/off the source output power (power to the device under test).
`POWer`	Specifies frequency sweeps or power sweeps.
`PROGram`	Interfaces IBASIC programs and commands with an external controller. For more information on IBASIC programming, refer to *HP Instrument BASIC User's Handbook*.
`ROUTe`	Selects transmission and reflection ports. Selects ports on multiport test sets (used with the 87075C multiport test set only).

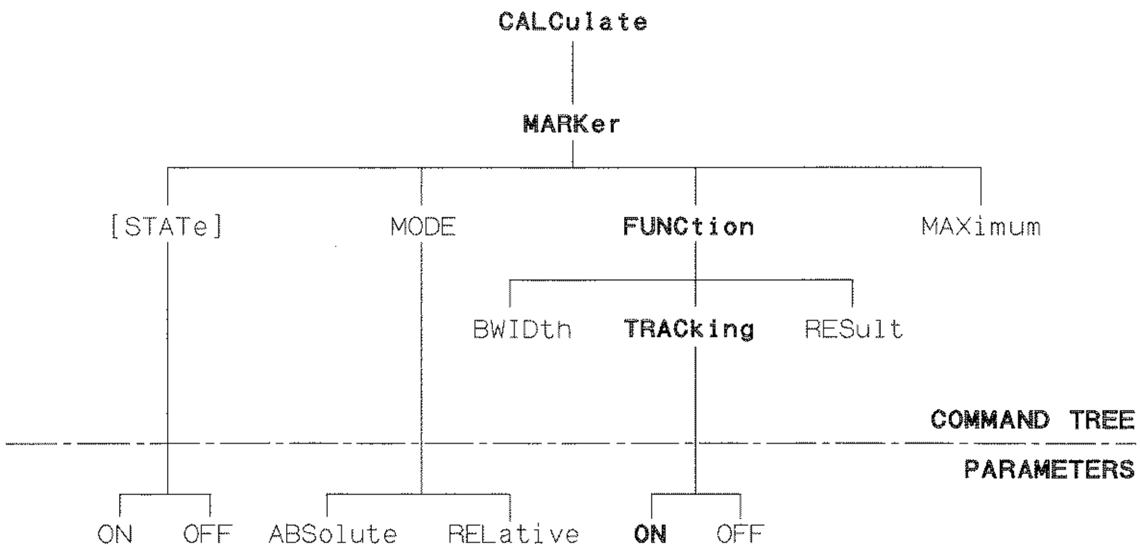
SENSe	Configures parameters (such as the frequency and measurement parameters) related to the sweep and the measured signal (from the device under test). This subsystem also controls the narrowband calibration routines.
SOURce	Controls the RF output power level of the source (power to the device under test).
STATus	Contains the commands for using the SCPI status registers. (For more information about using the status registers, refer to Chapter 5, "Using Status Registers.")
SYSTem	Contains miscellaneous system configuration commands (such as I/O port, clock and softkey control).
TEST	Performs instrument self-test functions.
TRACe	Interfaces with the internal data arrays (functions such as data transfer and trace memory).
TRIGger	Controls the source of the sweep triggering.

When many functions are grouped together on a particular branch, additional branching is used to organize these functions into groups that are even more closely related. The branching process continues until each analyzer function is assigned to its own branch. For example, the function that turns on and off the marker tracking feature is assigned to the `TRACKING` branch of the `FUNCTION` branch of the `MARKER` branch of the `CALCULATE` subsystem. The command looks like this:

```
CALCULATE:MARKER:FUNCTION:TRACKING ON
```

NOTE Colons are used to indicate branching points on the command tree. A parameter is separated from the rest of the command by a space.

Figure 9-2 **Partial Diagram for the CALCulate Subsystem Command Tree**



Sending Multiple Commands

Multiple commands can be sent within a single program message by separating the commands with semicolons. For example, the following program message — sent within an HP BASIC `OUTPUT` statement — turns on the marker reference and moves the main marker to the highest peak on the trace:

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

One of the analyzer's command parser main functions is to keep track of a program message's position in the command tree. This allows the previous program message to be simplified. Taking advantage of this parser function, the simpler equivalent program message is:

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

In the first version of the program message, the semicolon that separates the two commands is followed by a colon. Whenever this occurs, the command parser is reset to the base of the command tree. As a result, the next command is only valid if it includes the entire mnemonic path from the base of the tree.

In the second version of the program message, the semicolon that separates the two commands is not followed by a colon. Whenever this occurs, the command parser assumes that the mnemonics of the second command arise from the same branch of the tree as the final mnemonic of the preceding command. `MODE`, the final mnemonic of the first command, arises from the `MARKER` branch. So `MAXIMUM`, the first mnemonic of the second command, is also assumed to arise from the `MARKER` branch.

The following is a longer series of commands — again sent within HP BASIC `OUTPUT` statements — that can be combined into a single program message:

```
OUTPUT 716;"CALCULATE:MARKER:STATE ON"
OUTPUT 716;"CALCULATE:MARKER:MODE RELATIVE"
OUTPUT 716;"CALCULATE:MARKER:MAXIMUM"
OUTPUT 716;"CALCULATE:MARKER:FUNCTION:TRACKING ON"
```

The single program message is:

```
OUTPUT 716;"CALCULATE:MARKER:STATE ON;MODE
            RELATIVE;MAXIMUM;FUNCTION:TRACKING ON"
```

Command Abbreviation

Each command mnemonic has a long form and a short form. The short forms of the mnemonics allow you to send abbreviated commands. Only the exact short form or the exact long form is accepted.

The short form mnemonics are created according to the following rules:

- If the long form mnemonic has four characters or less, the short form is the same as the long form. For example, `DATA` remains `DATA`.
- If the long form mnemonic has more than four characters and the fourth character is a consonant, the short form consists of the first four characters of the long form. For example, `CALCULATE` becomes `CALC`.
- If the long form mnemonic has more than four characters and the fourth character is a vowel, the short form consists of the first three characters of the long form. For example, `LIMIT` becomes `LIM`.

NOTE

The short form of a particular mnemonic is indicated by the use of UPPER-CASE characters in this manual.

SCPI is not case sensitive so any mix of upper and lower-case lettering can be used when sending commands to the analyzer.

If the rules listed in this section are applied to the last program message in the preceding section, the statement:

```
OUTPUT 716;"CALCULATE:MARKER:STATE ON;MODE
          RELATIVE;MAXIMUM;FUNCTION:TRACKING ON"
```

becomes:

```
OUTPUT 716;"CALC:MARK:STAT ON;MODE REL;MAX;FUNC:TRAC ON"
```

Implied Mnemonics

Some mnemonics can be omitted from GPIB commands without changing the effect of the command. These special mnemonics are called implied mnemonics, and they are used in many subsystems. In addition to entire mnemonics, variable parts of some mnemonics may also be implied. These are usually a number indicating a particular measurement channel, marker, or similar choice.

NOTE

When a number is not supplied for an implied variable, a default choice is assumed; this choice is always `1`.

The `INITIATE` subsystem contains both the implied mnemonic `IMMEDIATE` at its first branching point and an implied variable for the measurement channel. The command to trigger a new sweep is shown in the "SCPI Command Summary" as:

```
OUTPUT 716;"INITiate[1|2][:IMMediate]
```

Any of the following forms of the command can be sent to the analyzer (using HP BASIC) to trigger a new sweep on measurement channel 1:

```
OUTPUT 716;"INITIATE1:IMMEDIATE"
OUTPUT 716;"INITIATE:IMMEDIATE"
OUTPUT 716;"INITIATE1"
OUTPUT 716;"INITIATE"
```

If the sweep is to be triggered for measurement channel 2, the channel number *must* be specified:

```
OUTPUT 716;"INITIATE2:IMMEDIATE"
OUTPUT 716;"INITIATE2"
```

Parameter Types

Parameters are used in many commands. The analyzer uses several types of parameters with different types of commands and queries. When a parameter is sent with a SCPI command, it must be separated from the command by a space. If more than one parameter is sent, they are separated from each other by commas.

Numeric Parameters

Most subsystems use numeric parameters to specify physical quantities. Simple numeric parameters accept all commonly used decimal representations of numbers, including optional signs, decimal points, and scientific notation. If an instrument setting programmed with a numeric parameter can only assume a finite number of values, the instrument automatically rounds the parameter. In addition to numeric values, all numeric parameters accept `MAXimum` and `MINimum` as values (note that `MAXimum` and `MINimum` can be used to set or query values).

`<num>` is used in this document to denote a numeric parameter.

An example is the command to set the stop frequency for a measurement. The first command below sets the stop frequency to a specific value. The second command below sets the stop frequency to its maximum possible value (1300 MHz for 8712ET/ES or 3000 MHz for 8714ET/ES).

```
OUTPUT 716;"SENSE1:FREQUENCY:STOP 1300 MHZ"

OUTPUT 716;"SENSE1:FREQUENCY:STOP MAX"
```

Query Response

When a numeric parameter is queried, the number is returned in one of the three numeric formats.

NR1	Integers (such as +1, 0, -1, 123, -12345)
NR2	Floating point number with an explicit decimal point (such as 12.3, +1.234, -0.12345)
NR3	Floating point number in scientific notation (such as +1.23E+5, +123.4E-3, -456.789E+6)

An example is the response to a query of the stop frequency after executing the above commands (this response is of the NR3 type).

```
OUTPUT 716;"SENSE1:FREQUENCY:STOP?"
```

returns the value `1.3E+9`.

Character Parameters

Character parameters (sometimes referred to as discrete parameters) consist of ASCII characters. They are typically used for program settings that have a finite number of values.

These parameters use mnemonics to represent each valid setting. They have a long and a short form which follow the same rules as command mnemonics.

`<char>` is used in this document to denote a character parameter.

An example of a command using a character parameter is the command that selects the format in which the measurement data is displayed:

```
OUTPUT 716;"CALCULATE1:FORMAT MLOGARITHMIC"
```

Query Response

When a character parameter is queried the response is always the short form of the mnemonic that represents the current setting. An example is the response to a query of the data format after executing the above command.

```
OUTPUT 716;"CALCULATE1:FORMAT?"
```

returns the value `MLOG`.

Boolean Parameters

Boolean parameters are used for program settings that can be represented by a single binary condition. Commands that use this type of parameter accept the values ON (or 1) and OFF (or 0).

<ON | OFF> is used in this document to denote a boolean parameter.

An example of a command that uses a boolean parameter is the command that makes the analyzer continuously trigger (or stop triggering) measurements.

```
OUTPUT 716;"INITIATE:CONTINUOUS ON"
```

A special group of commands uses boolean parameters to control automatic functions of the instrument, such as automatically selecting the fastest possible sweep speed. With these automatic functions an additional value is available for the parameter. This value ONCE causes the function to execute once before turning off.

Query Response

The response when a boolean parameter is queried is a single NR1 number indicating the state 1 for on or 0 for off. An example is the response to a query on the sweep trigger status after executing the above command.

```
OUTPUT 716;"INITIATE:CONTINUOUS?"
```

returns the value 1.

String Parameters

String parameters can contain virtually any set of ASCII characters. The string must begin with a single quote (') or a double quote (") and end with the same character (called the delimiter). The delimiter can be included as a character (embedded) inside the string by typing it twice without any characters in between. For example:

```
OUTPUT 716;"DISP:ANN:TITL:DATA 'DUT''S PHASE'"
```

`<string>` is used in this document to denote a string parameter.

An example of a command that uses a string parameter is the `CONFIGURE` command:

```
OUTPUT 716;"CONFIGURE 'FILTER:TRANSMISSION'"
```

Some of the string parameters used by the analyzer, like `'FILTER:TRANSMISSION'` in the example above, follow the same rules that apply to mnemonics. They may have branching (`'FILTER:REFLECTION'` is a related command) and abbreviated versions.

Query Response

The response when a string parameter is queried is a string. The only difference is that the response string will only use double quotes as delimiters. Embedded double quotes may be present in string response data. When the string follows the "SCPI" mnemonic rules, the string returned in response to a query is in the abbreviated form. An example is the response to the configuration status of the analyzer (after executing the last command).

```
OUTPUT 716;"CONFIGURE?"
```

returns the value `"FILT:TRAN"`.

Block Parameters

Block parameters are typically used to transfer large quantities of related data (like a data trace). Blocks can be sent as definite length blocks or indefinite length blocks — the instrument will accept either form. For more information on block data transfers refer to Chapter 4, “Data Types and Encoding.” `<block>` is used in this document to denote a block parameter.

Syntax Summary

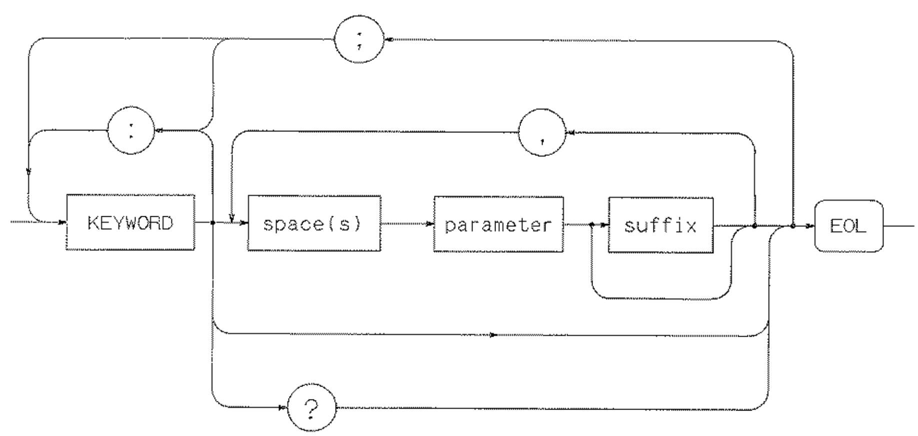
The following conventions are used throughout this manual whenever SCPI mnemonics are being described.

angle brackets (< >) are used to enclose required parameters within a command or query. The definition of the variable is usually explained in the accompanying text.

square brackets ([ ]) are used to enclose implied or optional parameters within a command or query.

UPPERlower case are used to indicate the short form (upper-case) of a given mnemonic. The remaining (lower-case) letters are the rest of the long form mnemonic.

Figure 9-3 **SCPI Command Syntax**



The following elements have special meanings within a SCPI program message (or combination or mnemonics).

colon (:) — When a command or query contains a series of mnemonics, they are separated by colons. A colon immediately following a mnemonic tells the command parser that the program message is proceeding to the next level of the command tree. A colon immediately following a semicolon tells the command parser that the program message is returning to the base of the command tree.

semicolon (;) — When a program message contains more than one command or query, a semicolon is used to separate them from each other.

comma (,) — A comma separates the data sent with a command or returned with a response.

space ( ) — One space is required to separate a command or query from its data (or parameters). Spaces are not allowed inside a command or query.

IEEE 488.2 Common Commands

IEEE 488.2 defines a set of common commands. All instruments are required to implement a subset of these commands, specifically those commands related to status reporting, synchronization and internal operations. The rest of the common commands are optional. The following list details which of these IEEE 488.2 common commands are implemented in the analyzer and the response of the analyzer when the command is received.

`*CLS` — Clears the instrument Status Byte by emptying the error queue and clearing all event registers, also cancels any preceding `*OPC` command or query (does not change the enable registers or transition filters).

`*ESE <num>` — Sets bits in the Standard Event Status Enable Register — current setting is saved in non-volatile memory.

`*ESE?` — Reads the current state of the Standard Event Status Enable Register.

`*ESR?` — Reads and clears the current state of the Standard Event Status Register.

`*IDN?` — Returns a string that uniquely identifies the analyzer. The string is of the form

```
"HEWLETT-PACKARD,8712,<serial number>,<software
evision>"
```

`*LRN?` — This returns a string of device specific characters that, when sent back to the analyzer will restore the instrument state active when `*LRN?` was sent. Data formatting (`ENTER USING "-K"` in HP BASIC) or a similar technique should be used to ensure that the transfer does not terminate on a carriage return or line feed (both $^{C}{}_{R}$ and $^{L}{}_{F}$ are present in the learn string as part of the data).

`*OPC`	Operation complete command. The analyzer will generate the OPC message in the Standard Event Status Register when all pending overlapped operations have been completed (e.g. a sweep, or a preset). For more information about overlapped operations, refer to "Overlapped Commands" on page 2-3.
`*OPC?`	Operation complete query. The analyzer will return an ASCII "1" when all pending overlapped operations have been completed.
`*OPT?`	Returns a string identifying the analyzer's option configuration. The string is of the form "1E1,100". The options are identified by the following:

1EC	75 ohm
1E1	60 dB step attenuator
100	SRL and Fault Location

`*PCB <num>`	Sets the pass-control-back address (the address of the controller before a pass control is executed).
`*PSC <num>`	Sets the state of the Power-on Status Clear flag — flag is saved in non-volatile memory. This flag determines whether or not the Service Request enable register and the Event Status enable register are cleared at powerup.

`*RST`	Executes a device reset and cancels any pending `*OPC` command or query. The contents of the instrument's nonvolatile memory are *not* affected by this command. This command is different from the front panel (PRESET) function in the state of the commands (and their reset states) listed below. The preset instrument state is described in the *User's Guide*.

`INITiate:CONTinuous`	= OFF
`OUTPut[:STATe]`	= OFF
`CALibration:ZERO:AUTO`	= OFF
`SENSe:CORRection[:STATe]`	= OFF
`SENSe:SWEep:POINts`	= MAX
`SOURce:POWer`	= MIN

`*SRE <num>`	Sets bits in the Service Request Enable Register. Current setting is saved in non-volatile memory.
`*SRE?`	Reads the current state of the Service Request Enable Register.
`*STB?`	Reads the value of the instrument Status Byte. This is a non-destructive read—the Status Byte is cleared by the `*CLS` command.
`*TRG`	Triggers a sweep on the active measurement channel when in Trigger Hold mode. Ignored if in continuous sweep.
`*TST?`	Returns the result of a complete self-test. An ASCII `0` indicates no failures found. Any other character indicates a specific self-test failure. Does not perform any self-tests. See the *Service Guide* for further information.
`*WAI`	Prohibits the instrument from initiating any new commands until all pending overlapped commands have been initiated.

10 Menu Map with SCPI Commands

This chapter shows all softkey menu choices available. Each hardkey on the instrument front panel has a corresponding table in this chapter showing all softkey choices available after pressing the hardkey.

Hardkeys and softkeys are shown as **HARDKEY** and **Soft Key**.

Each softkey is shown with an associated SCPI command, if one exists. SCPI commands are shown as the short form.

NOTE *The configuration of the currently active measurement channel determines the order of appearance and the content of the softkey menus.*

Command Conventions

Commands requiring a choice of measurement channel are *shown* with both choices: **SENS[1|2]:** (for example). The command is *entered* with the appropriate channel given: **SENS1:** (channel 1 chosen) or **SENS2:** (channel 2 chosen).

Parameter Types

`<num>` and `<string>` refer to parameter types described in the "Parameter Types" section. `<string>` parameters are enclosed in single quotes: `'the string data'`. `<value>` parameters include `<num>` and an optional `<unit>`.

Softkey Menus

Most softkey choices and associated SCPI commands are shown in the following Menu Map tables. Because the analyzer provides great flexibility in measurement configuration, and because many softkey menus and user operations are very similar to each other, not every variation of each softkey menu is shown.

SCPI commands that are associated with menu selections can be found by reading the program that is created after keystroke recording has been enabled and the desired keys have been pressed.

NOTE Refer to the Hardkey/Softkey Reference in your instrument's *User's Guide* for a description of each key.

Entering Frequency, Power, and Other Numeric Values

Entry of frequency, power, and other numeric values requires a choice of measurement unit. All entries of this type follow the same sequence: Enter the number then choose the appropriate unit, or press **Enter** for default units.

Menu Map Tables and Instrument Types

The instrument displays menu choices based on the instrument type and the active measurement channel configuration. In the tables of softkey menu choices that follow, the instrument type is shown in the table title, and the channel configuration is shown in the table body. See Figure 10-1, below.

Figure 10-1 **Example Menu Map Table**

Table 10-8 CAL **Functions, 8712ES/8714ES**

KEYSTROKES		SCPI COMMAND
CAL (Reflection)		(hardkey entry)
Default 1-Port		`SENS[1|2]:CORR:CSET DEF; *WAI`
Default 2-Port		`SENS[1|2]:CORR:CLASS DEF2; *WAI`
User 1-Port		`SENS[1|2]:CORR:CLASS DEF1; *WAI`
		`SENS[1|2]:CORR:CSET DEF; *WAI`

Softkey
Channel configuration
Hardkey
SCPI command
Applicable models

How to Enter Numbers and Characters

For many tasks in the following menus, you enter numeric values or characters. You might enter 10.0 MHz to set a marker location, or "state5" for a file name. The following pages show you how to enter values for frequency, time, power and voltage, and how to enter text characters.

How to Enter Frequency Values

You enter frequency values the same way each time. First select a numeric value ("100") and then select a frequency unit ("MHz"). *RPG knob does not allow unit entry.

Figure 10-2 **Frequency—Enter the Value**

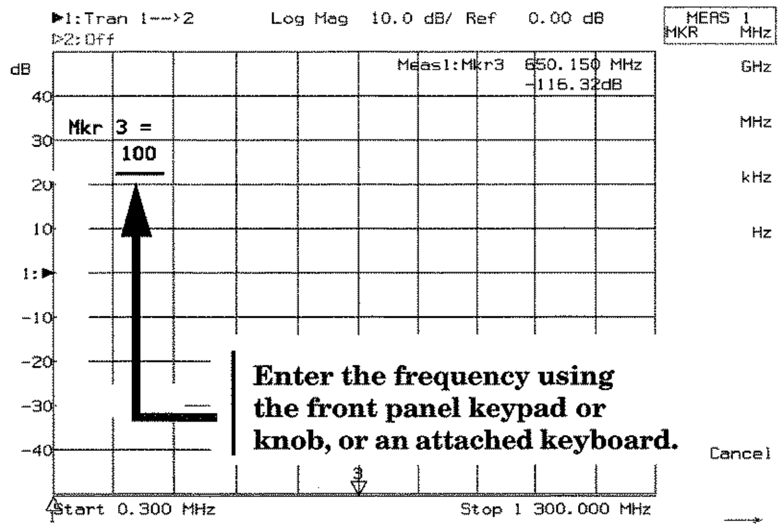
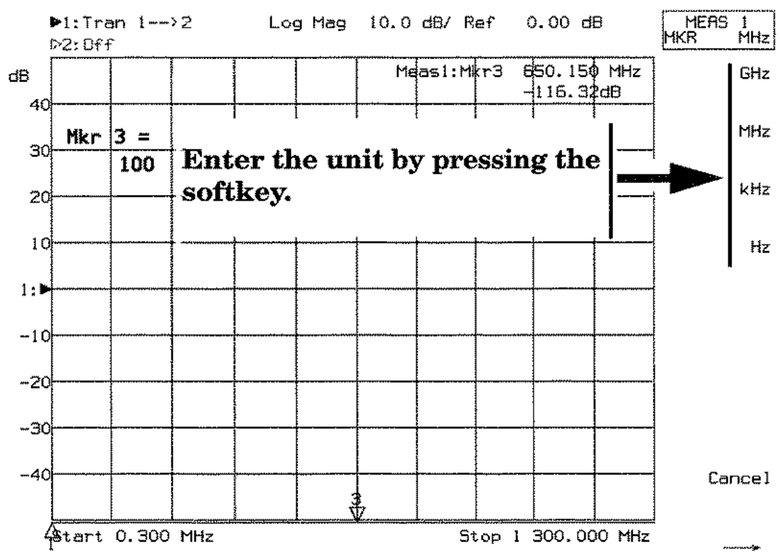


Figure 10-3 **Frequency—Enter the Unit**



How to Enter Time Values

You enter time values the same way each time. First select a numeric value (“100”) and then select a time unit (“Sec”).

Figure 10-4 Time—Enter the Value

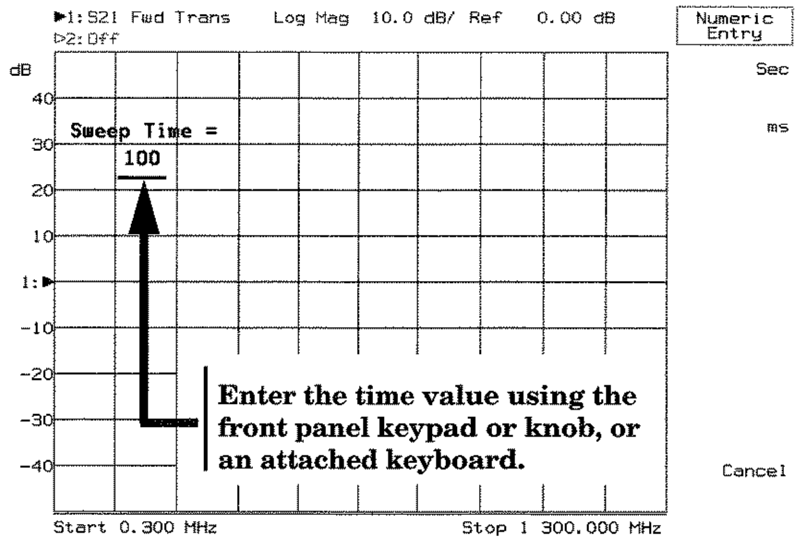
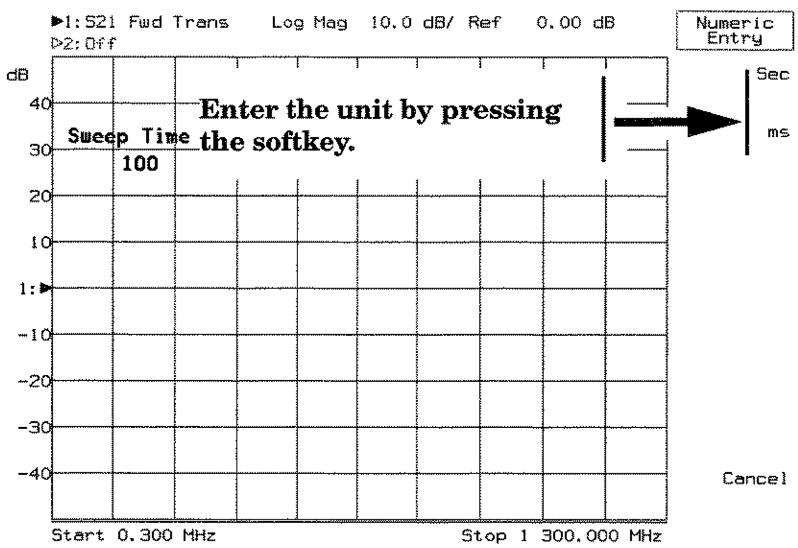


Figure 10-5 Time—Enter the Unit



How to Enter Power and Voltage Values

You enter power and voltage values the same way each time. First select a numeric value ("10") and then select a unit ("W").

Figure 10-6 **Power and Voltage—Enter the Value**

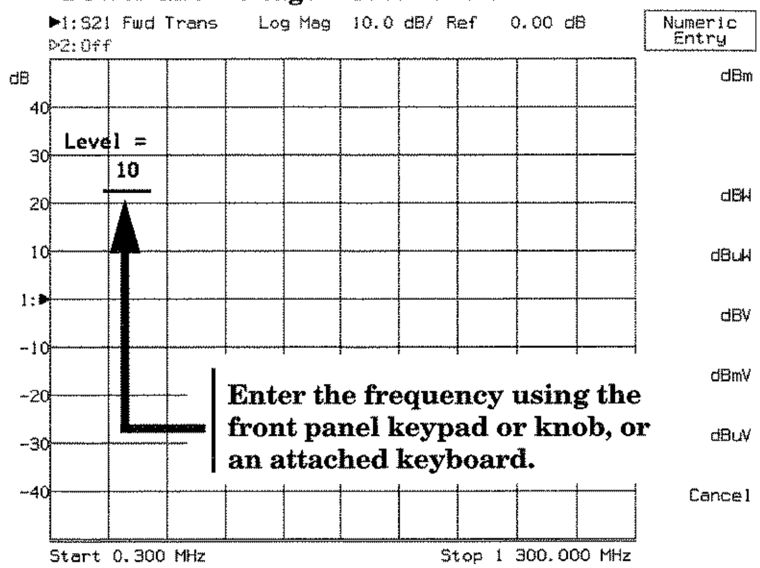
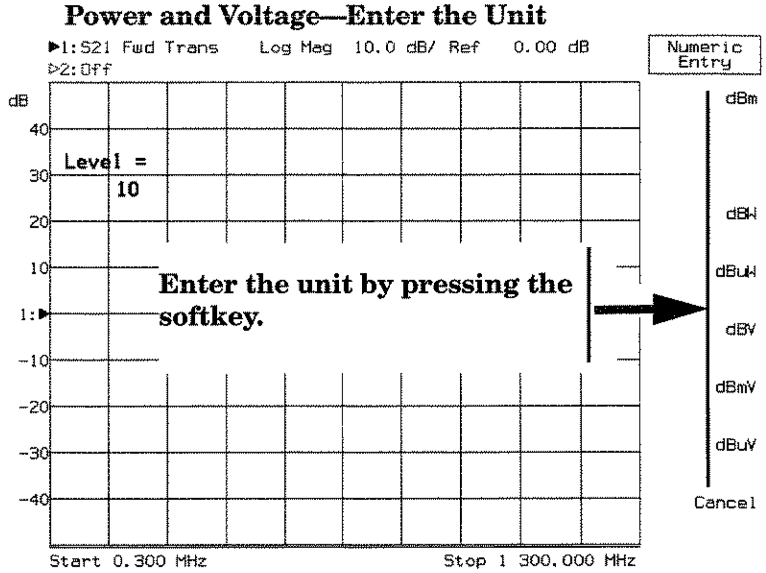


Figure 10-7 **Power and Voltage—Enter the Unit**



How to Enter Text

Choose characters with the front panel knob and press **Select Char**, or use the keyboard. The numeric key pad can be used to select numbers. Edit the text using the softkeys or keyboard.

Figure 10-8 **Text—Choose the Character**

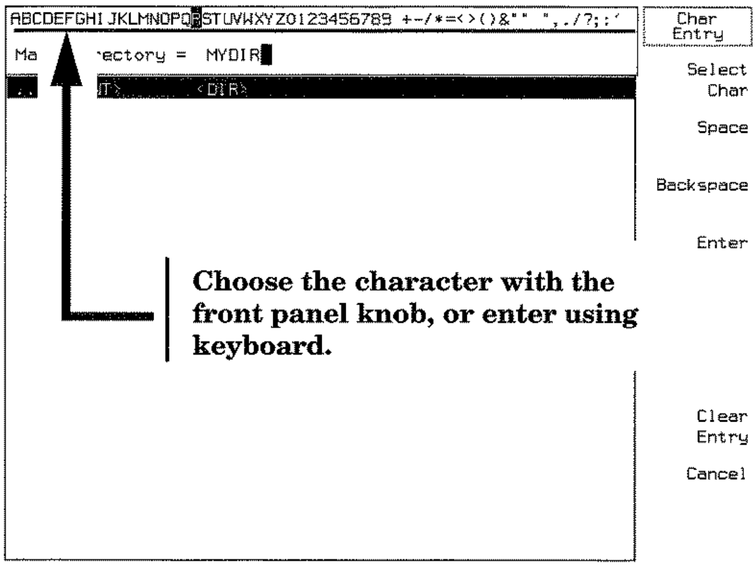
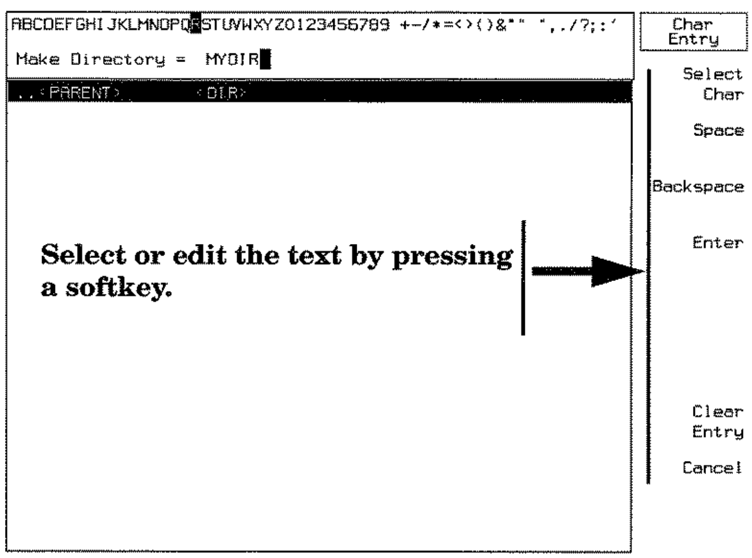


Figure 10-9 **Text—Select or Edit the Character**



Menu Map for 8712ET/ES and 8714ET/ES

Table 10-1 AVG Functions, 8712ES and 8714ES

KEYSTROKES[1]	SCPI COMMAND
AVG	(hardkey entry)
Average on OFF	`SENS[1|2]:AVER[ON|OFF];*WAI`
Restart Average	`SENS[1|2]:AVER:CLE;*WAI`
Average Factor	`SENS[1|2]:AVER:COUN <num>;*WAI`
System Bandwidth	(menu selection only)
Wide (6500 Hz)	`SENS[1|2]:BWID 6500 HZ;*WAI`
Med Wide (4000 Hz)	`SENS[1|2]:BWID 4000 HZ;*WAI`
Medium (3700 Hz)	`SENS[1|2]:BWID 3700 HZ;*WAI`
Med Narrow (1200 Hz)	`SENS[1|2]:BWID 1200 HZ;*WAI`
Narrow (250 Hz)	`SENS[1|2]:BWID 250 HZ;*WAI`
Fine (15 Hz)	`SENS[1|2]:BWID 15 HZ;*WAI`
Fault Window[2]	(menu selection only)
Minimum	`SENS[1|2]:WIND RECT`
Medium	`SENS[1|2]:WIND HAMM`
Maximum	`SENS[1|2]:WIND KBES`
Delay Aperture[3]	(menu selection only)
Aperture (Hz)(enter value)	`CALC[1|2]:GDAP:SPAN 9950000 Hz;*WAI`
Aperture (%)(enter value)	`CALC[1|2]:GDAP:APER <%/100>;*WAI`

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. Option 100 (SRL and Fault Location) only.
3. Use with Delay format only.

Table 10-2 BEGIN Functions, 8712ET and 8714ET (1 of 3)

KEYSTROKES	SCPI COMMAND
BEGIN[1]	(hardkey entry)
Amplifier	(menu selection only)
Transmissn	CONF 'AMPL:TRAN';*WAI
Reflection	CONF 'AMPL:REFL';*WAI
Power	CONF 'AMPL:POW';*WAI
Filter	(menu selection only)
Transmissn	CONF ‘FILT:TRAN’;*WAI
Reflection	CONF ‘FILT:REFL’;*WAI
Broadband Passive	(menu selection only)
Transmissn	CONF ‘BBAN:TRAN’;*WAI
Reflection	CONF ‘BBAN:REFL’;*WAI
Mixer	(menu selection only)
Conversion Loss	CONF ‘MIX:CLOS’;*WAI
Reflection	CONF ‘MIX:REFL’;*WAI
Cable[2]	(menu selection only)
Transmissn	CONF[1|2] ‘CABL:TRAN’;*WAI
Reflection	CONF[1|2] ‘CABL:REFL’;*WAI
Fault Location	CONF[1|2] ‘CABL:FAULT’;*WAI
Start Distance	SENS[1|2]:DIST:STAR <num>[FEET|MET];*WAI
Stop Distance	SENS[1|2]:DIST:STOP <num>[FEET|MET];*WAI

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. Option 100 (SRL and Fault Location) only.

Table 10-2 BEGIN Functions, 8712ET and 8714ET (2 of 3)

KEYSTROKES	SCPI COMMAND
BEGIN[1], Cable[2], (continued)	
Fault Location, (continued)	
Feet	SENS:DIST:UNIT FEET
Meters	SENS:DIST:UNIT MET
Low Pass	SENS:FREQ:MODE LOWP;*WAI
Band Pass	SENS:FREQ:MODE CENT;*WAI
Center Frequency	SENS[1|2]:FREQ:CENT <value>;*WAI
SRL	CONF[1|2] 'CABL:SRL';*WAI
Start Freq	DISP:ANN:FREQ[1|2]:MODE SSTOP
(enter value and units)	SENS[1|2]:FREQ:STAR <num> [MHZ|KHZ|HZ];*WAI
Stop Freq	DISP:ANN:FREQ[1|2]:MODE SSTOP
(enter value and units)	SENS[1|2]:FREQ:STOP <num> [MHZ|KHZ|HZ];*WAI
Connector Model	(menu selection only)
Measure Connector	SENS[1|2]:CORR:MODEL:CONN
Connector Length	SENS[1|2]:CORR:LENG:CONN <num>
Connector C	SENS[1|2]:CORR:CAP:CONN <num>
Z Cutoff Frequency	SENS:FREQ:ZST <num>
Auto Z ON off	SENS1:FUNC:SRL:MODE [MANUAL|AUTO]
Manual Z	SENS1:FUNC:SRL:IMP <num>

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. Option 100 (SRL and Fault Location) only.

Table 10-2 BEGIN Functions, 8712ET and 8714ET (3 of 3)

KEYSTROKES	SCPI COMMAND
BEGIN[1], Cable[2], (continued)	
SRL, (continued)	
Connector Fault	SENS1:FUNC:FAULT:CONN; *WAI
SRL Cable Scan	SENS[1|2]:FUNC:SRL:SCAN; *WAI
Number of Points	SENS[1|2]:SWE:POIN <num>;*WAI
Autost	(menu selection only)
User Begin on OFF	(menu selection only)

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. Option 100 (SRL and Fault Location) only.

Table 10-3 BEGIN Functions, 8712ES and 8714ES (1 of 3)

KEYSTROKES	SCPI COMMAND
BEGIN[1]	
Amplifier	(menu selection only)
S11 Refl Port1	CONF 'AMPL:REFL'; *WAI
S21 Fwd Trans	CONF 'AMPL:TRAN'; *WAI
S12 Rev Trans	CONF 'AMPL:TRAN:REV'; *WAI
S22 Refl Port2	CONF 'AMPL:REFL:REV'; *WAI
Power	CONF 'AMPL:POW'; *WAI
Filter	(menu selection only)
S11 Refl Port1	CONF 'FILT:REFL'; *WAI
S21 Fwd Trans	CONF 'FILT:TRAN'; *WAI
Broadband Passive	(menu selection only)
S11 Refl Port1	CONF 'BBAN:REFL'; *WAI
S21 Fwd Trans	CONF 'BBAN:TRAN'; *WAI
S12 Rev Trans	CONF 'BBAN:TRAN:REV'; *WAI
S22 Refl Port2	CONF 'BBAN:REFL:REV'; *WAI
Mixer	(menu selection only)
Conversion Loss	CONF 'MIX:CLOS'; *WAI
S11 Refl Port1	CONF 'MIX:REFL'; *WAI

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.

Table 10-3 BEGIN Functions, 8712ES and 8714ES (2 of 3)

KEYSTROKES	SCPI COMMAND
BEGIN,(continue)	
Cable[1] (continue)	
Transmissn	`CONF[1|2] 'CABL:TRAN';*WAI`
Reflection	`CONF[1|2] 'CABL:REFL';*WAI`
Fault Location	`CONF[1|2] 'CABL:FAULT';*WAI`
Start Distance	`SENS[1|2]:DIST:STAR <num>[FEET|MET];*WAI`
Stop Distance	`SENS[1|2]:DIST:STOP <num>[FEET|MET];*WAI`
Feet	`SENS:DIST:UNIT FEET`
Meters	`SENS:DIST:UNIT MET`
Low Pass	`SENS:FREQ:MODE LOWP;*WAI`
Band Pass	`SENS:FREQ:MODE CENT;*WAI`
Center Frequency	`SENS[1|2]:FREQ:CENT <value>;*WAI`
SRL	`CONF[1|2] 'CABL:SRL';*WAI`
Start Freq	`DISP:ANN:FREQ[1|2]:MODE SSTOP`
(enter value and units)	`SENS[1|2]:FREQ:STAR <num>[MHZ|KHZ|HZ];*WAI`
Stop Freq	`DISP:ANN:FREQ[1|2]:MODE SSTOP`
(enter value and units)	`SENS[1|2]:FREQ:STOP <num>[MHZ|KHZ|HZ];*WAI`

1. Option 100 (SRL and Fault Location) only.

Table 10-3 BEGIN **Functions, 8712ES and 8714ES (3 of 3)**

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>BEGIN, Cable,[1,2] (continued)</td><td></td></tr>
<tr><td>SRL, (continued)</td><td></td></tr>
<tr><td>Connector Model</td><td>(menu selection only)</td></tr>
<tr><td>Measure Connector</td><td><code>SENS[1|2]:CORR:MODEL:CONN</code></td></tr>
<tr><td>Connector Length</td><td><code>SENS[1|2]:CORR:LENG:CONN <num></code></td></tr>
<tr><td>Connector C</td><td><code>SENS[1|2]:CORR:CAP:CONN <num></code></td></tr>
<tr><td>Z Cutoff Frequency</td><td><code>SENS:FREQ:ZST <num></code></td></tr>
<tr><td>Auto Z ON off</td><td><code>SENS1:FUNC:SRL:MODE [MANUAL|AUTO]</code></td></tr>
<tr><td>Manual Z</td><td><code>SENS1:FUNC:SRL:IMP <num></code></td></tr>
<tr><td>Connector Fault</td><td><code>SENS1:FUNC:FAULT:CONN; *WAI</code></td></tr>
<tr><td>SRL Cable Scan</td><td><code>SENS[1|2]:FUNC:SRL:SCAN; *WAI</code></td></tr>
<tr><td>Number of Points</td><td><code>SENS[1|2]:SWE:POIN <num>;*WAI</code></td></tr>
<tr><td>Autost</td><td>(menu selection only)</td></tr>
<tr><td>User Begin on OFF</td><td>(menu selection only)</td></tr>
</table>

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. Option 100 (SRL and Fault Location) only.

Table 10-4 CAL Functions, 8712ES and 8714ES (1 of 3)

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>CAL, (Reflection)</td><td>(hardkey entry)</td></tr>
<tr><td>Default 1-Port</td><td>SENS[1|2]:CORR:CSET DEF; *WAI</td></tr>
<tr><td>Default 2-Port</td><td>SENS[1|2]:CORR:CLASS DEF2; *WAI</td></tr>
<tr><td>User 1-Port</td><td>SENS[1|2]:CORR:CLASS DEF1; *WAI</td></tr>
<tr><td>Default 1-Port</td><td>SENS[1|2]:CORR:CSET DEF; *WAI</td></tr>
<tr><td>User 1-Port</td><td>SENS[1|2]:CORR:COLL:IST OFF;METH REFL3; *WAI</td></tr>
<tr><td>Measure Standard[1]</td><td>SENS[1|2]:CORR:COLL STAN[1|2|3];*WAI</td></tr>
<tr><td>User 2-Port</td><td>SENS[1|2]:CORR:CLASS DEF2; *WAI</td></tr>
<tr><td>Default 2-Port</td><td>SENS[1|2]:CORR:CLASS DEF2; *WAI</td></tr>
<tr><td>User 2-Port</td><td>SENS[1|2]:CORR:COLL:IST OFF;METH TWOP; *WAI</td></tr>
<tr><td>Measure Standard[2]</td><td>SENS[1|2]:CORR:COLL STAN[1|2|...|7];*WAI</td></tr>
<tr><td>Isolation on OFF</td><td>SENS:CORR:ISOL ON; *WAI</td></tr>
<tr><td>Normalize on OFF</td><td>TRAC CH[1|2]SMEM,CH[1|2]SDATA;:CALC[1|2]:MATH (IMPL/CH[1|2]SMEM);:DISP:WIND[1|2]:TRAC[1|2] ON;TRAC[1|2] OFF</td></tr>
<tr><td>More Cal (see Table 10-11)</td><td>(menu selection only)</td></tr>
</table>

1. Pressing this key begins a guided calibration procedure using three standards. When the calibration is complete, the command `*WAI;:SENS[1|2]:CORR:COLL:SAVE;*WAI` is executed. The order of appearance and the content of the softkey menus depend on the measurement channel 1 and measurement channel 2 configurations.
2. Pressing this softkey begins a guided 2-port calibration procedure, using seven standards. When the calibration is complete, the command `*WAI;:SENS[1|2]:CORR:COLL:SAVE;*WAI` is executed. The order of appearance and the content of the softkey menus depend on the measurement channel 1 and measurement channel 2 configurations.

Table 10-4 CAL, Functions, 8712ES and 8714ES (2 of 3)

KEYSTROKES	SCPI COMMAND
CAL, (Transmissn) (continued)	
Default Response	SENS[1|2]:CORR:CSET DEF; *WAI
Default 2-Port	SENS[1|2]:CORR:CLASS DEF2; *WAI
User Response	SENS[1|2]:CORR:CLASS DEF1; *WAI
Default Response	SENS1:CORR:CSET DEF; *WAI
Response	SENS[1|2:CORR:COLL:IST OFF;METH TRAN1; *WAI
Measure Standard[1]	SENS[1|2]:CORR:COLL STAN[1|2|3];*WAI
Response & Isolation	SENS[1|2:CORR:COLL:IST OFF;METH TRAN2; *WAI
Measure Standard[2]	SENS[1|2]:CORR:COLL STAN[1|2|3];*WAI
Enhanced Response	SENS[1|2:CORR:COLL:IST OFF;METH TRAN3; *WAI
Measure Standard[3]	SENS[1|2]:CORR:COLL STAN[1|2|3];*WAI

1. Pressing this key begins a guided calibration procedure using one standard. When the calibration is complete, the command `*WAI;:SENS[1|2]:CORR:COLL:SAVE;*WAI` is executed.
2. Pressing this key begins a guided calibration procedure using two standards. When the calibration is complete, the command `*WAI;:SENS[1|2]:CORR:COLL:SAVE;*WAI` is executed.
3. Pressing this key begins a guided calibration procedure using four standards. When the calibration is complete, the command `*WAI;:SENS[1|2]:CORR:COLL:SAVE;*WAI` is executed. The order of appearance and the content of the softkey menus depend on the measurement channel 1 and measurement channel 2 configuration.

Table 10-4 CAL, Functions, 8712ES and 8714ES (3 of 3)

KEYSTROKES	SCPI COMMAND
CAL, (Transmissn) (continued)	
User 2-Port	`SENS[1|2]:CORR:CLASS DEF2; *WAI`
Default 2-Port	`SENS[1|2]:CORR:CLASS DEF2; *WAI`
User 2-Port	`SENS[1|2]:CORR:COLL:IST OFF;METH TWOP; *WAI`
Measure Standard[1]	`SENS[1|2]:CORR:COLL STAN[1|2|...|7];*WAI`
Isolation on OFF	`SENS[1|2]:ISOL [ON|OFF];*WAI`
Normalize on OFF	`TRAC CH[1|2]SMEM,CH[1|2]SDATA;:CALC[1|2]:MATH (IMPL/CH[1|2]SMEM);:DISP:WIND[1|2]:TRAC[1|2] ON;TRAC[1|2] OFF`
More Cal (see Table 10-11 on page 10-26)	(menu selection only)
See Table 10-11 on page 10-30 for Test Set calibration.	
See Table 10-7 on page 10-22 for Fault Location calibration.	
See Table 10-8 on page 10-23 for SRL calibration.	
See Table 10-9 on page 10-24 for Power or Conversion Loss calibration.	

1. Pressing this softkey begins a guided 2-port calibration procedure, using seven standards. When the calibration is complete, the command `*WAI;:SENS[1|2]:CORR:COLL:SAVE;*WAI` is executed. The order of appearance and the content of the softkey menus depend on the measurement channel 1 and measurement channel 2 configurations.

Table 10-5 CAL Functions, 8712ET and 8714ET (1 of 3)

KEYSTROKES	SCPI COMMAND
CAL, (Reflection)	(hardkey entry)
Default 1-Port	SENS[1|2]:CORR:CSET DEF; *WAI
1-Port	SENS[1|2]:CORR:COLL:IST OFF;METH REFL3; *WAI
Measure Standard[1]	SENS[1|2]:CORR:COLL:IST OFF;METH REFL3; *WAI
Measure Standard — Open	SENS[1|2]:CORR:COLL STAN1;*WAI
Measure Standard — Short	SENS[1|2]:CORR:COLL STAN2;*WAI
Measure Standard — Load	SENS[1|2]:CORR:COLLSTAN3;*WAI; :SENS[1|2]:CORR:COLL:SAVE;*WAI
Normalize on OFF	TRAC CH[1|2]SMEM,CH[1|2]SDATA;:CALC[1|2]: MATH (IMPL/CH[1|2]SMEM);:DISP:WIND[1|2]: TRAC[1|2] ON;TRAC[1|2] OFF
Cal Check (see Table 10-10 on page 10-25)	(menu selection only)

1. Pressing this softkey begins a guided 1-port calibration procedure, using three standards. When the calibration is complete, the command `*WAI;:SENS[1|2]:CORR:COLL:SAVE;*WAI` is executed. The order of appearance and the content of the softkey menus depend on the measurement channel 1 and measurement channel 2.

Table 10-5 CAL **Functions, 8712ET and 8714ET (2 of 3)**

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>CAL, (Transmissn)</td><td>(hardkey entry)</td></tr>
<tr><td>Default Response</td><td>SENS[1|2]:CORR:CSET DEF; *WAI</td></tr>
<tr><td>Response</td><td>SENS[1|2]:CORR:COLL:IST OFF;METH TRAN1;
*WAI</td></tr>
<tr><td>Measure Standard</td><td>SENS[1|2]:CORR:COLL STAN1;*WAI
:SENS[1|2]:CORR:COLL:SAVE;*WAI</td></tr>
<tr><td>Response & Isolation</td><td>SENS[1|2]:CORR:COLL:IST OFF;METH TRAN2;
*WAI</td></tr>
<tr><td>Measure Standard - Load</td><td>SENS[1|2]:CORR:COLL STAN1;*WAI</td></tr>
<tr><td>Measure Standard - Through</td><td>SENS[1|2]:CORR:COLL STAN2;*WAI</td></tr>
<tr><td>Enhanced Response</td><td>SENS[1|2]:CORR:COLL:IST OFF;METH TRAN3;
*WAI</td></tr>
<tr><td>Measure Standard - Open</td><td>SENS[1|2]:CORR:COLL STAN1;*WAI</td></tr>
<tr><td>Measure Standard - Short</td><td>SENS[1|2]:CORR:COLL STAN2;*WAI</td></tr>
<tr><td>Measure Standard - Load</td><td>SENS[1|2]:CORR:COLLSTAN3;*WAI;
:SENS[1|2]:CORR:COLL:SAVE;*WAI</td></tr>
<tr><td>Measure Standard - Through</td><td>SENS[1|2]:CORR:COLL STAN4;*WAI
:SENS[1|2]:CORR:COLL:SAVE;*WAI</td></tr>
</table>

Table 10-6 CAL Functions, 8712ET and 8714ET (3 of 3)

KEYSTROKES	SCPI COMMAND
CAL, (continued) (Transmissn)	
Normalize on OFF	`TRAC CH[1|2]SMEM,CH[1|2]SDATA;:CALC[1|2]:MATH (IMPL/CH[1|2]SMEM);:DISP:WIND[1|2]:TRAC[1|2] ON;TRAC[1|2] OFF`
Cal Check (see Table 10-10 on page 10-25)	(menu selection only)
More Cal (see Table 10-11 on page 10-26)	(menu selection only)
See Table 10-11 on page 10-30 for Test Set calibration.	
See Table 10-7 on page 10-22 for Fault Location calibration.	
See Table 10-8 on page 10-23 for SRL calibration.	
See Table 10-9 on page 10-24 for Power or Conversion Loss calibration.	

Table 10-7 CAL, Functions, 8712ET/ES and 8714ET/ES

KEYSTROKES	SCPI COMMAND
CAL, (Fault Location)	
Default Cal	`SENS[1|2]:CORR:CSET DEF; *WAI`
Full Band Cal	`SENS[1|2]:CORR:EXT [ON|OFF]`
Measure Standard[1]	`SENS[1|2]:CORR:COLL STAN[1|2|3];*WAI`
Velocity Factor	SENS[1|2]:CORR:RVEL:COAX <value>
Cable Loss	SENS[1|2]:CORR:LOSS:COAX <value>
Calibrate Cable	(menu selection only)
Specify Length	SENS[1|2]:CORR:LENG:COAX <value>; *WAI
Measure Cable	`SENS[1|2]:CORR:RVEL; *WAI`
Multi Peak	(menu selection only)
Multi Peak Corr on OFF	`SENS[1|2]:CORR:PEAK:COAX ON`
Multi Peak Threshold	`SENS[1|2]:CORR:THRESHOLD:COAX -10.00`
Connector Values	(menu selection only)
Connector Length	SENS[1|2]:CORR:LENG:CONN <value>
Connector C	SENS1:CORR:CAP:CONN <value>
More Cal (see Table 10-11 on page 10-26)	(menu selection only)

1. Pressing this key begins a guided calibration procedure using three standards. When the calibration is complete, the command `*WAI;:SENS[1|2]:CORR:COLL:SAVE;*WAI` is executed. The order of appearance and the content of the softkey menus depend on the measurement channel 1 and measurement channel 2 configuration.

Table 10-8 Functions, 8712ET/ES and 8714ET/ES

KEYSTROKES	SCPI COMMAND
CAL, (SRL)	
Default Cal	SENS[1|2]:CORR:CSET DEF; *WAI
Full Band Cal	SENS[1|2]:CORR:EXT [ON|OFF]
Measure Standard[1]	SENS[1|2]:CORR:COLL STAN[1|2|3];*WAI
Connector Model	(menu selection only)
Measure Connector	(menu selection only)
Measure	SENS[1|2]:CORR:MODEL:CONN
Connector Length	SENS[1|2]:CORR:LENG:CONN <num>
Connector C	SENS[1|2]:CORR:CAP:CONN <num>
Z cutoff Frequency	SENS:FREQ:ZST <num>
Auto Z ON off	SENS[1|2]:FUNC:SRL:MODE [MANUAL|AUTO]
Manual Z	SENS[1|2]:FUNC:SRL:IMP <num>
Connector Fault	(sets instrument to connector fault display)
Cal Check (see Table 10-10 on page 10-25)	(menu selection only)
More Cal (see Table 10-11 on page 10-26)	(menu selection only)

1. Pressing this key begins a guided calibration procedure using three standards. When the calibration is complete, the command `*WAI;:SENS[1|2]:CORR:COLL:SAVE;*WAI` is executed.

Table 10-9 CAL, Functions, 8712ET/ES and 8714ET/ES

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>CAL, (Power or Conversion Loss)</td><td></td></tr>
<tr><td>Auto Zero</td><td>CAL:ZERO:AUTO ON</td></tr>
<tr><td>Manual Zero</td><td>CAL:ZERO:AUTO ONCE</td></tr>
<tr><td>Normalize on OFF</td><td>TRAC CH[1|2]SMEM,CH[1|2]SDATA;:CALC[1|2]:MATH (IMPL/CH[1|2]SMEM);:DISP:WIND[1|2]:TRAC[1|2] ON;TRAC[1|2] OFF</td></tr>
<tr><td>More Cal
(see Table 10-11 on page 10-26)</td><td>(menu selection only)</td></tr>
</table>

Table 10-10 **CAL, Cal Check Functions, 8712ET/ES and 8714ET/ES**

KEYSTROKES	SCPI COMMAND
CAL, Cal Check	(menu selection only)
Do Cal Check	SENS1:CORR:COLL:IST OFF;METH VERIFY;*WAI
Measure Standard[1]	SENS[1|2]:CORR:COLL:VER:REFL STAN[1|2|3];*WAI;
View Cal Check[1]	(menu selection only)
Directivity	DIAG:MDIS[1|2]:CORR C_DIRECT; *WAI
Source Match	DIAG:MDIS[1|2]:CORR C_SRCMATCH; *WAI
Reflection Tracking	DIAG:MDIS[1|2]:CORR C_RTRACKING; *WAI
Load Match	DIAG:MDIS[1|2]:CORR C_LDMATCH; *WAI
Transmissn Tracking	DIAG:MDIS[1|2]:CORR C_TTRACKING; *WAI
Isolation	DIAG:MDIS[1|2]:CORR C_ISOLATION; *WAI
Restore Meas	DIAG:MDIS[1|2]:REST; *WAI

1. Only those calibration checks valid for the current measurement type are active, other choices are dimmed. For example, **Transmission Tracking** is not valid for reflection measurements.

Table 10-11 CAL, More CAL Functions, 8712ET/ES and 8714ET/ES (1 of 4)

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>CAL, More Cal[1, 2]</td><td>(menu selection only)</td></tr>
<tr><td>Port Extensions</td><td>(menu selection only)</td></tr>
<tr><td>Port Ext's ON off</td><td>SENS[1|2]:CORR:EXT [ON|OFF]</td></tr>
<tr><td>Port 1 Extension[3, 4]
(Refl Port Extensions)[3, 5]</td><td>SENS[1|2]:CORR:EXT:REFL:TIME <num> s</td></tr>
<tr><td>Port 2 Extension[3, 4]
(Trans Port Extensions)[3, 5]</td><td>SENS[1|2]:CORR:EXT:TRAN:TIME <num> s</td></tr>
<tr><td>Velocity Factor</td><td>SENS[1|2]:CORR:RVEL:COAX <num></td></tr>
<tr><td>Smith Chart Z0</td><td>SENS[1|2]:CORR:IMP:INP:MAGN <num> OHM</td></tr>
<tr><td>System Z0</td><td>SENS[1|2]:CORR:IMP:INP:MAGN:SEL
Z0_[50|75]</td></tr>
<tr><td>50 Ω</td><td>SENS[1|2]:CORR:EXT:REFL:TIME <num> s</td></tr>
<tr><td>75 Ω</td><td>SENS[1|2]:CORR:EXT:REFL:TIME <num> s</td></tr>
<tr><td>Cal Kit</td><td>(menu selection only)</td></tr>
<tr><td>Type-N(f) (Default)</td><td>SENS:CORR:COLL:CKIT:PORT[1|2]
'COAX,7MM,TYPE-N,75,FEMALE'</td></tr>
<tr><td>Type-N(m)</td><td>SENS:CORR:COLL:CKIT:PORT[1|2]
'COAX,7MM,TYPE-N,75,MALE'</td></tr>
</table>

1. The port number in each command is the port selected from the screen using the up and down keys.
2. The instrument system impedance determines the set of valid connector choices in this menu. Connectors with a characteristic impedance other than the instrument system impedance are dimmed on the screen.
3. This Key is active if port extensions are on.
4. For 8712ES and 8714ES analyzers.
5. For 8712ET and 8714ET analyzers.

Table 10-10 CAL, More CAL Functions, 8712ET/ES and 8714ET/ES (2 of 4)

KEYSTROKES	SCPI COMMAND
CAL, More Cal (continued)	(menu selection only)
Cal Kit (continued)	(menu selection only)
3.5 mm	SENS:CORR:COLL:CKIT:PORT1 'COAX,3.5MM,APC-3.5,50,IMPLIED'
Type-F	SENS:CORR:COLL:CKIT:PORT1 'COAX,7MM,TYPE-F,75,IMPLIED'
APC-7	SENS:CORR:COLL:CKIT:PORT1 'COAX,7MM,APC-7,50,IMPLIED'
7-16	SENS:CORR:COLL:CKIT:PORT1 'COAX,7MM,TYPE-7-16,50,IMPLIED'
User Cal Kit A[1]	SENS:CORR:COLL:CKIT:PORT[1|2] 'USER,IMPLIED,IMPLIED,IMPLIED,IMPLIED'
User Cal Kit B[1]	SENS:CORR:COLL:CKIT:PORT[1|2] 'USER2,IMPLIED,IMPLIED,IMPLIED,IMPLIED'
User Cal Kit C[1]	SENS:CORR:COLL:CKIT:PORT[1|2] 'USER3,IMPLIED,IMPLIED,IMPLIED,IMPLIED'
User Cal Kit D[1]	SENS:CORR:COLL:CKIT:PORT[1|2] 'USER4,IMPLIED,IMPLIED,IMPLIED,IMPLIED'
User Cal Kit E[1]	SENS:CORR:COLL:CKIT:PORT[1|2] 'USER5,IMPLIED,IMPLIED,IMPLIED,IMPLIED'
User Cal Kit F[1]	SENS:CORR:COLL:CKIT:PORT[1|2] 'USER6,IMPLIED,IMPLIED,IMPLIED,IMPLIED'
User Cal Kit G[1]	SENS:CORR:COLL:CKIT:PORT[1|2] 'USER7,IMPLIED,IMPLIED,IMPLIED,IMPLIED'

1. The port number in each command is the port selected from the screen using the up and down keys.

Table 10-10 CAL, More CAL Functions, 8712ET/ES and 8714ET/ES (3 of 4)

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>CAL, More Cal (continued)</td><td></td></tr>
<tr><td>Cal Kit (continued)</td><td></td></tr>
<tr><td>User Cal
Kit H[1]</td><td>SENS:CORR:COLL:CKIT:PORT[1|2]
'USER8,IMPLIED,IMPLIED,IMPLIED,IMPLIED'</td></tr>
<tr><td>User Cal
Kit I[1]</td><td>SENS:CORR:COLL:CKIT:PORT[1|2]
'USER9,IMPLIED,IMPLIED,IMPLIED,IMPLIED'</td></tr>
<tr><td>User Cal
Kit J[1]</td><td>SENS:CORR:COLL:CKIT:PORT[1|2]
'USER10,IMPLIED,IMPLIED,IMPLIED,IMPLIED'</td></tr>
<tr><td>Modify (Cal Kit Type)</td><td>SENS[1|2]:CORR:CKIT:MOD[:SEL][TYPenf|
TYPenm|UD1|TYPe35mm|TYPeff|TYPe716f|
TYPe716m|UD2|TYPeapc7|UD3|4|...10]</td></tr>
<tr><td>Open:</td><td>(menu selection only)</td></tr>
<tr><td>C0</td><td>SENS[1|2]:CORR:CKIT:OPEN:MOD:CZER <num></td></tr>
<tr><td>C1</td><td>SENS[1|2]:CORR:CKIT:OPEN:MOD:CONE <num></td></tr>
<tr><td>C2</td><td>SENS[1|2]:CORR:CKIT:OPEN:MOD:CTWO <num></td></tr>
<tr><td>C3</td><td>SENS[1|2]:CORR:CKIT:OPEN:MOD:CTHR <num></td></tr>
<tr><td>Delay</td><td>SENS[1|2]:CORR:CKIT:OPEN:MOD:DEL <num></td></tr>
<tr><td>Loss</td><td>SENS[1|2]:CORR:CKIT:OPEN:MOD:LOSS <num></td></tr>
<tr><td>Z0</td><td>SENS[1|2]:CORR:CKIT:OPEN:MOD:ZOFF <num></td></tr>
</table>

Table 10-10 CAL, More CAL Functions, 8712ET/ES and 8714ET/ES (4 of 4)

KEYSTROKES	SCPI COMMAND
CAL, More Cal, (continued)	
Cal kit, (continued)	
Modify (*Cal Kit Type*) (continued)	
Short:	(menu selection only)
Delay	`SENS[1|2]:CORR:CKIT:SHOR:MOD:DEL <num>`
Loss	`SENS[1|2]:CORR:CKIT:SHOR:MOD:LOSS <num>`
Z0	`SENS[1|2]:CORR:CKIT:OPEN:MOD:ZOFF <num>`
Load:	(menu selection only)
Delay	`SENS[1|2]:CORR:CKIT:LOAD:MOD:DEL <num>`
Loss	`SENS[1|2]:CORR:CKIT:LOAD:MOD:LOSS <num>`
Z0	`SENS[1|2]:CORR:CKIT:LOAD:MOD:ZOFF <num>`
Thru	(menu selection only)
Delay	`SENS[1|2]:CORR:CKIT:THRU:MOD:DEL <num>`
Loss	`SENS[1|2]:CORR:CKIT:THRU:MOD:LOSS <num>`
Z0	`SENS[1|2]:CORR:CKIT:THRU:MOD:ZOFF <num>`
Mod Kit Save Recall	(menu selection only)
Save	`SENS:CORR:CKIT:SAVE-? <KIT1|2...|10>`
Recall	`SENS:CORR:CKIT:MOD <UD1|2|...10>`
Add Kit Description	`SENS:CORR:CKIT:NAME <KIT1|2...|10>, <STRING>`
Preset User Kit	(menu selection only)
Preset All User Kits	`SENS:CORR:CKIT:PRES[:IMM]-? [{KIT1|2|...|10}]`

Table 10-11 CAL, Test Set Cal Functions, 8712ET/ES and 8714ET/ES

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>CAL, Test Set Cal[1]</td><td>SENS[1|2]:CORR:TESTSET;*WAI</td></tr>
<tr><td>Create "TSET_CAL"[2]</td><td>SENS[1|2]:CORR:COLL:METHOD TEST;</td></tr>
<tr><td>XX Ports</td><td>SENS[1|2]:CORR:COLL:PORTS
<2|4|6|8|10|12></td></tr>
<tr><td>Enh Resp 1 - Port [3]</td><td>SENS[1|2]:CORR:COLL:METHOD TEST 1</td></tr>
<tr><td>2 - Port[3]</td><td>SENS[1|2]:CORR:COLL:METHOD TEST 2</td></tr>
<tr><td>Measure Opens</td><td>SENS[1|2]:CORR:COLL:MP:OPEN
<STAN1|STAN2|...|STAN12>;*WAI;</td></tr>
<tr><td>Measure Shorts</td><td>SENS[1|2]:CORR:COLL:MP:SHORT
<STAN1|STAN2|...|STAN12>;*WAI;</td></tr>
<tr><td>Measure Loads</td><td>SENS[1|2]:CORR:COLL:MP:LOAD
<STAN1|STAN2|...|STAN12>;*WAI;</td></tr>
<tr><td>Measure Thrus</td><td>SENS[1|2]:CORR:COLL:MP:THRU
<STAN1|STAN2|...|STAN6>;*WAI;</td></tr>
<tr><td>All Stds Done</td><td>SENS[1|2]:CORR:COLL:SAVE;*WAI;</td></tr>
<tr><td>Periodic SelfCal</td><td>CAL:SELF ON</td></tr>
<tr><td>SelfCal Once</td><td>CAL:SELF ONCE</td></tr>
<tr><td>SelfCal Timer</td><td>CAL:SELF:TIMER <num></td></tr>
<tr><td>SelfCal All Ports</td><td>CAL:SEL0F:ALL</td></tr>
<tr><td>2-Port Cal on OFF[3]</td><td>CAL:SELF:METHOD [ONEP|TWOP]*WAI</td></tr>
<tr><td>Isolation on OFF[3]</td><td>SENS[1|2]:CORR:ISOL [ON|OFF]*WAI</td></tr>
</table>

1. For use with multiport test sets only.
2. You select the number of test set ports, and a guided calibration procedure begins.
3. For 8712ES and 8714ES analyzers only.

Table 10-12 DISPLAY Functions, 8712ET/ES and 8714ET/ES (1 of 5)

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>DISPLAY[1]</td><td>(hardkey entry)</td></tr>
<tr><td>Normalize</td><td>TRAC CH[1|2]SMEM,CH[1|2]SDATA;:
CALC[1|2]:MATH (IMPL/CH[1|2]SMEM);:
DISP: WIND[1|2]:TRAC1 ON;TRAC2 OFF</td></tr>
<tr><td>Data ->Mem</td><td>TRAC CH[1|2]SMEM,CH[1|2]SDATA</td></tr>
<tr><td>Data</td><td>CALC[1|2]:MATH (IMPL);:DISP:
WIND[1|2]:TRAC1 ON;TRAC2 OFF</td></tr>
<tr><td>Memory</td><td>DISP:WIND[1|2]:TRAC1 OFF;TRAC2 ON</td></tr>
<tr><td>Data/Mem</td><td>CALC[1|2]:MATH (IMPL/CH[1|2]SMEM);:
DISP:WIND[1|2]:TRAC1 ON;TRAC2 OFF</td></tr>
<tr><td>Data and Memory</td><td>CALC[1|2]:MATH
(IMPL);:DISP:WIND[1|2]:TRAC1 ON;TRAC2 ON</td></tr>
<tr><td>Limit Menu[2]</td><td>CALC[1|2]:LIM:DISP ON</td></tr>
<tr><td>Add Limit</td><td>(menu selection only)</td></tr>
<tr><td>Add Max Line</td><td>CALC[1|2]:LIM:SEGM[n]:TYPE LMAX;STAT ON[3]</td></tr>
<tr><td>Begin Frequency</td><td>CALC[1|2]:LIM:SEGM[n]:FREQ:STAR <num> HZ</td></tr>
<tr><td>End Frequency</td><td>CALC[1|2]:LIM:SEGM[n]:FREQ:STOP <num> HZ</td></tr>
<tr><td>Begin Limit</td><td>CALC[1|2]:LIM:SEGM[n]:AMPL:STAR <num></td></tr>
<tr><td>End Limit</td><td>CALC[1|2]:LIM:SEGM[n]:AMPL:STOP <num></td></tr>
<tr><td>Marker</td><td>CALC[1|2]:MARK1:X <num></td></tr>
</table>

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. Limit Menu choices are provided based on the current measurement channel configuration. If the instrument is configured for fault location measurements, the limit values are distances, for example.
3. Limit segments are numbered by the instrument as they are entered. At the same time the Limit menu is displayed, the currently defined limits are displayed.

Table 10-12 DISPLAY Functions, 8712ET/ES and 8714ET/ES (2 of 5)

KEYSTROKES	SCPI COMMAND
DISPLAY[1], **Limit Menu**, (continued)	
Add Limit (continued)	
Add Min Line	CALC[1|2]:LIM:SEGM[n]:TYPE LMIN;STAT ON[2]
Begin Frequency	CALC[1|2]:LIM:SEGM[n]:FREQ:STAR <num> HZ
End Frequency	CALC[1|2]:LIM:SEGM[n]:FREQ:STOP <num> HZ
Begin Limit	CALC[1|2]:LIM:SEGM[n]:AMPL:STAR <num>
End Limit	CALC[1|2]:LIM:SEGM[n]:AMPL:STOP <num>
Marker	CALC[1|2]:MARK1:X <num>
Add Max Point	CALC[1|2]:LIM:SEGM[n]:TYPE PMAX;STAT ON[2]
Frequency	CALC[1|2]:LIM:SEGM[n]:FREQ:STAR <num> HZ
Limit	CALC[1|2]:LIM:SEGM[n]:AMPL:STAR <num>
Marker	CALC[1|2]:MARK1:X <num>
Add Min Point	CALC[1|2]:LIM:SEGM[n]:TYPE PMIN;STAT ON[2]
Frequency	CALC[1|2]:LIM:SEGM[n]:FREQ:STAR <num> HZ
Limit	CALC[1|2]:LIM:SEGM[n]:AMPL:STAR <num>
Marker	CALC[1|2]:MARK1:X <num>
Delete Limit[3]	(menu selection only)
Delete all Limits	CALC1:LIM:DISP [ON|OFF]

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. Limit segments are numbered by the instrument as they are entered. At the same time the Limit menu is displayed, the currently defined limits are displayed.
3. Select the limit with the **up** and **down** keys or the RPG knob on the Limit Menu display, and select **Delete Limit**.

Table 10-12 DISPLAY Functions, 8712ET/ES and 8714ET/ES (3 of 5)

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>DISPLAY[1], Limit Menu, (continued)</td><td></td></tr>
<tr><td>Delete Limit (continued)</td><td></td></tr>
<tr><td>Edit Limit[2]</td><td>(menu selection only)</td></tr>
<tr><td>Begin Frequency (Begin Distance)[2]</td><td>CALC[1|2]:LIM:SEGM[n]:FREQ:STAR <num> HZ</td></tr>
<tr><td>End Frequency (End Distance)[2]</td><td>CALC[1|2]:LIM:SEGM[n]:FREQ:STOP <num> HZ</td></tr>
<tr><td>Begin Limit</td><td>CALC[1|2]:LIM:SEGM[n]:AMPL:STAR <num></td></tr>
<tr><td>End Limit</td><td>CALC[1|2]:LIM:SEGM[n]:AMPL:STOP <num></td></tr>
<tr><td>Limit Options</td><td>CALC[1|2]:LIM:SEGM[n]:TYPE PMAX;STAT ON[2]</td></tr>
<tr><td>Limit Line ON off</td><td>CALC[1|2]:LIM:DISP [ON|OFF]</td></tr>
<tr><td>Limit Text ON off</td><td>DISP:ANN:LIM:ICON2:TEXT [ON|OFF]</td></tr>
<tr><td>Limit Icon ON off</td><td>DISP:ANN:LIM:ICON2:FLAG [ON|OFF]</td></tr>
<tr><td>Limit Icon X Position</td><td>DISP:ANN:LIM:ICON1:POS:X <value></td></tr>
<tr><td>Limit Icon Y Position</td><td>DISP:ANN:LIM:ICON1:POS:Y <value></td></tr>
<tr><td>Mkr Limits</td><td>(menu selection only)</td></tr>
<tr><td>Edit Limit (Min/Max)[3]</td><td>(menu selection only)</td></tr>
<tr><td>Max Limit</td><td>CALC[1|2]:LIM:MARK:STAT:MEAN:MAX <value></td></tr>
<tr><td>Min Limit</td><td>CALC[1|2]:LIM:MARK:STAT:MEAN:MIN <value></td></tr>
<tr><td>Mkr Limit on OFF</td><td>CALC[1|2]:LIM:MARK:STAT:MEAN [ON|OFF]</td></tr>
<tr><td>Limit Test ON off</td><td>CALC[1|2]:LIM:STAT [ON|OFF]</td></tr>
</table>

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. For fault location measurements.
3. After **Edit Limit (Min/Max)** is pressed, the **up** key can be used to increase the limit (`CALC[1|2]:LIM:MARK:STAT:MEAN:MAX UP`) and the **down** key can be used to decrease the limit (`CALC[1|2]:LIM:MARK:STAT:MEAN:MAX DOWN`).

Table 10-12 DISPLAY Functions, 8712ET/ES and 8714ET/ES (4 of 5)

KEYSTROKES	SCPI COMMAND
DISPLAY[1], (continued)	
Split Disp FULL split	`DISP:FORM [ULOW|SING]`
Expand ON off	`DISP:FORM:EXPAND [ON|OFF]`
Title and Clock	`DISP:ANN:TITL ON`
Enter Line 1 (enter text, press Enter)	`DISP:ANN:TITL[1|2]:DATA, 'text'`
Enter Line 2 (enter text, press Enter)	`DISP:ANN:TITL[1|2]:DATA, 'text'`
Show Clock on Line 1	`DISP:ANN:CLOC:MODE LINE1`
Show Clock on Line 2	`DISP:ANN:CLOC:MODE LINE2`
Clock Off	`DISP:ANN:CLOC:MODE OFF`
Title+Clk ON off	`DISP:ANN:TITL [ON|OFF]`
Color Options	(menu selection only)
Factory default	`DISP:CMAP:SCHEME DEFAULT`
Default 2	`DISP:CMAP:SCHEME DEFAULT2`
Grey Scale	`DISP:CMAP:SCHEME GREY`
Inverse Video	`DISP:CMAP:SCHEME INV`

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.

Table 10-12 DISPLAY Functions, 8712ET/ES and 8714ET/ES (5 of 5)

KEYSTROKES	SCPI COMMAND
DISPLAY[1], Color Options, (continued)	
Custom Colors[2]	(menu selection only)
Select Item	(select item, 1-16)
Hue	`DISP:CMAP:COL[1|2|...|16]:HSL h,s,l`[3]
Saturation	`DISP:CMAP:COL[1|2|...|16]:HSL h,s,l`[3]
Luminance	`DISP:CMAP:COL[1|2|...|16]:HSL h,s,l`[3]
Int Disp Intensity	`DISP:CMAP:COL16:GREY [0-1.00]`
Annotation Options	(menu selection only)
Meas Annot ON off	`DISP:ANN:CHAN[1|2] [ON|OFF]`
Freq Annot ON off	`DISP:ANN:FREQ[1|2] [ON|OFF]`
Mkr Annot ON off	`DISP:ANN:MARK[1|2] [ON|OFF]`
Mkr Number ON off	`DISP:ANN:MARK[1|2]:NUMB [ON|OFF]`
Y-Axis Lbl ON off	`DISP:ANN:YAX [ON|OFF]`
Y-Axis Lbl rel ABS	`DISP:ANN:YAX:MODE [REL|ABS]`
Graticule ON off	`DISP:WIND1:TRAC:GRAT:GRID [ON|OFF]`

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. First choose an item (Background=1, User Graphics Pen 1=2 through User Graphics Pen 7=8, Inactive Text=9, Warning Text=10, Graticule=11, Trace 1=12, Mem 1=13, Trace 2=14, Mem 2=15, Text=16) with **Select Item**, then choose a color item to change, and enter a new value. The softkey menu displays **Enter** and **Cancel**. Press **Enter** to save your changes.
3. `COL[1|2|...|16]` selects one of the 16 items listed in Footnote 2, above. `HSL h,s,l` sets the hue, saturation and luminance values, respectively.

Table 10-13 FORMAT Functions, 8712ET/ES and 8714ET/ES

KEYSTROKES	SCPI COMMAND
FORMAT[1]	(hardkey entry)
Log Mag	CALC[1|2]:FORM MLOG
Lin Mag	CALC[1|2]:FORM MLIN
SWR	CALC[1|2]:FORM SWR
Delay	CALC[1|2]:FORM GDEL
Phase	CALC[1|2]:FORM PHAS
Smith Chart	CALC[1|2]:FORM SMIT
Polar	CALC[1|2]:FORM POL
More Format	(menu selection only)
Real	CALC[1|2]:FORM REAL
Imaginary	CALC[1|2]:FORM IMAG
Impedance Magnitude	CALC[1|2]:FORM MIMP
Mag Units[2]	(menu selection only)
dBW	CALC[1|2]:FORM:UNIT:MLOG DBW
dBm	CALC[1|2]:FORM:UNIT:MLOG DBMW
dBuW	CALC[1|2]:FORM:UNIT:MLOG DBUW
dBV	CALC[1|2]:FORM:UNIT:MLOG DBV
dBmV	CALC[1|2]:FORM:UNIT:MLOG DBMV
dBuV	CALC[1|2]:FORM:UNIT:MLOG DBUV

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. This menu lists the units available for selection using Log Mag or Lin Mag format only.

Table 10-14 FREQ Functions, 8712ET/ES and 8714ET/ES

KEYSTROKES	SCPI COMMAND
FREQ[1]	(hardkey entry)
Start	DISP:ANN:FREQ1:MODE SSTOP SENS[1|2]:FREQ:STAR <value> HZ;*WAI
Stop	DISP:ANN:FREQ1:MODE SSTOP SENS[1|2]:FREQ:STOP <value> HZ;*WAI
Center	DISP:ANN:FREQ1:MODE CSPAN SENS[1|2]:FREQ:CENT <value> HZ;*WAI
Span	DISP:ANN:FREQ1:MODE CSPAN SENS[1|2]:FREQ:SPAN <value> HZ;*WAI
CW	DISP:ANN:FREQ1:MODE CW;: SENS[1|2]:FREQ:SPAN 0 HZ;;*WAI SENS[1|2]:FREQ:CENT 300000 HZ;*WAI
Fault Loc Frequency[2]	(menu selection only)
Low Pass	SENS:FREQ:MODE LOWP; *WAI
Band Pass	SENS:FREQ:MODE CENT
Band Pass Max Span	SENS[1|2]:FREQ:SPAN:MAX <value>
Displ Freq Resolution	CALC[1|2]:FORM IMAG
MHz	DISP:ANN:FREQ:RES MHZ
kHz	DISP:ANN:FREQ:RES KHZ
Hz	DISP:ANN:FREQ:RES HZ

1. The active measurement channel configuration determines the order of appearance and the content of the softkey menus.
2. Available for fault location measurements only.

Table 10-15 HARD COPY Functions, 8712ET/ES and 8714ET/ES (1 of 4)

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>HARD COPY</td><td></td></tr>
<tr><td>Start</td><td>HCOP;*WAI</td></tr>
<tr><td>Abort</td><td>HCOP:ABOR</td></tr>
<tr><td>Select Copy Port</td><td>(menu selection only)</td></tr>
<tr><td>Restore Defaults</td><td>(no SCPI command)</td></tr>
<tr><td>Select</td><td>HCOP:DEV:LANG<PCL|HPGL|IBM|EPSON|PCX>;
PORT <CENT|SER|GPIB|MMEM|LAN></td></tr>
<tr><td>LAN Printr IP Addr</td><td>SYST:COMM:LAN:PRIN:HOST <addr></td></tr>
<tr><td>Print/Plot GPIB Addr</td><td>SYST:COMM:GPIB:HCOP:ADDR <addr></td></tr>
<tr><td>Baud Rate</td><td>(no SCPI command)</td></tr>
<tr><td>Xon/Xoff</td><td>SYST:COMM:SER:TRAN:HAND XON</td></tr>
<tr><td>DTR/DSR</td><td>SYST:COMM:SER:TRAN:HAND DTR</td></tr>
<tr><td>Define PCL5</td><td>(menu selection only)</td></tr>
<tr><td>Restore Defaults</td><td>(no SCPI command)</td></tr>
<tr><td>Monochrome</td><td>HCOP:DEV3:COL [ON|OFF]</td></tr>
<tr><td>Color</td><td>HCOP:DEV3:COL [ON|OFF]</td></tr>
<tr><td>Auto Feed ON off</td><td>HCOP:ITEM3:FFE:STAT [ON|OFF]</td></tr>
<tr><td>Portrait</td><td>HCOP:DEV3:PAGE:ORI PORT</td></tr>
<tr><td>Landscape</td><td>HCOP:DEV3:PAGE:ORI LAND</td></tr>
</table>

Table 10-15 HARD COPY Functions, 8712ET/ES and 8714ET/ES (2 of 4)

KEYSTROKES	SCPI COMMAND
HARD COPY, (continued)	
Define PCL5, (continued)	
More PCL5	(menu selection only)
Restore Defaults	(no SCPI command)
Top Margin	`HCOP:DEV3:PAGE:MARG:TOP <num>`
Left Margin	`HCOP:DEV3:PAGE:MARG:LEFT <num>`
Print Width	`HCOP:DEV3:PAGE:WIDT <num>`
Define Printer	(menu selection only)
Restore Defaults	(no SCPI command)
Monochrome	`HCOP:DEV1:COL OFF`
Color	`HCOP:DEV1:COL ON`
Portrait	`HCOP:DEV3:PAGE:ORI PORT`
Landscape	`HCOP:DEV3:PAGE:ORI LAND`
Auto Feed ON off	`HCOP:ITEM3:FFE:STAT [ON|OFF]`
More Printer	(menu selection only)
Restore Defaults	(no SCPI command)
Printer Resolution	`HCOP:DEV:RES <num>`
Top Margin	`HCOP:PAGE:MARG:TOP <num>`
Left Margin	`HCOP:PAGE:MARG:LEFT <num>`
Print Width	`HCOP:PAGE:WIDT <num>`

Table 10-15 HARD COPY Functions, 8712ET/ES and 8714ET/ES (3 of 4)

KEYSTROKES	SCPI COMMAND
HARD COPY, (continued)	
Define Plotter	(menu selection only)
Restore Defaults	(no SCPI command)
Monochrome	HCOP:DEV2:COL OFF
Color	HCOP:DEV2:COL ON
Set Pen Numbers	(menu selection only)
Monochrome Pen[1]	(no SCPI command)
Default Pen Colors	(no SCPI command)
Trace 1 Pen	(no SCPI command)
Trace 2 Pen	(no SCPI command)
Memory 1 Pen	(no SCPI command)
Memory 2 Pent	(no SCPI command)
Graticule Pen	(no SCPI command)
Graphics Pen	(no SCPI command)
Auto Feed ON off	HCOP:ITEM2:FFE:STAT ON

1. This selection available when monochrome is selected.

Table 10-15 HARD COPY Functions, 8712ET/ES and 8714ET/ES (4 of 4)

KEYSTROKES	SCPI COMMAND
HARD COPY, (continued)	
Define Hardcopy	(menu selection only)
Restore Defaults	(no SCPI command)
Graph and Mkr Table	`HCOP:DEV:MODE GMAR`
Graph Only	`HCOP:DEV:MODE GRAP`
Mkr Table Only	`HCOP:DEV:MODE MARK`
List Trace Values	`HCOP:DEV:MODE TABL`
Define Graph	(menu selection only)
Restore Defaults	(no SCPI command)
Trace Data ON off	`HCOP:ITEM:TRAC:STAT [ON|OFF]`
Graticule ON off	`HCOP:ITEM:GRAT:STAT [ON|OFF]`
Annotation ON off	`HCOP:ITEM:ANN:STAT [ON|OFF]`
Mkr Symbol ON off	`HCOP:ITEM:MARK:STAT [ON|OFF]`
Title + Clk ON off	`HCOP:ITEM:TITL:STAT [ON|OFF]`

Table 10-16 MARKER Functions, 8712ET/ES and 8714ET/ES (1 of 3)

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>MARKER</td><td>(hardkey entry)</td></tr>
<tr><td>1: (enter value and units)</td><td><code>CALC[1|2]:MARK1 ON</code>
<code>CALC[1|2]:MARK1:X <num> [MHZ|KHZ|HZ]</code></td></tr>
<tr><td>2: (enter value and units)</td><td><code>CALC[1|2]:MARK2 ON</code>
<code>CALC[1|2]:MARK2:X <num> [MHZ|KHZ|HZ]</code></td></tr>
<tr><td>3: (enter value and units)</td><td><code>CALC[1|2]:MARK3 ON</code>
<code>CALC[1|2]:MARK3:X <num> [MHZ|KHZ|HZ]</code></td></tr>
<tr><td>4: (enter value and units)</td><td><code>CALC[1|2]:MARK4 ON</code>
<code>CALC[1|2]:MARK4:X <num> [MHZ|KHZ|HZ]</code></td></tr>
<tr><td>5: (enter value and units)</td><td><code>CALC[1|2]:MARK5 ON</code>
<code>CALC[1|2]:MARK5:X <num> [MHZ|KHZ|HZ]</code></td></tr>
<tr><td>6: (enter value and units)</td><td><code>CALC[1|2]:MARK6 ON</code>
<code>CALC[1|2]:MARK6:X <num> [MHZ|KHZ|HZ]</code></td></tr>
<tr><td>7: (enter value and units)</td><td><code>CALC[1|2]:MARK7 ON</code>
<code>CALC[1|2]:MARK7:X <num> [MHZ|KHZ|HZ]</code></td></tr>
<tr><td>8: (enter value and units)</td><td><code>CALC[1|2]:MARK8 ON</code>
<code>CALC[1|2]:MARK8:X <num> MHZ|KHZ|HZ]</code></td></tr>
<tr><td>Active Marker Off</td><td><code>CALC[1|2]:MARK[1|2| 8] OFF</code></td></tr>
<tr><td>All Off</td><td><code>CALC[1|2]:MARK:AOFF</code></td></tr>
</table>

Table 10-16 MARKER Functions, 8712ET/ES and 8714ET/ES (2 of 3)

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>MARKER, (continued)</td><td></td></tr>
<tr><td>Marker Functions</td><td>(menu selection only)</td></tr>
<tr><td>Delta Mkr on OFF</td><td>CALC[1|2]:MARK:MODE <REL|ABS></td></tr>
<tr><td>Marker -> Center</td><td>SENS[1|2]:FREQ:CENT
(CALC[1|2]:MARK[1|2|...8]:X:ABS?);*WAI</td></tr>
<tr><td>Marker -> Reference</td><td>DISP:WIND[1|2]:TRAC:Y:RLEV
(CALC[1|2]:MARK[1|2|...8]:Y?);*WAI</td></tr>
<tr><td>Marker -> Elec Delay</td><td>SENS[1|2]:CORR:EDEL:TIME
(CALC[1|2]:MARK[1|2|...8]:GDEL?);*WAI</td></tr>
<tr><td>Marker Math</td><td>(menu selection only)</td></tr>
<tr><td>Statistics</td><td>CALC[1|2]:MARK:FUNC STAT</td></tr>
<tr><td>Flatness</td><td>CALC[1|2]:MARK:FUNC FLAT</td></tr>
<tr><td>RF Filter Stats</td><td>CALC[1|2]:MARK:FUNC FST</td></tr>
<tr><td>Math Off</td><td>CALC[1|2]:MARK:FUNC OFF</td></tr>
<tr><td>Marker Search</td><td>(menu selection only)</td></tr>
<tr><td>Max Search</td><td>CALC[1|2]:MARK:FUNC MAX</td></tr>
<tr><td>Mkr -> Max</td><td>CALC[1|2]:MARK:FUNC MAX</td></tr>
<tr><td>Next Peak Left</td><td>CALC[1|2]:MARK:MAX:LEFT</td></tr>
<tr><td>Next Peak Right</td><td>CALC[1|2]:MARK:MAX:RIGH</td></tr>
<tr><td>Min Search</td><td>CALC[1|2]:MARK:FUNC MIN</td></tr>
<tr><td>Marker -> Min</td><td>CALC[1|2]:MARK:FUNC MIN</td></tr>
</table>

Table 10-16 MARKER Functions, 8712ES and 8714ES (3 of 3)

KEYSTROKES	SCPI COMMAND
MARKER, Marker Search, (continued)	
Min Search, (continued)	
Next Min Left	CALC[1|2]:MARK:MIN:LEFT
Next Min Right	CALC[1|2]:MARK:MIN:RIGH
Target Search	CALC[1|2]:MARK:FUNC TARG
Target Value (enter value, and units)	CALC[1|2]:MARK:TARG <LEFT|RIGH>,<num> [DB]
Search Left (enter value, and units)	CALC[1|2]:MARK:TARG LEFT,<num> [DB]
Search Right (enter value, and units)	CALC[1|2]:MARK:TARG RIGH,<num> [DB]
Bandwidth (enter value, and units)	CALC[1|2]:MARK:FUNC BWID CALC[1|2]:MARK:BWID <num> [DB]
Notch (enter value, and units)	CALC[1|2]:MARK:FUNC NOTC CALC[1|2]:MARK:NOTC <num> [DB]
More	(menu selection only)
Multi Peak	CALC[1|2]:MARK:FUNC MPE
MultiNotch	CALC[1|2]:MARK:FUNC MNOT
Search Off	CALC[1|2]:MARK:FUNC OFF
Tracking ON off	CALC[1|2]:MARK:FUNC:TRAC [ON|OFF]

Table 10-17 MEAS1 | MEAS2 Functions, 8712ES and 8714ES (1 of 2)

KEYSTROKES	SCPI COMMAND
MEAS1 | MEAS2	SENS[1|2]:STAT ON; *WAI
S11 Refl Port1	SENS[1|2]:FUNC 'XFR:S 1,1';DET NBAN; *WAI
S21 Fwd Trans	SENS[1|2]:FUNC 'XFR:S 2,1';DET NBAN; *WAI
S12 Rev Trans	SENS[1|2]:FUNC 'XFR:S 1,2';DET NBAN; *WAI
S22 Refl Port2	SENS[1|2]:FUNC 'XFR:S 2,2';DET NBAN; *WAI
Fault Location	SENS[1|2]:FUNC 'FLOC 1,0';DET NBAN; *WAI
SRL	SENS[1|2]:FUNC 'SRL 1,0';DET NBAN; *WAI
Power	SENS[1|2]:FUNC 'XFR:POW 2';DET BBAN; *WAI
Conversion Loss	SENS[1|2]:FUNC 'XFR:POW:RAT 2,0';DET BBAN; *WAI
Detection Options	(menu selection only)
Narrowband Internal	(menu selection only)
A	SENS[1|2]:FUNC 'XFR:POW 1';DET NBAN; *WAI
B	SENS[1|2]:FUNC 'XFR:POW 2';DET NBAN; *WAI
R	SENS[1|2]:FUNC 'XFR:POW 0';DET NBAN; *WAI
A/R	SENS[1|2]:FUNC 'XFR:S 1,1';DET NBAN; *WAI
B/R	SENS[1|2]:FUNC 'XFR:S 2,1';DET NBAN; *WAI
--> Path FWD rev	ROUT[1|2]:PATH:DEF:PORT [(2,1)|(1,2)]; *WAI

Table 10-17 MEAS1 | MEAS2 Functions, 8712ES and 8714ES (2 of 2)

KEYSTROKES	SCPI COMMAND
MEAS1 | MEAS2, (continued)	
Broadband Internal	(menu selection only
B*	`SENS[1|2]:FUNC 'XFR:POW 2';DET BBAN; *WAI`
R*	`SENS[1|2]:FUNC 'XFR:POW 0';DET BBAN; *WAI`
B*/R*	`SENS[1|2]:FUNC 'XFR:POW:RAT 2,0';DET BBAN; *WAI`
Broadband External	(menu selection only)
X	`SENS[1|2]:FUNC 'XFR:POW 11';DET BBAN; *WAI`
Y	`SENS[1|2]:FUNC 'XFR:POW 12';DET BBAN; *WAI`
X/Y	`SENS[1|2]:FUNC 'XFR:POW:RAT 11,12';DET BBAN; *WAI`
Y/X	`SENS[1|2]:FUNC 'XFR:POW:RAT 12,11';DET BBAN; *WAI`
Y/R*	`SENS[1|2]:FUNC 'XFR:POW:RAT 12,0';DET BBAN; *WAI`
Aux Input	`SENS[1|2]:FUNC 'XFR:VOLT'; *WAI`
--> Path FWD rev	`ROUT[1|2]:PATH:DEF:PORT [(2,1)|(1,2)]; *WAI`
Meas OFF	`SENS[1|2]:STAT OFF; *WAI`

See Table 10-19 on page 10-49 for Multiport Test Set.

Table 10-18 MEAS1 | MEAS2 Functions, 8712ET and 8714ET (1 of 2)

KEYSTROKES	SCPI COMMAND
MEAS1 | MEAS2, (continued)	
Transmissn	SENS1:FUNC 'XFR:S 2,1';DET NBAN; *WAI
Reflection	SENS1:FUNC 'XFR:S 1,1';DET NBAN; *WAI
Fault Location	SENS1:FUNC 'FLOC 1,0';DET NBAN; *WAI
SRL	SENS1:FUNC 'SRL 1,0';DET NBAN; *WAI
Power	SENS1:FUNC 'XFR:POW 2';DET BBAN; *WAI
Conversion Loss	SENS1:FUNC 'XFR:POW:RAT 2,0';DET BBAN; *WAI
Detection Options	(menu selection only)
Narrowband Internal	(menu selection only)
A	SENS[1|2]:FUNC 'XFR:POW 1';DET NBAN; *WAI
B	SENS[1|2]:FUNC 'XFR:POW 2';DET NBAN; *WAI
R	SENS[1|2]:FUNC 'XFR:POW 0';DET NBAN; *WAI
A/R	SENS[1|2]:FUNC 'XFR:S 1,1';DET NBAN; *WAI
B/R	SENS[1|2]:FUNC 'XFR:S 2,1';DET NBAN; *WAI

Table 10-18 MEAS1 | MEAS2 Functions, 8712ET and 8714ET (2 of 2)

KEYSTROKES	SCPI COMMAND
MEAS1 | MEAS2,(continued)	
Broadband Internal	(menu selection only)
B*	`SENS[1|2]:FUNC 'XFR:POW 2';DET BBAN; *WAI`
R*	`SENS[1|2]:FUNC 'XFR:POW 0';DET BBAN; *WAI`
B*/R*	`SENS[1|2]:FUNC 'XFR:POW:RAT 2,0';DET BBAN; *WAI`
Broadband External	(menu selection only)
X	`SENS[1|2]:FUNC 'XFR:POW 11';DET BBAN; *WAI`
Y	`SENS[1|2]:FUNC 'XFR:POW 12';DET BBAN; *WAI`
X/Y	`SENS[1|2]:FUNC 'XFR:POW:RAT 11,12';DET BBAN; *WAI`
Y/X	`SENS[1|2]:FUNC 'XFR:POW:RAT 12,11';DET BBAN; *WAI`
Y/R*	`SENS[1|2]:FUNC 'XFR:POW:RAT 12,0';DET BBAN; *WAI`
Aux Input	`SENS[1|2]:FUNC 'XFR:VOLT'; *WAI`
Meas OFF	`SENS[1|2]:STAT OFF; *WAI`
See Table 10-20 on page 10-49 for Multiport Test Set.	

Table 10-19 MEAS Multiport Test Set Functions, 8712ES and 8714ES

KEYSTROKES	SCPI COMMAND
MEAS1 | MEAS2, (Multiport)	
S11 ReflPort	SENS1:FUNC 'XFR:S 1,1';DET NBAN; *WAI
S21 Tran	SENS1:FUNC 'XFR:S 2,1';DET NBAN; *WAI
S12 Tran	SENS1:FUNC 'XFR:S 1,2';DET NBAN; *WAI
S22 ReflPort	SENS1:FUNC 'XFR:S 2,2';DET NBAN; *WAI
Multiport Selection	(menu selection only)
(S11) Port	(menu selection only)
(S22) Port	(menu selection only)

Table 10-20 MEAS Multiport Test Set Functions, 8712ET and 8714ET

KEYSTROKES	SCPI COMMAND
MEAS1 | MEAS2, (Multiport)	
Reflection	SENS1:FUNC 'XFR:S 1,1';DET NBAN; *WAI
Transmissn	SENS1:FUNC 'XFR:S 2,1';DET NBAN; *WAI
Multiport Selection	(menu selection only)
Reflection Port Num	ROUT[1|2]REFL:PATH:DEF:PORT[1|2|...|10]
Transmissn Port Num	ROUT[1|2]TRAN:PATH:DEF:PORT[1|2|...|10]

Table 10-21 MENU Functions, 8712ET/ES and 8714ET/ES (1 of 2)

KEYSTROKES	SCPI COMMAND
MENU	(hardkey entry)
Trigger	(menu selection only)
Continuous	`ABOR;:INIT1:CONT ON;*WAI`
Hold	`ABOR;:INIT1:CONT OFF;*WAI`
Single	`ABOR;:INIT1:CONT OFF;:INIT1;*WAI`
Trigger Source	(menu selection only)
Internal	`TRIG:SOUR IMM;:SENS:SWE:TRIG:SOUR IMM;*WAI`
External Sweep	`TRIG:SOUR EXT;:SENS:SWE:TRIG:SOUR IMM;*WAI`
External Point	`TRIG:SOUR EXT;:SENS:SWE:TRIG:SOUR EXT;*WAI`
Number of Points (enter value, press **Enter**)	`SENS[1|2]:SWE:POIN <num>; *WAI`
Distance[1]	(menu selection only)
Start Distance (enter value, press **Enter**)	`SENS[1|2]:DIST:STAR <num>; *WAI`
Stop Distance (enter value, press **Enter**)	`SENS[1|2]:DIST:STOP <num>; *WAI`
Feet	`SENS:DIST:UNIT FEET`
Meters	`SENS:DIST:UNIT MET`

1. Used with fault location measurements only.

Table 10-21 MENU Functions, 8712ET/ES and 8714ET/ES (2 of 2)

KEYSTROKES	SCPI COMMAND
MENU, (continued)	
SRL Cable Scan[1]	`SENS[1|2]:FUNC:SRL:SCAN; *WAI`
Ext Ref on OFF	`SENS:ROSC:SOUR [EXT|INT]; *WAI`
Spur Avoid Options	(menu selection only)
None	`DIAG:SPUR:METH NONE;*WAI`
Dither	`DIAG:SPUR:METH DITH;*WAI`
Spur Avoid	`DIAG:SPUR:METH AVO;*WAI`

1. Used with SRL measurements only. This softkey starts an automated SRL cable scan.

Table 10-22 POWER Functions, 8712ET/ES and 8714ET/ES (1 of 2)

KEYSTROKES	SCPI COMMAND
POWER	(hardkey entry)
Level[1] (enter value, and unit)	`SOUR[1|2]:POW <value>; *WAI`
dBm	(units choice)
dBW	(units choice)
dBuW	(units choice)
dBV	(units choice)
dBmV	(units choice)
dBuV	(units choice)
Cancel	(cancels entry)
RF ON off	`OUTP[ON|OFF]; *WAI`
Start Power[2] (enter value, and unit)	`SOUR:POW:STAR <value>; *WAI`
dBm	(units choice)
dBW	(units choice)
dBuW	(units choice)
dBV	(units choice)
dBmV	(units choice)
dBuV	(units choice)
Cancel	(cancels entry)

1. This softkey is valid for frequency sweeps only.
2. This softkey is valid for power sweeps only.

Table 10-22 POWER Functions, 8712ET/ES and 8714ET/ES (2 of 2)

KEYSTROKES	SCPI COMMAND
POWER, (continued)	
Stop Power[1] (enter value, and unit)	`SOUR:POW:STOP <value>; *WAI`
dBm	(units choice)
dBW	(units choice)
dBuW	(units choice)
dBV	(units choice)
dBmV	(units choice)
dBuV	(units choice)
Cancel	(cancels entry)
Pwr Level at Preset (enter value, and unit)	`SOUR:POW:PRESET<value>`
dBm	(units choice)
dBW	(units choice)
dBuW	(units choice)
dBV	(units choice)
dBmV	(units choice)
dBuV	(units choice)
Cancel	(cancels entry)

1. This softkey is valid for power sweeps only.

Table 10-23 MENU Functions, 8712ET/ES and 8714ET/ES

KEYSTROKES	SCPI COMMAND
PRESET	`SYST: Preset (hardkey entry)`
Factory Preset	(menu selection only)
User Preset	(menu selection only)

Table 10-24 SAVE RECALL Functions, 8712ET/ES and 8714ET/ES (1 of 5)

KEYSTROKES	SCPI COMMAND
SAVE RECALL	(hardkey entry)
Save State	MMEM:STOR:STAT 1,'MEM:STATE1.STA'
Re-Save State (enter file name, press **Enter**)	MMEM:STOR:STAT 1,'MEM:STATE1.STA'
Define Save	(menu selection only)
Inst State ON off	MMEM:STOR:STAT:IST [ON|OFF]
Cal on OFF	MMEM:STOR:STAT:CORR[ON|OFF].
Data on OFF	MMEM:STOR:STAT:TRAC [ON|OFF]
TSet Cal on OFF[1]	MMEM:STOR:STAT:TSCAL [ON|OFF]
File Format	(menu selection only)
8711A/B Compatible	MMEM:STOR:STAT:FORM B8711
8711C Compatible	MMEM:STOR:STAT:FORM C8711
8712/14E Compatible	MMEM:STOR:STAT:FORM E8711
Save ASCII	(menu selection only)
Lotus 123 Format	MMEM:STOR:TRAC:FORM LOT
Touchstone Format	MMEM:STOR:TRAC:FORM TOUC
Save Meas 1	MMEM:STOR:TRAC CH1FDATA, 'MEM:TRACE0.S1P'
Save Meas 2	MMEM:STOR:TRAC CH2FDATA, 'MEM:TRACE1.S1P'
Recall State	MMEM:LOAD:STAT 1,'MEM:STATE2.STA'
Programs	(menu selection only)
Save Program	(no SCPI command)
Re-Save Program	(no SCPI command)
File Type bin ASCII	(no SCPI command)

1. A multi-port test set is required for this softkey choice.

Table 10-24 SAVE RECALL Functions, 8712ET/ES and 8714ET/ES (2 of 5)

KEYSTROKES	SCPI COMMAND
SAVE RECALL, Programs, (continued)	
Recall Program	(no SCPI command)
Run	PROG:STAT:RUN
Continue	PROG:STAT:CONT
Step	(no SCPI command)
Edit	(no SCPI command)
Key record on OFF	(no SCPI command)
Utilities	(menu selection only)
Clear Program	(no SCPI command)
Stack Size	(no SCPI command)
Secure	(no SCPI command)
IBASIC DISPLAY	(menu selection only)
None	DISP:PROG OFF
Full	DISP:PROG FULL
Upper	DISP:PROG UPP
Lower	DISP:PROG LOW
Save AUTOST	(no SCPI command)
IBASIC	(menu selection only)
Run[1]	(no SCPI command)
Continue[1]	(no SCPI command)
Step[1]	(no SCPI command)
Edit[1]	(no SCPI command)

1. This action valid with key record off.

Table 10-24 SAVE RECALL Functions, 8712ET/ES and 8714ET/ES (3 of 5)

KEYSTROKES	SCPI COMMAND
SAVE RECALL, programs, (continued)	
Key Record on OFF	(no SCPI command)
Utilities[1]	(menu selection only)
Clear Program	(no SCPI command)
Stack Size	(no SCPI command)
Secure	(no SCPI command)
IBASIC Display	(menu selection only)
None	`DISP:PROG OFF`
Full	`DISP:PROG FULL`
Upper	`DISP:PROG UPP`
Lower	`DISP:PROG LOW`
Select Disk	(menu selection only)
Non-Vol RAM Disk	`MMEM:MSIS 'MEM:'`
Volatile RAM Disk	`MMEM:MSIS 'RAM:'`
Internal 3.5" Disk	`MMEM:MSIS 'INT:'`
NFS Device	(menu selection only)
Configure VOL_RAM	(menu selection only)
Restore Defaults	(no SCPI command)
Modify Size	(no SCPI command)
Current Size	(no SCPI command)
File Utilities	(menu selection only)
Rename File (edit name, press **Enter**)	`MMEM:MOVE '<loc>:<name>', '<loc>:<name>'`

Table 10-24 SAVE RECALL Functions, 8712ET/ES and 8714ET/ES (4 of 5)

<table>
<tr><th>KEYSTROKES</th><th>SCPI COMMAND</th></tr>
<tr><td>SAVE RECALL, File Utilities, (continued)</td><td></td></tr>
<tr><td>Delete File</td><td>MMEM:DEL '<loc>:<name>'</td></tr>
<tr><td>Delete All Files</td><td>MMEM:DEL 'MEM:*.*'</td></tr>
<tr><td>Copy File</td><td>MMEM:COPY '<loc>:<name>','<loc>:<name>'</td></tr>
<tr><td>Copy to NonVol RAM</td><td>MMEM:COPY ['MEM:NAME[1]'|'RAM:NAME[1]'
|'INT:NAME[1]'|'\NAME[1]'], 'MEM:NAME[1]'</td></tr>
<tr><td>Copy to Vol RAM</td><td>MMEM:COPY ['MEM:NAME[1]'|'RAM:NAME[1]'
|'INT:NAME[1]'|'\NAME[1]'], 'RAM:NAME[1]'</td></tr>
<tr><td>Copy to 3.5" Disk</td><td>MMEM:COPY ['MEM:NAME[1]'|'RAM:NAME[1]'
|'INT:NAME[1]'|'\NAME[1]'], 'INT:NAME[1]'</td></tr>
<tr><td>Copy to NFS Device</td><td>MMEM:COPY ['MEM:NAME[1]'|'RAM:NAME[1]'
|'INT:NAME[1]'|'\NAME[1]'], '\:NAME[1]'</td></tr>
<tr><td>Copy All Files</td><td>MMEM:COPY '*.*', '<loc>'</td></tr>
<tr><td>Copy to NonVol RAM</td><td>MMEM:COPY '*.*', 'MEM:NAME'[1]</td></tr>
<tr><td>Copy to Vol RAM</td><td>MMEM:COPY '*.*', 'RAM:NAME'[1]</td></tr>
<tr><td>Copy to 3.5" Disk</td><td>MMEM:COPY '*.*', 'INT:NAME'[1]</td></tr>
<tr><td>Copy to NFS Device</td><td>MMEM:COPY '*.*', '\ NAME'[1]</td></tr>
<tr><td>Format Disk Menu</td><td>(menu selection only)</td></tr>
<tr><td>Format NonVol RAM</td><td>MMEM:INIT 'MEM:', DOS</td></tr>
<tr><td>Format Vol RAM</td><td>MMEM:INIT 'RAM:', DOS</td></tr>
<tr><td>Format 3.5" Disk</td><td>MMEM:INIT 'INT:', DOS</td></tr>
</table>

1. The name can include directory, for example: 'MEM:\DIRNAME\NAME' .

Table 10-24 SAVE RECALL Functions, 8712ET/ES and 8714ET/ES (5 of 5)

KEYSTROKES	SCPI COMMAND
SAVE RECALL, File Utilities, (continued)	
Directory Utilities	(menu selection only)
Change Directory	`MMEM:CDIR '<name>'`
Make Directory	`MMEM:MDIR '<name>'`
Remove Directory	`MMEM:RDIR '<name>'`
Fast Recall on OFF	`DISP:MENU:RECALL:FAST [ON|OFF]`

Table 10-25 SCALE **Functions, 8712ET/ES and 8714ET/ES**

KEYSTROKES	SCPI COMMAND
SCALE	(hardkey entry)
Autoscale	`DISP:WIND[1|2]:TRAC:Y:AUTO ONCE`
Scale/Div (enter value, press **Enter**)	`DISP:WIND[1|2]:TRAC:Y:PDIV <num>`
Reference Level (enter value, press **Enter**)	`DISP:WIND1:TRAC:Y:RLEV <num>`
Reference Position (enter value, press **Enter**)	`DISP:WIND1:TRAC:Y:RPOS <num>`
Reference Tracking	(menu selection only)
Off	`DISP:WIND[1|2]:TRAC:Y:TRACK [ON|OFF]`
Track Peak	`DISP:WIND[1|2]:TRAC:Y:TRACK PEAK`
Track Frequency	`DISP:WIND[1|2]:TRAC:Y:TRACK FREQ`
Set Track Frequency (enter value and unit press **Enter)**	`DISP:WIND[1|2]:TRAC:Y:TRACK:FREQ <num>`
Phase Offset (enter value, press **Enter**)	`SENS[1|2]:CORR:OFFS:PHAS <num> DEG`
Electrical Delay (enter value and unit, press **Enter**)	`SENS[1|2]:CORR:EDEL:TIME <num> <unit>`

Table 10-26 SWEEP Functions, 8712ET/ES and 8714ET/ES

KEYSTROKES	SCPI COMMAND
SWEEP	(hardkey entry)
Sweep Time (enter value, and unit)	`SENS[1|2]:SWE:TIME <num>[s|ms]; *WAI`
Sweep Time AUTO man	`SENS[1|2]:SWE:TIME:AUTO [ON|OFF]; *WAI`
Alt Sweep on OFF	`SENS:COUP [NONE|ALL]; *WAI`
Step Sweep on OFF	`SENS[1|2]:SWE:GEN STEPPED; *WAI`
Frequency Sweep	`POWER:MODE FIXED; *WAI`
Power Sweep	`POWER:MODE SWEEP; *WAI`

Table 10-27 SYSTEM OPTIONS Functions, 8712ET/ES and 8714ET/ES (1 of 9)

KEYSTROKES	SCPI COMMAND
SYSTEM OPTIONS	(hardkey entry)
IBASIC	(menu selection only)
Run[1]	`PROG:STAT:RUN`
Continue[1]	`PROG:STAT:CONT`
Step[1]	(no SCPI command)
Edit[1]	(no SCPI command)
Key Record on OFF	(no SCPI command)
Utilities[1]	(menu selection only)
Clear Program	(no SCPI command)
Stack Size	(no SCPI command)
Secure	(no SCPI command)
IBASIC Display	(menu selection only)
None	`DISP:PROG OFF`
Full	`DISP:PROG FULL`
Upper	`DISP:PROG UPP`
Lower	`DISP:PROG LOW`
LAN	(menu selection only)
Login User Setup	(menu selection only)

1. Key record must be off to use this softkey.

Table 10-27 SYSTEM OPTIONS Functions, 8712ET/ES and 8714ET/ES (2 of 9)

KEYSTROKES	SCPI COMMAND
SYSTEM OPTIONS, Lan, (continued)	
Add Login User	`SYST:COMM:LAN:LOG:USER:ADD-? <STRING>,<STRING>`
User Name	`SYST:COMM:LAN:LOG:USER:LIST: NAME? <STRING>`
Password	(menu selection only)
Confirm Password	(menu selection only)
Delete Login User	`SYST:COMM:LAN:LOG:USER:DEL-? <STRING>,<STRING>`
User Name	`SYST:COMM:LAN:LOG:USER:LIST: NAME? <STRING>`
Password	(menu selection only)
Display User List	`SYST:COMM:LAN:LOGin:USER:LIST: COUN?`
Delete All Users	(menu selection only)
LAN Port Setup	(menu selection only)
[1] **8712/14ES IP Address** (enter address, press **Enter**)	`SYST:COMM:LAN:IPAD <IP address>`
Gateway IP Address (enter address, press **Enter**)	`SYST:COMM:LAN:ROUT:GAT <IP address>`
Subnet Mask (enter address, press **Enter**)	`SYST:COMM:LAN:ROUT:SMAS <IP address>`

1. The analyzer model number appears here. **8712ES IP Address** is shown as an example.

Table 10-27 SYSTEM OPTIONS Functions, 8712ET/ES and 8714ET/ES (3 of 9)

KEYSTROKES	SCPI COMMAND
SYSTEM OPTIONS, Lan, (continued)	
Diagnostic Utilities	(menu selection only)
IP Address to Ping (enter address, press **Enter**)	DIAG:COMM:LAN:PING:IPAD <IP address>
Perform Ping	DIAG:COMM:LAN:PING:IMM
Network stat Capture	DIAG:COMM:LAN:NETW:STAT
Ethernet Address	SYST:COMM:LAN:EADD?
NFS Device Setup	(menu selection only)
Mount NFS Device	SYST:COMM:LAN:NFS:MOUN-? <STRING>,<STRING>,<STRING>
Remote IP Addr/Host	SYST:COMM:LAN:NFS:MOUNT:LIST:REMH? <#1-7>, <[STRING]>
Remote Path	SYST:COMM:LAN:NFS:MOUNT:LIST:REMF? <#1-7>, <[STRING]>
Local Path	SYST:COMM:LAN:NFS:MOUNT:LIST:LOCF? <#1-7>, <[STRING]>
Automount At Powerup	SYST:COMM:LAN:NFS:AUTO:ADD-? <string>,<string>,<string>
Unmount NFS Device	SYST:COMM:LAN:NFS:UNMOUN-? <STRING>
Authentication	(menu selection only)
User ID	SYST:COMM:LAN:NFS:AUTH:ID: USER #0~4.74836e+07#
Group ID	SYST:COMM:LAN:NFS:AUTH:ID: GRO #0~4.74836e+07#

Table 10-27 SYSTEM OPTIONS Functions, 8712ET/ES and 8714ET/ES (4 of 9)

KEYSTROKES	SCPI COMMAND
SYSTEM OPTIONS, Lan, (continued)	
Automount Setup	(menu selection only)
Remote IP Addr/Host	SYST:COMM:LAN:NFS:AUTO:LIST: REMH? <#1-7>, [,STRING]>
Remote Path	SYST:COMM:LAN:NFS:MOUNT:LIST:REMF? <#1-7>, [,STRING]>
Local Path	SYST:COMM:LAN:NFS:MOUNT:LIST:LOCF? <#1-7>, [,STRING]>
Automount at Powerup	SYST:COMM:LAN:NFS:AUTO:ADD-? <string>,<string>,<string>
Remove Automount	SYST:COMM:LAN:NFS:AUTO:REM-? <string>
Automount Dev Table	SYST:COMM:LAN:NFS:AUT:LIST:COUN?
NFS Device Table	SYST:COMM:LAN:NFS:MOUN:LIST:COUN?
BOOTP Setup	(menu selection only)
BOOTP on OFF	SYST:COMM:LAN:BOOT:STAT <OFF|0|ON|1>
TFTP	SYST:COMM:LAN:BOOT:TRAN:METH <TFTP>
FTP	SYST:COMM:LAN:BOOT:TRAN:METH <FTP>
FTP User Name	SYST:COMM:LAN:BOOT:TRAN:FTP:USER <STRING>
FTP Password	SYST:COMM:LAN:BOOT:TRAN:FTP:PASS-? <STRING>
Timeout	SYST:COMM:LAN:BOOT:TIM #1~MAX_AUTO_CAL_TIME[S]#
Optional Boot Host	SYST:COMM:LAN:BOOT:HOST <STRING>

Table 10-27 SYSTEM OPTIONS Functions, 8712ET/ES and 8714ET/ES (5 of 9)

KEYSTROKES	SCPI COMMAND
SYSTEM OPTIONS, Lan (continued)	
Optional File Path	SYST:COMM:LAN:BOOT:TRAN:FILE: NAME <STRING>
Parameters Received	(menu selection only)
SICL LAN Setup	(menu selection only)
Restore Defaults	(no SCPI command)
GPIB Name	SYST:COMM:LAN:SICL:GPIB: NAME <STRING>
GPIB Log. Unit	SYST:COMM:LAN:SICL:GPIB: LU #0~1024#
GPIB Dev Address	SYST:COMM:LAN:SICL:GPIB: ADDR #0~20, \22~23#
SCPI Sock. Setup	(menu selection only)
Restore Defaults	(no SCPI command)
Socket Port No.	SYST:COMM:LAN:SCPI:SOCK:DATA:PORT: NUM #0~4.74836e+07#
GPIB	(menu selection only)
8712ES Address (enter address, press **Enter**)	SYST:COMM:GPIB:ADDR <address>
Talker Listener	SYST:COMM:GPIB:CONT OFF
System Controller	SYST:COMM:GPIB:CONT ON
GPIB Echo ON off	SYST:COMM:GPIB:ECHO [ON|OFF]
8712ES Address (enter address, press **Enter**)	SYST:COMM:GPIB:ADDR <address>

Table 10-28 SYSTEM OPTIONS Functions, 8712ET/ES and 8714ET/ES (6 of 9)

KEYSTROKES	SCPI COMMAND
SYSTEM OPTIONS, GPIB, (continued)	
Talker Listener	SYST:COMM:GPIB:CONT OFF
System Controller	SYST:COMM:GPIB:CONT ON
GPIB Echo ON off	SYST:COMM:GPIB:ECHO [ON|OFF]
System Config	(menu selection only)
Set Clock	(menu selection only)
Set Year (enter year, press **Enter**)	SYST:DATE <yyyy>,<m>,<d>
Set Month (enter month, press **Enter**)	SYST:DATE <yyyy>,<m>,<d>
Set Day (enter day, press **Enter**)	SYST:DATE <yyyy>,<m>,<d>
Set Hour (enter hour, press **Enter**)	SYST:TIME <h>,<m>,<s>
Set Minute (enter minute, press **Enter**)	SYST:TIME <h>,<m>,<s>
Round Seconds	(no SCPI command)
Clock Format	(menu selection only)
YYYY-MM-DD HH:MM	DISP:ANN:CLOC:DATE:FORM YMD
MM-DD-YYYY HH:MM	DISP:ANN:CLOC:DATE:FORM MDY
DD-MM-YYYY HH:MM	DISP:ANN:CLOC:DATE:FORM DMY
Numeric	DISP:ANN:CLOC:DATE:MODE NUM

Table 10-28 SYSTEM OPTIONS Functions, 8712ET/ES and 8714ET/ES (7 of 9)

KEYSTROKES	SCPI COMMAND
SYSTEM OPTIONS, Sys Config, (continued)	
Alpha	`DISP:ANN:CLOC:DATE:MODE ALPH`
Seconds ON off	`DISP:ANN:CLOC:SEC [ON|OFF]`
Beeper Volume (enter value, press **Enter**)	`SYST:BEEP:VOL <num>`
CRT Adjust	(menu selection only)
Restore Defaults	(no SCPI command)
Vertical Position	(no SCPI command)
Horizontal Position	(no SCPI command)
Sync Green on OFF	(no SCPI command)
Remove Pattern	(no SCPI command)
Restore Defaults	(no SCPI command)
Vertical Back Porch	(no SCPI command)
Vertical Frnt Porch	(no SCPI command)
Horizontal Back Porch	(no SCPI command)
Horizontal Frnt Porch	(no SCPI command)
Options Setup	(menu selection only)
Install Option (enter keyword, press **Enter**)	(no SCPI command)
Special Option (enter key, press **Enter**)	(no SCPI command)

Table 10-28 SYSTEM OPTIONS Functions, 8712ET/ES and 8714ET/ES (8 of 9)

KEYSTROKES	SCPI COMMAND
SYSTEM OPTIONS, Sys Config, (continued)	
User TTL Config	(menu selection only)
Default	`SYST:COMM:TTL:USER:FEED DEFAULT`
Softkey Auto-Step	`SYST:COMM:TTL:USER:FEED KEY`
Sweep Out	`SYST:COMM:TTL:USER:FEED SWEEP`
Switching Test Set	(menu selection only)
Multiport ON off[1]	`CONT1:MULT:STATE [ON|OFF]`
Special Test Set	(special test set use)
Operating Parameters (seven pages of parameters)	(menu selection only)
Hardcopy Screen	(no SCPI command)
Hardcopy All	(no SCPI command)
Abort	`HCOP:ABOR`
Next Screen	(screen selection only)
Previous Screen	(screen selection only)
Service	(menu selection only)
Tests and Adjustments	(menu selection only)
Select Self-Test	(no SCPI command)
Select Adjustment	(no SCPI command)
Execute Test	(no SCPI command)
Stop Test	(no SCPI command)
Instrument Info	(no SCPI command)

1. For use with multiport test sets only.

Table 10-28 SYSTEM OPTIONS Functions, 8712ET/ES and 8714ET/ES (9 of 9)

KEYSTROKES	SCPI COMMAND
SYSTEM OPTIONS, Service, (continued)	
Update Corr Const	(menu selection only)
Install CC from Disk	(no SCPI command)
Help Message	(no SCPI command)
Load CC from Disk	(no SCPI command)
Store CC to EPROM	(no SCPI command)
Store CC to Disk	(no SCPI command)
Update Corr Const	(menu selection only)
Install CC from Disk	(no SCPI command)
Store CC to Disk	(no SCPI command)

11 SCPI Command Summary

This chapter describes all *device* commands recognized by the analyzer. Example programs using these commands are given in the *Example Programs Guide*. IEEE 488.2 *common* commands are described in Chapter 9, "Introduction to SCPI."

Queries, Forms, and Parameter Types

All device commands have both command and query forms unless specified as command only or query only. To create the query form of a command, replace the command parameter with a "?". For example, the following command and parameter selects the log magnitude format (`MLOGarithmic`) for the data display:

```
CALCulate[1|2]:FORMat:MLOGarithmic
```

To find which format is active, use the corresponding query command:

```
CALCulate[1|2]:FORMat?
```

The analyzer returns the short form of the mnemonic for the active state or value. In this example, the string `MLOG` (the short-form of `MLOGarithmic`) is returned to the device that sent the query.

`<num>`, `<char>`, `<string>` and `<block>` refer to the parameter type expected by the instrument as part of the command.

Parameter Types

In the following tables, the **FORM** column gives the **parameter type** returned by the instrument in response to a query. NR1, NR2 and NR3 refer to the different types of numeric data. CHAR (character data), STRING (string data) and BLOCK (block data) are also used to describe response types. The parameter types expected by the instrument as part of a command are summarized below:

NR1	Integers (such as +1, 0, –1, 123, –12345)
NR2	Floating point numbers with an explicit decimal point (such as 12.3, +1.234, –0.12345)
NR3	Floating point numbers in scientific notation (such as +1.23E+5, +123.4E-3, -456.789E+6)
CHAR	Character parameters (sometimes referred to as discrete parameters) consisting of ASCII characters. They are typically used for program settings that have a finite number of values.
STRING	String parameters can contain virtually any set of ASCII characters. The string must begin with a single quote (') or a double quote (") and end with the same character.
BLOCK	Block parameters are typically used to transfer large quantities of related data (like a data trace).

Some numeric parameters may be followed by an appropriate suffix. Commands that accept a suffix also allow standard metric multipliers to be combined with the suffix. For example, commands that set a frequency will accept HZ, KHZ, MHZ and GHZ. Commands that set a time will accept S, MS, US, NS, PS, FS and AS. Note that case is ignored. The multiplier "M" is interpreted as either milli- (10^{-3}) or mega- (10^{6}), depending on context. If no suffix is included, the default units for the parameter are used.

SCPI Device Command Summary

NOTE

This SCPI command reference is also available online. It is stored inside your analyzer in electronic form. To use it, you must connect your instrument to the network, and access it using your Web browser. See the *The LAN Interface User's Guide Supplement* for details.

NOTE

Throughout this chapter, the following conventions are used:

- **Square brackets** ([ ]) are used to enclose a keyword that is optional or implied when programming the command; that is, the instrument will process the command to have the same effect whether the option node is omitted or not.
- **Parameter types** (< >) are distinguished by enclosing the type name in angle brackets.
- **A vertical bar** (|) can be read as "or" and is used to separate alternative parameter options.
- `UPPER`-case letters (as found in the command **`MODify`**, for example) are used to indicate the short form of a given mnemonic. The remaining lower-case letters are the rest of the long form mnemonic.

Table 11-1 ABORt

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`ABORt`	command only	Aborts and resets the sweep in progress.

Table 11-2 CALCulate (1 of 7)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>CALCulate[1|2]:DATA?[1]</td><td>query only BLOCK or NR3 [2]</td><td>Queries the formatted data trace — functionally equivalent to the command TRAC? CH<1|2>FDATA.</td></tr>
<tr><td>CALCulate[1|2]:FORMat <char></td><td>CHAR</td><td>Selects the display format for measurement data — choose from MLOGarithmic|MLINear|SWR or PHASe|SMITh|POLar|GDELay|REAL|IMAGinary|MIMPedance.</td></tr>
<tr><td>CALCulate[1|2]:FORMat :UNIT:MLIN <char></td><td>CHAR</td><td>Selects linear magnitude units for Y-axis display. Choose from W|MW|UW|V|MV|UV.</td></tr>
<tr><td>CALCulate[1|2]:FORMat :UNIT:MLOG <char></td><td>CHAR</td><td>Selects log magnitude units for Y-axis display. Choose from DBW|DBM|DBUW|DBV|DBMV|DBUV.</td></tr>
<tr><td>CALCulate[1|2] :GDAPerture:APERture <num></td><td>NR3</td><td>Sets the group delay aperture as a ratio of desired aperture / measured frequency span.</td></tr>
<tr><td>CALCulate[1|2] :GDAPerture:SPAN <num></td><td>NR3</td><td>Specifies the group delay aperture in Hertz.</td></tr>
</table>

1. Refer to Chapter 6, "Trace Data Transfers," and to the ASCDATA and REALDATA example programs in the *Example Programs Guide* for more information on this command.
2. The parameter type of the data is determined by the format selected — FORMat REAL uses BLOCK data, FORMat ASCii uses NR3 data separated by commas.

Table 11-2 CALCulate (2 of 7)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>CALCulate[1|2]:LIMit:DISPlay <ON|OFF> [1]</td><td>NR1</td><td>Turns display of limit lines on/off.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :FLATness:MAXimum <num></td><td>NR3</td><td>Sets the maximum value for a flatness limit test.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :FLATness:MINimum <num></td><td>NR3</td><td>Sets the minimum value for a flatness marker limit test.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :FLATness:STATe <ON|OFF>[1]</td><td>NR1</td><td>Turns flatness marker limit test on/off.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :FREQuency:MAXimum <num>[2]</td><td>NR3</td><td>Sets the maximum value for delta frequency marker limit test.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :FREQuency:MINimum <num>[2]</td><td>NR3</td><td>Sets the minimum value for delta frequency marker limit test.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :FREQuency [:STATe] <ON|OFF>[1]</td><td>NR1</td><td>Turns delta frequency marker limit testing on or off.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :STATistic:MEAN:MAXimum <num></td><td>NR3</td><td>Sets the maximum value for a statistic mean limit test.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :STATistic:MEAN:MINimum <num></td><td>NR3</td><td>Sets the minimum value for a statistic mean limit test.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :STATistic:MEAN:STATe <ON|OFF>[1]</td><td>NR1</td><td>Turn statistic mean marker limit test on/off.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :STATistic:PEAK:MAXimum <num></td><td>NR3</td><td>Sets the maximum value for a statistic peak-to-peak limit test.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:MARKer :STATistic:PEAK:MINimum <num></td><td>NR3</td><td>Sets the minimum value for a statistic peak-to-peak limit test.</td></tr>
</table>

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
2. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is assumed.

Table 11-2 CALCulate (3 of 7)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`CALCulate[1|2]:LIMit:MARKer :STATistic:PEAK:STATe <ON|OFF>`[1]	NR1	Turns statistic peak-to-peak marker limit test on/off.
`CALCulate[1|2]:LIMit:MARKer :TILT:MAXimum <num>`[2]	NR3	Sets the maximum value for delta amplitude marker limit test.
`CALCulate[1|2]:LIMit:MARKer :TILT:MINimum <num>`[2]	NR3	Sets the minimum value for delta amplitude marker limit test.
`CALCulate[1|2]:LIMit:MARKer :TILT [:STATe] <ON|OFF>`[1]	NR1	Turns delta amplitude marker limit testing on or off.
`CALCulate[1|2]:LIMit:SEGMent [1|2|12]:AMPLitude:STARt <num>`[2]	NR3	Sets the Begin Limit for the specified limit segment.
`CALCulate[1|2]:LIMit:SEGMent [1|2|12]:AMPLitude:STOP <num>`[2]	NR3	Sets the End Limit for the specified limit segment.
`CALCulate[1|2]:LIMit:SEGMent :AOFF`	command only	Turns off all limit segments for a given channel — deletes all segments in the channel's limit table.
`CALCulate[1|2]:LIMit:SEGMent [1|2|12]:DISTance:STARt <num>`[2]	NR3	Sets the Begin Distance for the specified limit segment. (Option 100 only)
`CALCulate[1|2]:LIMit:SEGMent [1|2|12]:DISTance:STOP <num>`[2]	NR3	Sets the End Distance for the specified limit segment. (Option 100 only)
`CALCulate[1|2]:LIMit:SEGMent [1|2|12]:FREQuency:STARt <num>`[2]	NR3	Sets the Begin Frequency for the specified limit segment.

1. Binary parameters accept the values of `1` (on) and `0` (off) in addition to `ON` and `OFF`.
2. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (`HZ` for frequency or `S` for time) is assumed.

Table 11-2 CALCulate (4 of 7)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>CALCulate[1|2]:LIMit:SEGMent [1|2|12]:FREQuency:STOP <num>[2]</td><td>NR3</td><td>Sets the End Frequency for the specified limit segment.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:SEGMent [1|2|12]:POWer:STARt <num>[2]</td><td>NR3</td><td>Sets the Begin Power for the specified limit segment.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:SEGMent [1|2|12]:POWer:STOP <num>[2]</td><td>NR3</td><td>Sets the End Power for the specified limit segment.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:SEGMent [1|2|12]:STATe <ON|OFF>[1]</td><td>NR1</td><td>Turns the specified limit segment on/off — adds or deletes the segment.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:SEGMent [1|2|12]:TYPE <char></td><td>CHAR</td><td>Sets the limit type for the specified segment, choose from LMAX|LMIN|PMAX|PMIN (Max Line, Min Line, Max Point, Min Point) — sets all of the segment's limit parameters to their default values.</td></tr>
<tr><td>CALCulate[1|2]:LIMit:STATe <ON|OFF>[1]</td><td>NR1</td><td>Turns the limit test on/off.</td></tr>
<tr><td>CALCulate[1|2]:MARKer:AOFF</td><td>command only</td><td>Turns off all markers for a given channel — this has the effect of turning off marker functions and tracking as well.</td></tr>
<tr><td>CALCulate[1|2]:MARKer:BWIDth <num>[2]</td><td>NR3</td><td>Calculates the bandwidth of a bandpass filter — num is the target bandwidth (–3 for the 3 dB bandwidth).</td></tr>
</table>

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
2. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is assumed.

Table 11-2 CALCulate (5 of 7)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>CALCulate[1|2]:MARKer
:FUNCtion:RESult?</td><td>query only
NR3
[,NR3,
NR3,
NR3]</td><td>Queries the results of the active marker function — MAX and MIN return the amplitude; TARG returns the frequency; BWID returns bandwidth, center frequency, Q and loss; STAT returns the frequency span, the mean and standard deviation of the amplitude response, and the peak-to-peak ripple; FLAT returns the frequency span, gain, slope and flatness; and FSTAT returns the insertion loss and peak-to-peak ripple of the passband of a filter, as well as the maximum signal amplitude in the stopband. Refer to the MARKERS example program in the Example Programs Guide for more information.</td></tr>
<tr><td>CALCulate[1|2]:MARKer
:FUNCtion[:SELect] <char></td><td>CHAR</td><td>Selects the active marker function — choose from
OFF|MAXimum|MINimum|TARGet
|BWIDth|NOTCh|MPEak|MNOTch
|STATistics|
FLATness|FSTATistics.</td></tr>
<tr><td>CALCulate[1|2]:MARKer
:FUNCtion:TRACking
<ON|OFF> [1]</td><td>NR1</td><td>Turn marker function tracking on/off.</td></tr>
<tr><td>CALCulate[1|2]:MARKer
[1|2|8]:GDELay?</td><td>query only</td><td>Returns the group delay value, in seconds, at the specified marker.</td></tr>
<tr><td>CALCulate[1|2]:MARKer
[1|2|8]:MAXimum</td><td>command only</td><td>Sets the specified marker to the maximum value on the trace.</td></tr>
<tr><td>CALCulate[1|2]:MARKer
[1|2|8]:MAXimum:LEFT</td><td>command only</td><td>Moves the specified marker to the next local maximum to the left.</td></tr>
</table>

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF .

Table 11-2 CALCulate (6 of 7)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>CALCulate[1|2]:MARKer [1|2|8]:MAXimum:RIGHt</td><td>command only</td><td>Moves the specified marker to the next local maximum to the right.</td></tr>
<tr><td>CALCulate[1|2]:MARKer [1|2|8]:MINimum</td><td>command only</td><td>Sets the specified marker to the minimum value on the trace.</td></tr>
<tr><td>CALCulate[1|2]:MARKer [1|2|8]:MINimum:LEFT</td><td>command only</td><td>Moves the specified marker to the next local minimum to the left.</td></tr>
<tr><td>CALCulate[1|2]:MARKer [1|2|8]:MINimum:RIGHt</td><td>command only</td><td>Moves the specified marker to the next local minimum to the right.</td></tr>
<tr><td>CALCulate[1|2]:MARKer :MODE <char></td><td>CHAR</td><td>Turns delta marker state on/off — choose ABSolute or RELative.</td></tr>
<tr><td>CALCulate[1|2]:MARKer :NOTCh <num>[3]</td><td>NR3</td><td>Calculates the notch width of a notch filter — num is the marker search level (–6 for the default 6 dB bandwidth).</td></tr>
<tr><td>CALCulate[1|2] :MARKer[1|2|8]:POINt[1]</td><td>NR3</td><td>Sets the specified marker point.</td></tr>
<tr><td>CALCulate[1|2] :MARKer:REFerence:X?</td><td>query only NR3</td><td>Queries the frequency of the reference marker.</td></tr>
<tr><td>CALCulate[1|2] :MARKer:REFerence:Y?</td><td>query only NR3</td><td>Queries the amplitude of the reference marker.</td></tr>
<tr><td>CALCulate[1|2] :MARKer[1|2|8][:STATe] <ON|OFF>[2]</td><td>NR1</td><td>Turns the specified marker on/off.</td></tr>
<tr><td>CALCulate[1|2] :MARKer[1|2|8]:TARGet <char>,<num>[3]</td><td>CHAR, NR3</td><td>Performs a marker search for a target value — char is the direction LEFT or RIGHt.</td></tr>
</table>

1. Refer to "Displaying Measurement Results" in Chapter 7 of the *User's Guide* for more information on using this command.
2. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
3. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is assumed.

Table 11-2 CALCulate (7 of 7)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`CALCulate[1|2]` `:MARKer[1|2|8]:X <num>`	NR3	Sets the specified marker frequency (or power if in power sweep).
`CALCulate[1|2]` `:MARKer[1|2|8]:X:ABS <num>`	NR?	Sets a marker to an absolute value (such as frequency or amplitude). The set value is not relative to a reference marker if one is enabled.
`CALCulate[1|2]` `:MARKer[1|2|8]:Y?`	query only NR3	Queries the specified marker amplitude.
`CALCulate[1|2]` `:MARKer[1|2|8]:Y:INDuctance?`	query only NR3	Queries the specified marker's inductance when in Smith chart format.
`CALCulate[1|2]` `:MARKer[1|2|8]:Y:MAGNitude?`	query only NR3	Queries the specified marker's magnitude when in polar format.
`CALCulate[1|2]` `:MARKer[1|2|8]:Y:PHASe?`	query only NR3	Queries the specified marker's phase value when in polar format.
`CALCulate[1|2]` `:MARKer[1|2|8]:Y:REACtance?`	query only NR3	Queries the specified marker's reactance value when in Smith chart format.
`CALCulate[1|2]` `:MARKer[1|2|8]:Y:RESistance?`	query only NR3	Queries the specified marker's resistance value when in Smith chart format.
`CALCulate[1|2]` `:MATH[:EXPRession] <expr>` [1]	EXPR	Selects a trace math expression — choose measurement trace from `(IMPL)` for "data only" or `(IMPL/CH<1|2>SMEM)` for "data / memory".

1. `<expr>` and EXPR represent expressions, a parameter type that consists of mathematical expressions that use character parameters and are enclosed in parentheses.

Table 11-3 CALibration

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
CALibration :SELF:ALL	command only	Initiates a SelfCal on all ports that were calibrated during the Test Set Cal. [1]
CALibration :SELF:METHod <ONEPort|TWOPort>	CHAR	Selects the method of SelfCal: enhanced response/1-port or 2-port.
CALibration :SELF <ON|OFF|ONCE> [2]	NR1 CHAR	Initiates a SelfCal on the currently selected ports and selects Periodic SelfCal (ON) or SelfCal Once (OFF or ONCE). [1]
CALibration :SELF:TIMER <num>	NR1	Sets the time interval for automatic SelfCals. [1]
CALibration :ZERO:AUTO <ON|OFF|ONCE>[2]	NR1	Turns the broadband detector autozeroing function on/off.

1. For use with multiport test sets only.
2. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.

Table 11-4 CONFigure

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`CONFigure <string>`	STRING	Configures the analyzer to measure a specific device type and parameter (the BEGIN function) — choose from one of the following strings: 'AMPLifier:TRANsmission' 'AMPLifier:REFLection' 'AMPLifier:TRANsmission:REVerse' [1]'AMPLifier:REFLection:REVerse' [1]'AMPLifier:POWer' 'FILTer:TRANsmission' 'FILTer:REFLection' 'BBANd:TRANsmission' 'BBANd:REFLection' 'BBANd:TRANsmission:REVerse' [1]'BBANd:REFLection:REVerse' [1]'MIXer:CLOSs' 'MIXer:GDEL' 'MIXer:REFLection' 'CABLe:TRANsmission' 'CABLe:REFLection' 'CABLe:FAULT' 'CABLe:SRL'

1. For use with 8712ES and 8714ES only.

Table 11-5 CONTrol

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
CONTrol[1|2]:MULTiport :STATE <ON|OFF>	NR1	When on, configures analyzer for use with a multiport test set. [1]

1. For use with multiport test sets only.

Table 11-6 DIAGnostic (1 of 4)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
DIAGnostic:CCONstants:INSTalled?	query only NR1	Queries if correction constants are installed in flash. Returns a 1 if true, and a 0 if false.
DIAGnostic:CCONstants:LOAD	command only	Loads default factory calibration constants from floppy disk to memory.
DIAGnostic:CCONstants:STORE:DISK	command only	Stores default factory calibration constants from memory to floppy disk.
DIAGnostic:CCONstants:STORE :EEPRom	command only	Stores default factory calibration constants from memory to flash EEPROM.
DIAGnostic:COMMunicate:LAN :NETWork:STAT	command only	Displays the menu to set the IP address to ping and to perform ping.
DIAGnostic:COMMunicate:LAN:PING :IMM	command only	"Pings" a remote user-specified IP address. Used in troubleshooting or verifying a LAN connection.
DIAGnostic:COMMunicate:LAN:PING :IPAD <string>	STRING	Sets the IP address to ping.

Table 11-6 DIAGnostic (2 of 4)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>DIAGnostic:COMMunicate:LAN:SEND <IP_address>,<port_num>,<string>,<timeout></td><td>NR1, STRING</td><td>Instructs the analyzer to open a socket to the specified IP address and port number, and send the string specified.

<timeout> is an integer <num> in the range 0-75 specifying the number of seconds allocated for a successful transmission. If 0 is specified, a minimum interval of 0.10 seconds is used. If <timeout> is not specified, the default interval of 75 seconds is used.

DIAGnostic:COMMunicate:LAN:SEND? returns 0 if the last socket connection was successful, and -1 if the last socket connection failed. This may not be the last socket connection made by the IBASIC program. Multiple socket connections are possible, and telnet sessions may be interspersed among programmed socket connections.</td></tr>
<tr><td>DIAGnostic:MDISplay[1|2]:CORRection <I_Direct|I_SRcmatch|I_TRacking|I_Response|I_LDmatch|I_ISolation|M_Direct|M_ISolation|M_SRcmatch|M_TRacking|M_Response|M_LDmatch|M_XScalar|C_Direct|C_SRcmatch|C_RTracking|C_LDmatch|C_TTracking|C_Isolation></td><td>command only</td><td>Return to measurement mode and auto scales after viewing calibration uncertainties.</td></tr>
</table>

Table 11-6 DIAGnostic (3 of 4)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
DIAGnostic:MDISplay[1|2] :CORRection <string>	command only	Displays corrected measurement uncertainties. Choose from one of the following strings: Cal check C_DIRECTivity C_LDMATCH C_ISOLATION C_RTRACKING C_SRCMATCH C_TTRACKING Interpolated Array (accessed through the service menu.) I_DIRECTivity I_RESPONSE I_SRCMATCH I_TRACKING I_LDMATCH I_ISOLATION Master Array (accessed through the service menu.) M_DIRECTivity M_RESPONSE M_SRCMATCH M_TRACKING M_XSCALAR M_LDMATCH I_ISOLATION
DIAGnostic:MDISplay[1|2] :RESTore	command only	Returns to measurement mode and autoscales after viewing calibration uncertainties.

Table 11-6 DIAGnostic (4 of 4)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
DIAGnostic:PORT:READ? <port><register> [1]	query only NR1, NR1	Reads the rear panel I/O ports.
DIAGnostic:PORT:WRITE <port><register> [2]	NR1, NR1, NR1	Writes to the rear panel I/O ports.
DIAGnostic:SNUMber <string>?	query only STRING	Queries the instrument's serial number.
DIAGnostic:SPUR:METHod <NONE|DITHer|AVOid>	NR1	Selects the spur avoid mode.

1. Refer to "Controlling Peripherals" in Chapter 7 of the *User's Guide* for more information on using this command. See also Table 11-7 on page 11-18 and Table 11-8 on page 11-19.
2. See "Controlling Peripherals" in Chapter 7 of the User's Guide for more information on using this command. See also Table 11-7 on page 11-18 and Table 11-8 on page 11-19.

Table 11-7 Writeable Ports

Port Number	Register	Description
15	0	Outputs 8-bit data to the Cent_D0 through D7 lines of the Centronics port. Cent_D0 is the least significant bit, Cent_D7 is the most significant bit. Checks Centronics status lines for: • Out of Paper • Printer Not on Line • BUSY • ACKNOWLEDGE
15	1	Sets/clears the user bit according to the least significant bit of A. A least significant bit equal to 1 sets the user bit high. A least significant bit of 0 clears the user bit.
15	2	Sets/clears the limit pass/fail bit according to the least significant bit of A. A least significant bit equal to 1 sets the pass/fail bit high. A least significant bit of 0 clears the pass/fail bit.
15	3	Outputs 8-bit data to the Cent_D0 through D7 lines of the Centronics port. Cent_D0 is the least significant bit, Cent_D7 is the most significant bit. Does not check Centronics status lines.
9	0	Outputs a byte to the serial port. The byte is output serially according to the configuration for the serial port.

NOTE

When using the `WRITEIO(15,0)` or `WRITEIO(15,3)` command, the Printer_Select Line is set high. However, when the instrument is doing hardcopy, the Printer_Select Line is set low. The Printer_Select line may or may not be used by individual printers. Check with your printer manual.

Table 11-8 Readable Ports

Port Number	Register	Description
9	0	Reads the serial port.
15	0	Reads the 8-bit data port Cent_D0 through D7.
15	1	Reads the user bit.
15	2	Reads the limit test pass/fail bit.
15	10	Reads the 8-bit status port. • D0 — Cent_acknowledge • D1 — Cent_busy • D2 — Cent_out_of_paper • D3 — Cent_on_line • D4 — Cent_printer_err

Table 11-9 DISPlay (1 of 7)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`DISPlay:ANNotation:CHANnel[1|2][:STATe] <OFF|ON>` [1]	NR1	Enables/disables measurement channel annotation.
`DISPlay:ANNotation :CHANnel[1|2]:USER:LABel:DATA <string>` [2]	STRING	Specifies the string to be displayed in the measurement channel annotation area (above the graticule).
`DISPlay:ANNotation :CHANnel[1|2]:USER:STATe<OFF|ON>` [1,2]	NR1	Enables user-defined measurement channel annotation.
`DISPlay:ANNotation:CLOCk :DATE:FORMat <char>`	CHAR	Selects the Year/Month/Day ordering of the date in the clock display — choose from `YMD|MDY|DMY`.
`DISPlay:ANNotation:CLOCk :DATE:MODE <char>`	CHAR	Selects the format for the date in the clock display — choose `NUMeric` or `ALPHa`.
`DISPlay:ANNotation:CLOCk :MODE <char>`	CHAR	Selects how the clock will appear in the measurement display title area — choose from `LINE1|LINE2|OFF`.
`DISPlay:ANNotation:CLOCk :SEConds[:STATe]<ON|OFF>` [1]	NR1	Turns on/off display of seconds in the clock display.
`DISPlay:ANNotation :FREQuency[1|2]:MODE <char>`	CHAR	Sets the frequency annotation on the display — choose `SSTOP` (start/stop), `CSPAN` (center/span) or `CW`.
`DISPlay:ANNotation :FREQuency[1|2]:RESolution <char>`	CHAR	Sets the resolution of display frequency values — choose from `MHZ|KHZ|HZ`.
`DISPlay:ANNotation:FREQuency [1|2] [:STATe] <OFF|ON>` [1]	NR1	Enables/disables frequency annotation.
`DISPlay:ANNotation:FREQuency [1|2]:USER:LABel:DATA<string>`	STRING	A user-defined X-axis label.

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
2. Refer to "Displaying Measurement Results" in Chapter 7 of the *User's Guide* for more information on using this command.

Table 11-9 DISPlay (2 of 7)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>DISPlay:ANNotation:FREQuency [1|2]:USER:STARt<num>[2]</td><td>NR3</td><td>Specifies the start value for user-defined frequency annotation.</td></tr>
<tr><td>DISPlay:ANNotation:FREQuency [1|2]:USER:STATe[OFF|ON][1,2]</td><td>NR1</td><td>Enables user-defined frequency annotation.</td></tr>
<tr><td>DISPlay:ANNotation:FREQuency [1|2]:USER:STOP<num>[2]</td><td>NR3</td><td>Specifies the stop value for user-defined frequency annotation.</td></tr>
<tr><td>DISPlay:ANNotation:FREQuency [1|2]:USER:SUFFix[:DATA] <string>[2]</td><td>STRING</td><td>Specifies the suffix for user defined frequency annotation.</td></tr>
<tr><td>DISPlay:ANNotation:LIMit:ICON [1|2]:FLAG[:STATe] <ON|OFF>[1]</td><td>NR1</td><td>Enables/disables the display of the limit test fail icon.</td></tr>
<tr><td>DISPlay:ANNotation:LIMit:ICON [1|2] :POSition:X <num></td><td>NR1</td><td>Positions the limit test pass/fail text and icon on the display. Accepts whole number values from 0 (flush left) to 100 (flush right).</td></tr>
<tr><td>DISPlay:ANNotation:LIMit:ICON [1|2] :POSition:Y <num></td><td>NR1</td><td>Positions the limit test pass/fail text and icon on the display. Accepts whole number values from 0 (bottom of display) to 100 (top of display).</td></tr>
<tr><td>DISPlay:ANNotation:LIMit:ICON [1|2] :TEXT[:STATe] <ON|OFF>[1]</td><td>NR1</td><td>Turns the limit test "PASS/FAIL" text on or off.</td></tr>
<tr><td>DISPlay:ANNotation:MARKer[1|2] :NUMBers [:STATe] <OFF|ON>[1]</td><td>NR1</td><td>Enables/disables the display of marker numbers on trace markers.</td></tr>
<tr><td>DISPlay:ANNotation:MARKer[1|2] [:STATe]<ON|OFF>[1]</td><td>NR1</td><td>Enables/disables the active marker annotation for measurement channels 1 and 2.</td></tr>
</table>

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
2. Refer to “Displaying Measurement Results” in Chapter 7 of the *User’s Guide* for more information on using this command.

Table 11-9 DISPlay (3 of 7)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
DISPlay:ANNotation:MESSage:AOFF	command only	Turns off any currently showing message window — includes message window, active entry and IBASIC window.
DISPlay:ANNotation:MESSage :CLEar [1]	command only	Removes a user-defined pop-up message window.
DISPlay:ANNotation:MESSage :DATA <string> [1]	STRING	Displays a user-defined message in the pop-up message window. Optional argument specifies the timeout: choose from NONE | SHORt | MEDium | LONG.
DISPlay:ANNotation:MESSage :STATe <ON|OFF>[1]	NR1	Enables/disables the message window — CAUTION: this suppresses display of all messages (even ERROR messages).
DISPlay:ANNotation:TITLe[1|2] :DATA <string>[1]	STRING	Enters a string for the specified title line.
DISPlay:ANNotation:TITLe[:STATe] <ON|OFF>[1]	NR1	Turns on/off display of the title and clock.
DISPlay:ANNotation:YAXis:MODE <char>	CHAR	Sets mode for the Y-axis labels — choose RELative or ABSolute
DISPlay:ANNotation:YAXis[:STATe] <ON|OFF> [2]	NR1	Turns on/off Y-axis labels.
DISPlay:CMAP:COLor[1|2|...16] :GREYscale <num>	NR2	Changes the default intensity of the selected item on the analyzer's internal monitor.

1. Refer to "Operator Interaction" in Chapter 7 of the *User's Guide* for more information on using this command.
2. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.

Table 11-9 **DISPlay (4 of 7)**

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`DISPlay:CMAP:COLor[1|2|...16]` `:HSL <num>,<num>,<num>`	NR2	For use with an external VGA compatible monitor. Sets hue, saturation, and luminance for the selected display item. Accepted values for each parameter are 0 to 1.
`DISPlay:CMAP:COLor[1|2|...16]` `:RGB <num,num,num>`	NR2	For use with an external monitor. Sets the color map based on the Red/Green/Blue model. Accepted values for each parameter are 0 to 1.
`DISPlay:CMAP:DEFault`	command only	For use with an external monitor. Sets the color scheme to the factory default.
`DISPlay:CMAP:SCHeme <char>`	CHAR	Sets the color scheme for an external monitor. Choose from `DEFault|DEFault2|GREY|INVerse|CUSTom`.
`DISPlay:FORMat <char>`	CHAR	Selects the format (full or split screen) for displaying trace data — choose `SINGle` (overlay) or `ULOWer` (split).
`DISPlay:FORMat:EXPAND` `<ON|OFF>`	NR1	Enables/disables expand measurement mode.
`DISPlay:MENU:KEY[1|2|7]` `<string>` [1]	STRING	Specifies the softkey menu labels from a remote controller or IBASIC
`DISPlay:MENU[2]:KEY[1|2|7]` `<string>` [1]	STRING	Specifies the softkey menu labels when using user-defined BEGIN key.
`DISPlay:MENU:RECall:FAST` `[:STATe] <ON|OFF>` [2]	NR1	Turns on/off fast recall mode.

1. Refer to "Operator Interaction" in Chapter 7 of the *User's Guide* for more information on using this command.
2. Binary parameters accept the values of `1` (on) and `0` (off) in addition to `ON` and `OFF`.

Table 11-9 DISPlay (5 of 7)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
DISPlay:PROGram[:MODE] <char>	CHAR	Selects the portion of the analyzer's screen to be used as an HP Instrument BASIC display — choose from OFF|FULL|UPPer|LOWer.
DISPlay:WINDow[1|2|10] :GEOMetry:LLEFT?	query only NR1, NR1	Queries the absolute pixel coordinates of the lower left corner of the selected display window.
DISPlay:WINDow[1|2|10] :GEOMetry:SIZE?	query only NR1, NR1	Queries the width and height (in pixels) of the selected display window.
DISPlay:WINDow[1|2|10] :GEOMetry:URIGHT?	query only NR1,NR1	Queries the absolute pixel coordinates of the upper right corner of the selected display window.
DISPlay:WINDow:GRAPhics :BUFFer[:STATe]<ON|OFF>[1]	NR1	Turn on/off buffering of user graphics commands.
DISPlay:WINDow[1|2|10] :GRAPhics :CIRCle <num>[2]	command only	Draws a circle of the specified Y-axis radius centered at the current pen location — num is the radius in pixels. [3]
DISPlay:WINDow[1|2|10] :GRAPhics :CLEar[3]	command only	Clears the user graphics and graphics buffer for the specified window.
DISPlay:WINDow[1|2|10] :GRAPhics :COLor <num>[3]	NR1	Sets the color of the user graphics pen — choose from 0 for erase, 1 for bright, and 2 for dim.
DISPlay:WINDow[1|2|10] :GRAPhics [:DRAW] <num1>,<num2>[3]	command only	Draws a line from the current pen position to the specified new pen position — num1 and num2 are the new absolute X and Y coordinates in pixels. [3]

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
2. Refer to Chapter 7, "Using Graphics," for more information.
3. Refer to Chapter 7, and to the example program titled "GRAPHICS" in the *Example Programs Guide* for more information.

Table 11-9 DISPlay (6 of 7)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>DISPlay:WINDow[1|2|10]
:GRAPhics:LABel <string>[1]</td><td>command only</td><td>Draws a label with the lower left corner at the current pen location.</td></tr>
<tr><td>DISPlay:WINDow[1|2|10]
:GRAPhics:LABel:FONT <char>[1]</td><td>CHAR</td><td>Selects the user graphics label font — choose from SMALl|HSMall|NORMal|HNORmal|BOLD|HBOLd|SLANt|HSLant.</td></tr>
<tr><td>DISPlay:WINDow[1|2|10]
:GRAPhics :MOVE <num1>,<num2>[1]</td><td>NR1,NR1</td><td>Moves the pen to the specified new pen position — num1 and num2 are the new absolute X and Y coordinates in pixels. [1]</td></tr>
<tr><td>DISPlay:WINDow[1|2|10]
:GRAPhics:RECTangle
<num1>,<num2></td><td>command only</td><td>Draws a rectangle of the specified size with lower left corner at the current pen position — num1 and num2 are the width and height in pixels. [1]</td></tr>
<tr><td>DISPlay:WINDow[1|2|10]
:GRAPhics:SCALe
<xmin>,<xmax>,<ymin,<ymax></td><td>NR1</td><td>Specifies new coordinates for window.</td></tr>
<tr><td>DISPlay:WINDow[1|2|10]
:GRAPhics :STATe?</td><td>query onlyNR1</td><td>Queries whether a window is enabled for user graphics commands.</td></tr>
<tr><td>DISPlay:WINDow[1|2]:TRACe
:GRATicule:GRID[:STATe]<ON|OF
F>[2]</td><td>NR1</td><td>Turns display graticule on/off.</td></tr>
<tr><td>DISPlay:WINDow[1|2]
:TRACe[1|2][:STATe]<ON|OFF>[2]</td><td>NR1</td><td>Turns the display of trace and memory data from the specified measurement channel on/off.</td></tr>
<tr><td>DISPlay:WINDow[1|2]:TRACe
:Y[:SCALe]:AUTO ONCE</td><td>command only</td><td>Scales the measurement data for a best fit display.</td></tr>
</table>

1. Refer to “Using Graphics” in Chapter 7 and the “GRAPHICS” program in the *Example Programs Guide* for more information on using this command.
2. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.

Table 11-9 DISPlay (7 of 7)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>DISPlay:WINDow[1|2]:TRACe
:Y[:SCALe]:PDIVision<num>[1]</td><td>NR3</td><td>Specifies the height (dB or units per division) of each vertical division of the specified measurement channel.</td></tr>
<tr><td>DISPlay:WINDow[1|2]:TRACe
:Y[:SCALe]:RLEVel<num>[1]</td><td>NR3</td><td>Specifies the value for the Y-axis reference position for the specified measurement channel.</td></tr>
<tr><td>DISPlay:WINDow[1|2]:TRACe
:Y[:SCALe]:RPOSition <num>[1]</td><td>NR3</td><td>Specifies the Y-axis reference position for the specified measurement channel.</td></tr>
<tr><td>DISPlay:WINDow[1|2|10]
:TRACe[1|2]:Y:TRACk
<OFF|PEAK|FREQ></td><td>CHAR</td><td>Selects the method for reference offset tracking.</td></tr>
<tr><td>DISPlay:WINDow[1|2|10]
:TRACe[1|2]:Y:TRACk:FREQuency
<num>[1]</td><td>NR3</td><td>Selects frequency to track with reference tracking.</td></tr>
</table>

1. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is assumed.

Table 11-10 FORMat

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`FORMat:BORDer <char>`	CHAR	Specifiesa the byte order used for GPIB data transfer — choose `NORMal` or `SWAPped` (for PC-compatible systems).
`FORMat[:DATA] <char>[,<num>]`	CHAR [,NR1]	Specifies the data format for use during data transfer — choose from `REAL,64|REAL,32|INTeger,16 |ASCii`.

Table 11-11 HCOPy (1 of 2)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>HCOPy:ABORt</td><td>command only</td><td>Aborts any hardcopy currently in progress.</td></tr>
<tr><td>HCOPy:DEVice[1|2|3]:COLor <ON|OFF> [1, 2]</td><td>NR1</td><td>Selects monochrome OFF or color ON mode for hardcopy output.</td></tr>
<tr><td>HCOPy:DEVice[1|2|3]:LANGuage <char>[2]</td><td>CHAR</td><td>Selects the language for hardcopy output — choose from PCL|HPGL|EPSon|IBM|PCX|PCL5[3]</td></tr>
<tr><td>HCOPy:DEVice[1|2|3]:MODE <char>[2]</td><td>CHAR</td><td>Selects the graph and/or table(s) to appear on a hardcopy plot — choose from GMARker|GRAPh|ISETtings|MARKer|TABLe.</td></tr>
<tr><td>HCOPy:DEVice[1|2|3]:PAGE:MARGin :LEFT <num>[2]</td><td>NR2</td><td>Sets the left margin (for printer output) in millimeters.</td></tr>
<tr><td>HCOPy:DEVice[1|2|3]:PAGE:MARGin :TOP <num>[2]</td><td>NR2</td><td>Sets the top margin (for printer output) in millimeters.</td></tr>
<tr><td>HCOPy:DEVice[1|2|3]:PAGE :ORIentation <char>[2]</td><td>CHAR</td><td>Sets printer output page orientation — choose PORTrait or LANDscape.</td></tr>
<tr><td>HCOPy:DEVice[1|2|3]:PAGE:WIDTh <num>[2]</td><td>NR2</td><td>Sets the print width (for printer output) in millimeters.</td></tr>
<tr><td>HCOPy:DEVice[1|2|3]:PORT <char>[2]</td><td>CHAR</td><td>Selects the communications port for hardcopy output — choose from CENTronics|SERial|GPIB|MMEMory|LAN.</td></tr>
</table>

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
2. For DEVice, use 1 for PCL/Epson printers, 2 for plotters, and 3 for PCL5 printers.
3. EPSon and IBM produce the same results.

Table 11-11 HCOPy (2 of 2)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>HCOPy:DEVice[1|2]:RESolution <num>[1]</td><td>NR1</td><td>Sets the printer resolution in dots per inch.</td></tr>
<tr><td>HCOPy[:IMMediate]</td><td>command only</td><td>Initiates a hardcopy output (print or plot).
For DEVice, use 1 for PCL/Epson printers, or 2 for plotters.</td></tr>
<tr><td>HCOPy:ITEM[1|2|3]:ANNotation :STATe <ON|OFF>[1, 2]</td><td>NR1</td><td>Turns on/off channel and frequency annotation as part of hardcopy output.</td></tr>
<tr><td>HCOPy:ITEM[1|2|3]:FFEed:STATe <ON|OFF>[1, 2]</td><td>NR1</td><td>Turns on/off an automatic form feed at the completion of hardcopy output — use item 1 for printers and 2 for plotters.</td></tr>
<tr><td>HCOPy:ITEM[1|2|3]:GRATicule :STATe <ON|OFF>[1, 2]</td><td>NR1</td><td>Turns on/off graticule as part of hardcopy output.</td></tr>
<tr><td>HCOPy:ITEM[1|2|3]:MARKer:STATe <ON|OFF>[2, 3]</td><td>NR1</td><td>Turns on/off marker symbols as part of hardcopy output.</td></tr>
<tr><td>HCOPy:ITEM[1|2|3]:TITLe:STATe <ON|OFF>[2, 3]</td><td>NR1</td><td>Turns on/off title and clock lines as part of hardcopy output.</td></tr>
<tr><td>HCOPy:ITEM[1|2|3]:TRACe:STATe <ON|OFF>[2, 3]</td><td>NR1</td><td>Turns on/off trace data as part of hardcopy output.</td></tr>
</table>

1. For DEVice, use 1 for PCL/Epson printers, or 2 for plotters.
2. For DEVice, use 1 for PCL/Epson printers, 2 for plotters, and 3 for PCL5 printers.
3. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.

Table 11-12 INITiate

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`INITiate[1|2]:CONTinuous <ON|OFF>`[1]	NR1	Sets the trigger system to continuously sweep or to stop sweeping.
`INITiate[1|2][:IMMediate]`	command only	Initiates a new measurement sweep.

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.

Table 11-13 INPut

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`INPut:GAIN:AUTo <R|A|B>,<ON|OFF>`	command only	Sets the R, A, or B IF input automatic gain control on or off.
`INPut:GAIN:SETTing <R|A|B>, <HIGH|MEDHIGH|MEDIUM|LOW>`	command only	Sets the R, A, or B IF input gain to one of four choices: high, medium-high, medium, or low.

Table 11-14 MMEMory (1 of 3)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
MMEMory:CATalog?<string>[1]	query only STRING	Lists the names of the files in memory. (a directory listing, file names only.)
MMEMory:CDIRectory <string>	STRING	Changes the current directory on a DOS formatted disk — new directory must be on the same mass storage device.
MMEMory:COPY <string1>, <string2>[1, 2]	command only	Copies a file — string1 is the source file, string2 is the destination file.
MMEMory:DELete <string>[1, 2]	command only	Deletes a file — string is the filename.
MMEMory:INITialize [<string>[,<char>[,<num>]]]	command only	Formats a disk — string is the mass storage device MEM: (internal memory), or INT: (internal floppy disk). Disk format char is DOS, and the interleave factor num.
MMEMory:LOAD:STATe 1, <string>[1, 3]	command only	Recalls an instrument state from mass storage — string is the filename.
MMEMory:LOAD:CKIT:USER[:SELect] <KIT1|KIT2...KIT10>	command only	Recalls a user cal kit.
MMEMory:FILE:INFO?<string> [1]	query only STRING	Returns file information such as date/time. (a directory listing, file names and more info.)
MMEMory:MDIRectory <string> [2]	command only	Makes a new directory on a DOS formatted disk.

1. Filenames may include the mass storage device — MEM: (internal non-volatile memory), RAM: (internal volatile memory), INT: (internal 3.5 disk drive) or *NFS local path*. Wildcards ? and * may be used.
2. Be sure to catalog the desired disk using MMEM:MSIS before using this command.
3. Refer to "Automated Measurement Setup and Control" in Chapter 7 of the *User's Guide* for more information on using this command.

Table 11-14 MMEMory (2 of 3)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
MMEMory:MOVE <string1>,<string2>[1, 2]	command only	Moves or renames a file — string1 is the source (or old) filename and string2 is the destination (or new) filename.
MMEMory:MSIS <string>	STRING	Selects a mass storage device — choose MEM: (internal memory), INT: (internal floppy disk drive), etc.
MMEMory:RDIRectory <string>[2]	command only	Deletes a directory from a DOS formatted disk.
MMEMory:STORe:STATe 1, <string>[1, 2, 3]	command only	Saves an instrument state to mass storage — string is the filename.
MMEMory:STORe:STATe:CORRection <ON|OFF>[3]	NR1	Turns the calibration on/off — part of the definition of a saved file.
MMEMory:STORe:STATe:FORMat <char>	CHAR	Saves instrument state files to be compatible with older "A/B" model analyzers (choose B8711), or with current "C" model analyzers (choose C8711).
MMEMory:STORe:STATe:ISTate <ON|OFF>[3]	NR1	Turns the instrument state on/off — part of the definition of a saved file.
MMEMory:STORe:STATe:TRACe <ON|OFF>[3]	NR1	Turns the data trace on/off — part of the definition of a saved file.

1. File names may include the mass storage device name—MEM: (internal non-volatile memory), RAM:(internal volatile memory), INT: (internal 3.5" disk drive), or *NFS local path*. Wildcards ? and * may be used.
2. Be sure to catalog the desired disk using MMEM:MSIS before using this command.
3. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.

Table 11-14 MMEMory (3 of 3)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>MMEMory:STORe:STATe:TSCAL <ON|OFF>[1]</td><td>NR1</td><td>When on, the saved state will be the test set cal only.</td></tr>
<tr><td>MMEMory:STORe:TRACe <char>,<string>[2, 3]</td><td>command only</td><td>Stores an ASCII list of trace and frequency values to a file — char is the formatted data trace CH<1|2>FDATA and string is the filename.</td></tr>
<tr><td>MMEMory:STORe:TRACe:FORMat <char></td><td>CHAR</td><td>Selects the format that the ASCII data will be saved in. Choose from LOTus 123 or TOUChstone.</td></tr>
<tr><td>MMEMory:TRANsfer:BDAT <string>[2][,<block>][4]</td><td>STRING, BLOCK</td><td>Copies a file to or from the analyzer's disk drive.[5]</td></tr>
<tr><td>MMEMory:TRANsfer[:HFS] <string>[2][,<block>][4]</td><td>STRING,</td><td>Copies a file to or from the analyzer's disk drive.[5]</td></tr>
</table>

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
2. File names may include the mass storage device name—MEM:(internal non-volatile memory), RAM: (internal volatile memory), INT: (internal 3.5” disk drive), or *NFS local path*. Wildcards ? and * may be used.
3. Refer to “Automated Measurement and Control” in Chapter 7 of the *User's Guide* for more information on using this command
4. Refer to the *Example Programs Guide* for more information on using this command.
5. Refer to the example programs PUTFILE and GETFILE in the *Example Programs Guide*.

Table 11-15 OUTPut

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td><code>OUTPut[:STATe] <ON|OFF></code> [1]</td><td>NR1</td><td>Turns RF power from the source on/off.</td></tr>
</table>

1. Binary parameters accept the values of `1` (on) and `0` (off) in addition to `ON` and `OFF`.

Table 11-16 POWer

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td><code>POWer[1|2]:MODE <char></code></td><td>CHAR</td><td>Specifies either frequency sweep (<code>FIXed</code>) or power sweep (<code>SWEep</code>).</td></tr>
</table>

Table 11-17 PROGram (1 of 2)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
PROGram[1] :CATalog?	query only STRING	Lists the names of the defined IBASIC programs — response is "PROG" (if a program is present) or the null string ("").
PROGram[1][:SELected][2] :DEFine <block>	BLOCK	Downloads an IBASIC program from an external controller.
PROGram[1][:SELected][2]:DELete:ALL	command only	Deletes all IBASIC programs from the program buffer — equivalent to an HP BASIC SCRATCH A command.
PROGram[1][:SELected][2] :DELete[:SELected]	command only	Deletes the active IBASIC program — equivalent to an HP BASIC SCRATCH A command.
PROGram[1][:SELected][2]:EXECute <string>	command only	Executes an IBASIC command.
PROGram[1][:SELected][2] :MALLocate <num>	NR1	Allocates memory space for IBASIC programs — choose an integer between 2048 and 4000000 bytes.

1. IBASIC programs can be generated and controlled in the instrument.
2. Commands grouped under the SELected mnemonic in the PROGram subsystem operate on the active program buffer.

Table 11-17 PROGram (2 of 2)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`PROGram [:SELected][1]:NAME 'PROG'`	STRING	Selects the IBASIC program in the program buffer to be active.
`PROGram [:SELected][1] :NUMBer <string>,<data>` [2]	BLOCK or NR3 [2]	Loads a new value for a numeric variable `string` in the active IBASIC program — `num` is the new value.
`PROGram [:SELected][1] :STATe <char>`	CHAR	Selects the state of the active IBASIC program — choose from `STOP|PAUSe|RUN|CONTinue`.
`PROGram [:SELected][1]:STRing <string1>,<string2>`	STRING	Loads a new value for a string variable `string1` in the active IBASIC program — `string2` is the new value.
`PROGram [:SELected][1] :WAIT`	NR1	Waits until the IBASIC program completes.

1. Commands grouped under the `SELected` mnemonic in the `PROGram` subsystem operate on the active program buffer.
2. The parameter type of the data is determined by the format selected — `FORMat REAL` uses BLOCK data, `FORMat ASCii` uses NR3 data separated by commas.

Table 11-18 ROUTe

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>ROUTe[1|2]:REFLection:PATH
:DEFine:PORT<1|2|...|12></td><td>NR1</td><td>Selects which port of the test set is connected to the REFLECTION port of the analyzer.[1]</td></tr>
<tr><td>ROUTe[1|2]:TRANsmission:PATH
:DEFine: PORT <1|2|...|12></td><td>NR1</td><td>Selects which port of the test set is connected to the TRANSMISSION port of the analyzer.[1]</td></tr>
<tr><td>ROUTe[1|2]:PATH:DEFine
: PORT <num>,<num></td><td>NR1</td><td>Selects which port of the analyzer is to function as the reflection (RF out) port and which port is to function as the transmission (RF in) port. Choose from 1,2 (forward), or 2,1 (reverse).[2]</td></tr>
</table>

1. For use with multiport test sets only.
2. For use with the 8712ES and 8714ES models only.

Table 11-19 SENSe (1 of 16)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td><code>SENSe[1|2]:ACTIVE?</code></td><td>query only</td><td>Returns true (1) if the channel is active, false (0) if the channel is not active. (Only one channel can be "active" at a time.)</td></tr>
<tr><td><code>SENSe[1|2]:AVERage:CLEar</code></td><td>command only</td><td>Re-starts the trace averaging function.</td></tr>
<tr><td><code>SENSe[1|2]:AVERage:COUNt <num></code></td><td>NR1</td><td>Specifies a count or weighting factor for the averaged measurement data.</td></tr>
<tr><td><code>SENSe[1|2]:AVERage[:STATe] <ON|OFF></code>[1]</td><td>NR1</td><td>Turns the trace averaging function on/off.</td></tr>
<tr><td><code>SENSe[1|2]:BWIDth [:RESolution] <num> HZ</code></td><td>NR2</td><td>Specifies the bandwidth of the IF receiver (fine, narrow, medium or wide) to be used in the measurement — choose
15 (fine)
250 (narrow)
1200 (medium narrow)
3700 (medium)
4000 (medium wide)
6500 (wide)</td></tr>
<tr><td><code>SENSe[1|2]:CORRection: [OFF/ON]</code>[1]</td><td>NR1</td><td>Turns measurement calibration function on/off. Uncorrected readings are used when "off."</td></tr>
<tr><td><code>SENSe[1|2]:CORRection: ANNotation?</code></td><td>query only</td><td>Returns the current calibration annotation: "C", "C?", "Cx", or "".</td></tr>
<tr><td><code>SENSe[1|2]:CORRection: CAPacitance:CONNector <num></code></td><td>NR3</td><td>Selects a connector compensating capacitance value. (For use with structural return loss measurements on analyzers with Option 100 only.)</td></tr>
</table>

1. NR1 values of 1 (on) and 0 (off) can be used in place of CHAR values ON and OFF.

Table 11-19 SENSe (2 of 16)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:LOAD
:MODify:DELay? #0~1e-06[S]#</td><td>NR3</td><td>Sets or queries the cal kit load standard offset delay time.</td></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:LOAD
:MODify:LOSS? #0~1e+12#</td><td>NR3</td><td>Sets or queries the cal kit load standard offset loss value.</td></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:LOAD
:MODify:ZOFFset? #25~100[OHM]#</td><td>NR1</td><td>Sets or queries the cal kit load standard offset impedance value.</td></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:MODify
[:SElect]
<TYPenf|TYPenm|UD1|TYPe35mm
|TYPeff|TYPe716f|TPe716m|UD2
|TYPeapc7|UD3|UD4|...|UD10></td><td>CHAR</td><td>Selects or queries the cal kit connector type. Select TYPe716f or TYPe716m for modifiable cal kit.</td></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:NAME
<KIT1|KIT2|...|KIT10>,<STRING></td><td>STRING</td><td>Sets the description of the user-defined calibration kit.</td></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:OPEN
:MODify:CONE? #-10000~1[FARAD]#</td><td>NR3</td><td>Sets or queries the cal kit open standard fringe capacitance C1 (f^1, Farad/Hz) value.</td></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:OPEN
:MODify:CTHRee? #-10000~1[FARAD]#</td><td>NR3</td><td>Sets or queries the cal kit open standard fringe capacitance C3 (f^3, Farad/Hz^3) value.</td></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:OPEN
:MODify:CTWO? #-10000~1[FARAD]#</td><td>NR3</td><td>Sets or queries the cal kit open standard fringe capacitance C2 (f^2, Farad/Hz^2) value.</td></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:OPEN
:MODify:CZERo? #-10000~1[FARAD]#</td><td>NR3</td><td>Sets or queries the cal kit open standard fringe capacitance model C0 (f^0, Farad) value.</td></tr>
<tr><td>SENSe[1|2]:CORRection:CKIT:OPEN
:MODify:DELay? #0~1e-06[S]#</td><td>NR3</td><td>Modifies or queries the cal kit open standard offset delay time.</td></tr>
</table>

Table 11-19 SENSe (3 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SENSe[1|2]:CORRection:CKIT:OPEN :MODify:LOSS? #0~1e+12#	NR3	Modifies or queries the cal kit open standard offset loss value.
SENSe[1|2]:CORRection:CKIT:OPEN :MODify:ZOFFset? #25~100[OHM]#	NR3	Modifies or queries the cal kit open standard offset impedance value.
SENSe[1|2]:CORRection:CKIT:PRESet [:IMMediate]-? [<KIT1|KIT2|...|KIT10>]	Command Only	Sets all values of all user-defined cal kits to the default values.
SENSe[1|2]:CORRection:CKIT:SAVE? <KIT1|KIT2|...|KIT10>	CHAR	Saves the user-defined cal kit or queries whether the selected user-defined cal kit has been saved.
SENSe[1|2]:CORRection:CKIT:SHORt :MODify:DELay? #0~1e-06[S]#	NR3	Modifies or queries the cal kit short standard offset delay time.
SENSe[1|2]:CORRection:CKIT:SHORt :MODify:LOSS? #0~1e+12#	NR3	Modifies or queries the cal kit short standard offset loss value.
SENSe[1|2]:CORRection:CKIT:SHORt :MODify:ZOFFset? #25~100[OHM]#	NR1	Modifies or queries the cal kit short standard offset impedance value.
SENSe[1|2]:CORRection:CKIT:THRU :MODify:DELay? #0~1e-06[S]#	NR3	Modifies or queries the cal kit thru standard offset delay time.
SENSe[1|2]:CORRection:CKIT:THRU :MODify:LOSS? #0~1e+12#	NR3	Modifies or queries the cal kit thru standard offset loss value.
SENSe[1|2]:CORRection:CKIT:THRU :MODify:ZOFFset? #25~100[OHM]#	NR3	Modifies or queries the cal kit thru standard offset impedance value.

Table 11-19 SENSe (4 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SENSe[1|2]:CORRection:CLASs [:SELect]?	CHAR	Returns the current calibration choice. Returns string from the <item> list (below). (ES models only)
SENSe[1|2]:CORRection:CLASs [:SELect] <item>	command only	Selects an existing calibration from the following <item> list:
DEFault1		Default factory one-port calibration
DEFault2		Default factory two-port calibration
REFLection3*		User one-port calibration
TRANsmission1*		User response calibration
TRANsmission2*		User response and isolation calibration
TRANsmission3*		User enhanced response calibration
TESTset1		Multi-port testset calibration
TWOPort* calibration		User two-port calibration Multi-port testset
* The requested user calibration may or may not be stored in memory, and may or may not be valid for the current instrument settings. If the requested user calibration is invalid, a valid user calibration will be selected, if found. If no valid user calibration is found, the default factory calibration will be selected. The instrument can be queried with SENSe[1|2]:CORRection:CLASs[:SELect]? to determine the current calibration choice.		
SENSe[1|2]:CORRection:COLLect :ABORt	command only	Aborts the calibration that is currently in progress.

Table 11-19 SENSe (5 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SENSe[1|2]:CORRection:COLLect[:ACQuire] <char>	command only	Measures a calibration standard — select <char> from: STANdard1 STANdard2 STANdard3 STANdard4 STANdard5 STANdard6 STANdard7
SENSe[1|2]:CORRection:COLLect:CKIT:PORT[1|2|…|12][:SELect]<string>	STRING	Selects which port of the analyzer or multiport test set will be assigned a Cal Kit. Also, selects a Cal Kit— choose from one of the following strings: 'COAX,7MM,TYPE-N,50,FEMALE' 'COAX,7MM,TYPE-N,50,MALE' 'COAX,3.5MM,APC-3.5,50,IMPLIED' 'USER|USER1|USER2,IMPLIED, IMPLIED,IMPLIED, IMPLIED' 'COAX,7MM,TYPE-F,75,IMPLIED' 'COAX,7MM,TYPE-N,75,FEMALE' 'COAX,7MM,TYPE-N,75,MALE' 'COAX,7MM,APC-7,50,IMPLIED' 'COAX,7MM,TYPE-7-16,50,FEMALE'
SENSe[1|2]:CORRection:COLLect:CKIT[:SELect]<string>	STRING	Selects a Cal Kit— choose from one of the following strings: 'COAX,7MM,TYPE-N,50,FEMALE' 'COAX,7MM,TYPE-N,50,MALE' 'COAX,3.5MM,APC-3.5,50,IMPLIED' 'USER|USER1|USER2,IMPLIED, IMPLIED,IMPLIED, IMPLIED' 'COAX,7MM,TYPE-F,75,IMPLIED' 'COAX,7MM,TYPE-N,75,FEMALE' 'COAX,7MM,TYPE-N,75,MALE' 'COAX,7MM,TYPE-7-16,50,FEMALE' 'COAX,7MM,TYPE-7-16,50,MALE'

Table 11-19 SENSe (6 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SENSe[1|2]:CORRection:COLLect:ISTate[:AUTO]<ON|OFF> [1]	NR1	Selects the instrument state for calibration — choose Full Band (ON) or User Defined (OFF).
SENSe[1|2]:CORRection:COLLect:METHod <char>	command only	Selects the type of calibration — choose from: TRAN1 — Transmission response TRAN2 — Transmission response & Isolation TRAN3 — Transmission enhanced response REFL3 — Reflection one port TESTset — Test Set Calibration (Enhanced Response/1-Port)[1] TESTset1 — Test Set Calibration (Enhanced Response/1-Port)[1] TESTset2 — Test Set Calibration (2-Port)[1] VERIFY — Calibration Check TWOPort — Two-port calibration NONE — No calibration
SENSe[1|2]:CORRection:COLLect:MP:OPEN <STAN1|STAN2|...|STAN12>	command only	Measures an open on the port selected during a test set calibration. [2]
SENSe[1|2]:CORRection:COLLect:MP:SHORT <STAN1|STAN2|...|STAN12>	command only	Measures a short on the port selected during a test set calibration. [2]

1. Binary parameters accept the values of 1(on) and 0(off) in addition to ON and OFF.
2. For use with multiport test sets only.

Table 11-19 SENSe (7 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SENSe[1|2]:CORRection:COLLect :MP:LOAD <STAN1|STAN2|...|STAN12>	command only	Measures a load on the port selected during a test set calibration. [2]
SENSe[1|2]:CORRection:COLLect :MP:THRU <STAN1|STAN2|...|STAN6>	command only	Measures a thru on the port selected during a test set calibration. [2]
SENSe[1|2]:CORRection:COLLect :PORTS <2|4|6|8|10|12>	NR1	Selects the number of ports to perform a test set calibration on. [1]
SENSe[1|2]:CORRection:COLLect :SAVE	command only	Completes and saves current calibration.
SENSe[1|2]:CORRection:COLLect :VERify:REFLection <STAN1|STAN2|...|STAN12>	command only	Measures a calibration standard during a cal check procedure for reflection measurements.
SENSe[1|2]:CORRection:COLLect :VERify:TRANsmission <STAN1|STAN2|...|STAN12>;	command only	Measures a calibration standard during a cal check procedure for transmission measurements.
SENSe[1|2]:CORRection:CSET [:SELect] DEFault	command only	Restores the "factory" default calibration for the current measurement and channel.
SENSe[1|2]:CORRection:CSET [:SELect]?	query only CHAR	Queries the current calibration type — returns DEF (factory default), FULL (full band) or USER (user defined).
SENSe[1|2]:CORRection:EDELay :TIME <num> [2]	NR3	Specifies the electrical delay in seconds.
SENSe[1|2]:CORRection:EXTension:REFLection[:TIME]<num> [2]	NR3	Specifies the port extension at the reflection port, in seconds.

1. For use with multiport test sets only.
2. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is assumed.

Table 11-19 SENSe (8 of 16)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>SENSe[1|2]:CORRection:EXTension[:STATe]<ON|OFF> [1]</td><td>NR1</td><td>Enables port extensions.</td></tr>
<tr><td>SENSe[1|2]:CORRection:EXTension:TRANsmission[:TIME] <num> [2]</td><td>NR3</td><td>Specifies the port extension at the transmission port, in seconds.</td></tr>
<tr><td>SENSe[1|2]:CORRection:IMPedance:INPut:MAGNitude<num>[2]</td><td>NR3</td><td>Specifies the reference impedance for the Smith chart display. The default is the analyzer's system impedance.</td></tr>
<tr><td>SENSe[1|2]:CORRection:IMPedance:INPut:MAGNitude:SELect ZO_50</td><td>NR1</td><td>Selects 50 ohms as the system impedance.</td></tr>
<tr><td>SENSe[1|2]:CORRection:IMPedance:INPut:MAGNitude:SELect ZO_75</td><td>NR1</td><td>Selects 75 ohms as the system impedance.</td></tr>
<tr><td>SENSe[1|2]:CORRection:ISOLation[:STATe] {OFF|0|ON|1}</td><td>NR1</td><td>Enables or disables use of isolation error correction when 2-port calibrations are selected.</td></tr>
<tr><td>SENSe[1|2]:CORRection:LENGth:COAX <num>[2]</td><td>NR2</td><td>Specifies the length of cable to be calibrated, in feet or meters.[3]</td></tr>
<tr><td>SENSe[1|2]:CORRection:LENGth:CONNector <num>[2]</td><td>NR2</td><td>Specifies the length of an interface connector, in mm or inches. [3]</td></tr>
<tr><td>SENSe[1|2]:CORRection:LOSS:COAX <num>[2]</td><td>NR2</td><td>Specifies the loss of a cable under test, in dB/100 ft. [3]</td></tr>
</table>

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
2. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is used.
3. For use with structural return loss measurements using analyzers with Option 100 only.

Table 11-19 SENSe (9 of 16)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>SENSe[1|2]:CORRection:MODel :CONNector[:IMMediate]</td><td>command only</td><td>Measures the cable connector and determine the optimum values for connector length and connector capacitance. [3]</td></tr>
<tr><td>SENSe[1|2]:CORRection:OFFSet :PHASe</td><td>NR3</td><td>Specifies the phase offset.</td></tr>
<tr><td>SENSe[1|2]:CORRection:ONEPort :REFLection[:IMMediate]</td><td>command only</td><td>Retrieves the user one-port reflection error correction factors from internal memory and applies them to the current measurement.</td></tr>
<tr><td>SENSe[1|2]:CORRection:ONEPort :TRANSmission[:IMMediate]</td><td>command only</td><td>Retrieves the user one-port transmission error correction factors from internal memory and applies them to the current measurement.</td></tr>
<tr><td>SENSe[1|2]:CORRection:TWOPort [:IMMediate]</td><td>command only</td><td>Retrieves the user two-port error correction factors from internal memory and applies them to the current measurement.</td></tr>
<tr><td>SENSe[1|2]:CORRection:PEAK :COAX[:STATe]<ON|OFF> [1]</td><td>NR1</td><td>Turns multi-peak correction on or off.[2]</td></tr>
<tr><td>SENSe:CORRection:RVELocity :COAX <num> [3]</td><td>NR3</td><td>Specifies the velocity factor to be used when displaying the distance for electrical length and port extensions. 1.0 the speed of light.</td></tr>
<tr><td>SENSe[1|2]:CORRection:RVELocity [:IMMediate]</td><td>command only</td><td>Measures the cable and determine the optimum values for cable loss and velocity factor. [2]</td></tr>
</table>

1. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.
2. For use with fault location measurements on analyzers with Option 100 only.
3. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (Hz for frequency or S for time) is assumed.

Table 11-19 SENSe (10 of 16)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>SENSe[1|2]:CORRection:TESTSET</td><td>command only</td><td>Brings up the Test Set Cal menu. [2]</td></tr>
<tr><td>SENSe[1|2]:CORRection:THReshold:COAX <num></td><td>NR2</td><td>Selects multi-peak threshold value, in dB.[1]</td></tr>
<tr><td>SENSe[1|2]:COUPle <char></td><td>CHAR</td><td>Turns the alternate sweep mode on/off — choose ALL (coupled sweep) or NONE (alternate sweep).</td></tr>
<tr><td>SENSe[1|2]:DETector[:FUNCtion] <char></td><td>CHAR</td><td>Specifies which detection mode is used to make the measurement — choose BBANd (broadband) or NBANd (narrowband).</td></tr>
<tr><td>SENSe[1|2]:DISTance:CENTer <num></td><td>NR3</td><td>Sets the center distance for a fault location measurement, in feet or meters. [1]</td></tr>
<tr><td>SENSe[1|2]:DISTance:STARt <num></td><td>NR3</td><td>Sets the start distance for a fault location measurement, in feet or meters.[1]</td></tr>
<tr><td>SENSe[1|2]:DISTance:UNITs <char></td><td>CHAR</td><td>Specifies distance units. Choose METers or FEET.[1]</td></tr>
<tr><td>SENSe[1|2]:DISTance:STOP <num></td><td>NR3</td><td>Sets the stop distance for a fault location measurement, in feet or meters. [1]</td></tr>
<tr><td>SENSe[1|2]:FREQuency:CENTer <num> [2]</td><td>NR3</td><td>Sets the center frequency of the RF source.</td></tr>
</table>

1. For use with fault location measurements on analyzers with Option 100 only.
2. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is assumed.

Table 11-19 SENSe (11 of 16)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>SENSe[1|2]:FREQuency:MODE <char></td><td>CHAR</td><td>Sets the fault location measurement to CENTer (bandpass) or LOWPass.[2]</td></tr>
<tr><td>SENSe[1|2]:FREQuency:SPAN <num> [1]</td><td>NR3</td><td>Sets the frequency span of the RF source.</td></tr>
<tr><td>SENSe[1|2]:FREQuency:SPAN :MAXimum <num> [1]</td><td>NR3</td><td>Sets the maximum frequency span of the RF source for bandpass fault location measurements. [2]</td></tr>
<tr><td>SENSe[1|2]:FREQuency:STARt <num>[1]</td><td>NR3</td><td>Sets the start frequency of the RF source.</td></tr>
<tr><td>SENSe[1|2]:FREQuency:STOP <num> [1]</td><td>NR3</td><td>Sets the stop frequency of the RF source.</td></tr>
<tr><td>SENSe[1|2]:FREQuency:ZSTOp <num> [1]</td><td>NR3</td><td>Sets the Z cutoff frequency for cable impedance calculations.[2]</td></tr>
<tr><td>SENSe[1|2]:FUNCtion?</td><td>query only STRING</td><td>See SENSe[1|2]:FUNCtion '. . . . .' commands for string definitions. Queries the measurement function — returns a string that defines the current measurement function.</td></tr>
</table>

1. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (Hz for frequency or S for time) is assumed.
2. For use with fault location measurements on analyzers with Option 100 only.

Table 11-19 SENSe (12 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`SENSe[1|2]:FUNCtion 'FLOC <num>,<num>'`	command only	Specifies that the receiver will measure the ratio of the power (fault location) into the specified measurement channel. *For 8712ET/8714ET models*—choose `1,0` (Forward A/R). *For 8712ES/8714ES models*—choose `1,0` (Forward A/R), or `22, 20` (Reverse B/R). (For use with fault location measurement on analyzers with Option 100 only.)
`SENSe[1|2]:FUNCtion 'XFRequency:POWer <num>'`	command only	Specifies that the receiver will measure the power into a specific detector on the specified measurement channel. *For 8712ET/8714ET models*— choose from detectors `0` (R), `1` (A), `2` (B), `11` (Ext X) or `12` (Ext Y). *For 8712ES/8714ES models*—choose from detectors: `0` (Forward R) `20` (Reverse R) `1` (Forward A) `21` (Reverse A) `11` (Forward Ext X) `12` (Forward Ext Y) `2` (Forward B) `22` (Reverse B)

Table 11-19 SENSe (13 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`SENSe[1|2]:FUNCtion 'XFRequency:GDELay:RATio <num>,<num>'`	command only	Specifies that the receiver will measure the ratio of the power (group delay) into the specified measurement channel. *For 8712ET / 8714ET models*—choose ratio `2,0` (B/R). *For 8712ES / 8714ES models*—choose from ratios `2,0` (Forward B/R), or `21,20` (Reverse A/R).
`SENSe[1|2]:FUNCtion 'XFRequency:POWer:RATio <num>,<num>'`	command only	Specifies that the receiver will measure a ratio of the power into the specified measurement channel. *For 8712ET / 8714ET models*— choose from ratios `1,0` (A/R), `2,0` (B/R), `12,0` (Ext Y/R), `11,12` (Ext X/Ext Y), or `12,11` (Ext Y/Ext X). *For 8712ES / 8714ES models*—choose from ratios: `1,0` (Forward A/R) `21,20` (Reverse A/R) `2,0` (Forward B/R) `22,20` (Reverse B/R) `12,0` (Forward Ext Y/R) `11,12` (Forward Ext X/Ext Y) `12,11` (Forward Ext Y/Ext X)

Table 11-19 SENSe (14 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SENSe[1|2]:FUNCtion 'XFRequency:S <num>,<num>'	NR1,NR1	Specifies that the receiver will measure an s-parameter into the specified measurement channel. The s-parameter choices will depend on the number of ports. For example, with a 2-port device, choose from 1,1 (S_{11}), 1,2 (S_{12}), 2,2 (S_{22}), or 2,1 (S_{21}). As another example, with a 12-port device using the 87075C multiport test set, some choices would include 1,8 (S_{18}), or 12,1 ($S_{12,1}$), or 7,7 (S_{77}). This command may be used in place of: SENS[1|2]:FUNC 'XFR:POW:RAT and ROUT[1|2}:PATH:DEF:PORT. If using the 87075C multiport test set, this command may be used in place of the two commands just listed, as well as: ROUT[1|2]:REFL:PATH:DEF:PORT and OUT[1|2]:TRAN:PATH:DEF:PORT.[1]
SENSe[1|2]:FUNCtion:FAULt :CONNector [:IMMediate]	command only	Forces a connector verification measurement on the alternate channel. (For use with SRL measurements on analyzers with Option 100 only.)

1. For use with the 8712ES and 8714ES models only.

Table 11-19 SENSe (15 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`SENSe[1|2]:FUNCtion:SRL :<num>,<num>`[1]	NR1,NR1	Specifies that the receiver will measure the ratio of the power (SRL) into the specified measurement channel[2]. *For 8712ET / 8714ET models*—choose `1,0` (Forward A/R). *For 8712ES / 8714ES models*—choose `1,0` (Forward A/R), or 22, 20 (Reverse B/R).
`SENSe[1|2]:FUNCtion:SRL :IMPedance <num>`[1]	NR2	Sets the cable impedance. [2]
`SENSe[1|2]:FUNCtion:SRL :MODE <char>`	CHAR	Sets the auto z function to `AUTO` or `MANual`.[2]
`SENSe[1|2]:FUNCtion:SRL :SCAN[:IMMediate]`	command only	Starts a cable scan.[2]
`SENSe[1|2]:ROSCillator :SOURce <char>`	CHAR	Specifies the source of the reference oscillator — select `INTernal` or `EXTernal`.
`SENSe[1|2]:STATe <ON|OFF>` [3]	NR1	Turns the specified channel on/off .

1. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is assumed.
2. For use with structural return loss measurements on analyzers with Option 100 only.
3. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF.

Table 11-19 SENSe (16 of 16)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`SENSe[1|2]:SWEep:POINts <num>`	NR1	Sets the number of data points for the measurement — choose from `3|5|11|21|51|101|201|401|801|1601`.
`SENSe[1|2]:SWEep:TIME <num>` [1]	NR3	Sets the sweep time.
`SENSe[1|2]:SWEep:TIME:AUTO <ON|OFF|ONCE>` [2]	CHAR or NR1	Turns the automatic sweep time function on/off.
`SENSe:SWEep:TRIGger:SOURce <char>`	CHAR	Sets the trigger source for each point in a sweep — choose `IMMediate` or `EXTernal` (used in conjunction with `TRIGger[:SEQuence]:SOURce`).
`SENSe[1|2]:WINDow[:TYPE] <char>`	CHAR	Sets the window selection for fault location measurements. Choose from `RECTangular` (Minimum), `HAMMing` (Medium), or `KBESsel` (Maximum). (For use with fault location measurements on analyzers with Option 100 only.)

1. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is assumed.
2. Binary parameters accept the values of 1 (on) and 0 (off) in addition to ON and OFF S_{21}.

Table 11-20 SOURce

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>SOURce[1|2]:POWer[:LEVel]
[:IMMediate][:AMPLitude]<num> [1]</td><td>NR3</td><td>Sets the RF power output from the source.</td></tr>
<tr><td>SOURce:POWer:PRESet <num></td><td>NR3</td><td>Sets the power level that the analyzer will always return to after an instrument preset.</td></tr>
<tr><td>SOURce[1|2]:POWer:RANGe <char></td><td>CHAR</td><td>Specifies the power sweep range. Choose from ATTen0
|ATTen10|ATTen20|ATTen30
|ATTen40 |ATTen50|ATTen60.</td></tr>
<tr><td>SOURce[1|2]:POWer:STARt <num></td><td>NR3</td><td>Sets the power sweep start power.</td></tr>
<tr><td>SOURce[1|2]:POWer:STOP <num></td><td>NR3</td><td>Sets the power sweep stop power.</td></tr>
</table>

1. Numeric parameters may include an appropriate suffix; if no suffix is included, the default (HZ for frequency or S for time) is assumed.

Table 11-21 STATus (1 of 4)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`STATus:DEVice:CONDition?`	query only NR1	Reads the Device Status condition register. [1]
`STATus:DEVice:ENABle <num>`	NR1	Sets and queries bits in the Device Status enable register. [2]
`STATus:DEVice[:EVENt]?`	query only NR1	Reads and clears the Device Status event register. [1]
`STATus:DEVice:NTRansition <num>`	NR1	Sets and queries bits in the Device Status negative transition register. [2]
`STATus:DEVice:PTRansition <num>`	NR1	Sets and queries bits in the Device Status positive transition register. [2]
`STATus:OPERation:AVERaging :CONDition?`	query only NR1	Reads the Averaging status condition register. [1]
`STATus:OPERation:AVERaging: ENABle <num>`	NR1	Sets and queries bits in the Averaging status enable register. [2]
`STATus:OPERation:AVERaging [:EVENt]?`	query only NR1	Reads and clears the Averaging status event register. [1]
`STATus:OPERation:AVERaging :NTRansition <num>`	NR1	Sets and queries bits in the Averaging status negative transition register. [2]

1. Returns the sum of the decimal weights (2^n where n is the bit number) of all bits currently set. For more information on using the status registers, refer to Chapter 5, "Using Status Registers."
2. <num> is the sum of the decimal weights of all bits to be set.

Table 11-21 STATus (2 of 4)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`STATus:OPERation:AVERaging :PTRansition <num>`	NR1	Sets and queries bits in the Averaging status positive transition register. [1]
`STATus:OPERation:CONDition?`	query only NR1	Reads the Operational status condition register. [2]
`STATus:OPERation:ENABle <num>`	NR1	Sets and queries bits in the Operational status enable register. [1]
`STATus:OPERation[:EVENt]?`	query only NR1	Reads and clears the Operational status event register. [2]
`STATus:OPERation:MEASuring :CONDition?`	query only NR1	Reads the Measuring status condition register. [2]
`STATus:OPERation:MEASuring :ENABle <num>`	NR1	Sets and queries bits in the Measuring status enable register. [1]
`STATus:OPERation:MEASuring [:EVENt]?`	query only NR1	Reads and clears the Measuring status event register. [2]
`STATus:OPERation:MEASuring :NTRansition <num>`	NR1	Sets and queries bits in the Measuring status negative transition register. [1]
`STATus:OPERation:MEASuring :PTRansition <num>`	NR1	Sets and queries bits in the Measuring status positive transition register. [1]

1. <num> is the sum of the decimal weights of all bits to be set.
2. Returns the sum of the decimal weights (2^n where n is the bit number) of all bits currently set. For more information on using the status registers refer to Chapter 5, "Using Status Registers."

Table 11-21 STATus (3 of 4)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
STATus:OPERation:NTRansition <num>	NR1	Sets and queries bits in the Operational Status negative transition register. [1]
STATus:OPERation:PTRansition <num>	NR1	Sets and queries bits in the Operational Status positive transition register. [1]
STATus:PRESet	command only	Sets bits in most enable and transition registers to their default state.
STATus:QUEStionable:CONDition?	query only NR1	Reads the Questionable Status condition register. [2]
STATus:QUEStionable:ENABle <num>	NR1	Sets and queries bits in the Questionable Status enable register. [1]
STATus:QUEStionable[:EVENt]?	query only NR1	Reads and clears the Questionable Status event register. [2]
STATus:QUEStionable:LIMit :CONDition?	query only NR1	Reads the Limit Fail condition register. [2]

1. <num> is the sum of the decimal weights of all bits to be set.
2. Returns the sum of the decimal weights (2^n where n is the bit number) of all bits currently set. For more information on using the status registers refer to Chapter 5, "Using Status Registers."

Table 11-21 STATus (4 of 4)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
STATus:QUEStionable:LIMit:ENABle <num>	NR1	Sets and queries bits in the Limit Fail enable register. [1]
STATus:QUEStionable:LIMit[:EVENt]?	query only NR1	Reads and clears the Limit Fail event register. [2]
STATus:QUEStionable:LIMit:NTRansition <num>	NR1	Sets and queries bits in the Limit Fail negative transition register. [1]
STATus:QUEStionable:LIMit:PTRansition <num>	NR1	Sets and queries bits in the Limit Fail positive transition register. [1]
STATus:QUEStionable:NTRansition <num>	NR1	Sets and queries bits in the Questionable Status negative transition register. [1]
STATus:QUEStionable:PTRansition <num>	NR1	Sets and queries bits in the Questionable Status positive transition register. [1]

1. <num> is the sum of the decimal weights of all bits to be set.
2. Returns the sum of the decimal weights (2^n where n is the bit number) of all bits currently set. For more information on using the status registers refer to Chapter 5, "Using Status Registers."

Table 11-22 SYSTem (1 of 8)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SYSTem:BEEPer[:IMMediate] [<freq>[,<dur>[,<vol>]][1]	NR3, NR3, NR3	Instructs the analyzer to beep. Arguments are frequency (Hz), duration (seconds), and volume (0 to 1).
SYSTem:BEEPer:VOLume <num>	NR2	Sets the volume of the beeper — num is a number between 0 for 0% and 1 for 100%.
SYSTem:COMMunicate:GPIB :CONTroller[:STATe]<ON|OFF>[2,3]	NR1	Makes the analyzer the system controller.
SYSTem:COMMunicate:GPIB:ECHO <ON|OFF>[2]	NR1	Turns GPIB mnemonic echo on/off .
SYSTem:COMMunicate:GPIB:HCOPy :ADDRess <num>	NR1	Sets the address of an GPIB printer or plotter for hardcopy output — num must be an integer between 0 and 30.
SYSTem:COMMunicate:GPIB[:SELF] :ADDRess <num>[4]	NR1	Sets the analyzer's GPIB address — num must be an integer between 0 and 30.
SYSTem:COMMunicate:LAN:BOOTp :HOST <string>	STRING	Sets the host name or host IP address of the remote host that receives the BOOTP requests.
SYSTem:COMMunicate:LAN:BOOTp :STATE <OFF|0|ON|1>	NR1	Sets a BOOTP request when the analyzer boots up.

1. <freq>, <dur>, and <vol> are optional <num> parameters.
2. Binary parameters accept the values of 1 (on) and 0 (off) inaddition to ON and OFF.
3. For use with IBASIC—this command cannot be executed from an external controller.
4. A delay of 5 seconds is required before a command is sent to the new address.

Table 11-23 SYSTem (2 of 8)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SYSTem:COMMunicate:LAN:BOOTp :TIMeout #1~MAX_AUTO_CAL_TIME[S]#	NR1	Sets the number of seconds the analyzer will retry the BOOTP requests at boot time.
SYSTem:COMMunicate:LAN:BOOTp :TRANsfer:FILE:NAME <string>	STRING	Sets the path file name of the boot file you want to receive at boot time.
SYSTem:COMMunicate:LAN:BOOTp :TRANsfer:FTP:PASSword-? <string>	command only	Selects the password of the remote BOOTP host.
SYSTem:COMMunicate:LAN:BOOTp :TRANsfer:FTP:USERname <string>	STRING	Selects the user name of the remote BOOTP host.
SYSTem:COMMunicate:LAN:BOOTp :TRANsfer:METHod <TFTP|FTP>	NR1	Selects TFTP as the file transfer program.
SYSTem:COMMunicate:LAN:EADDress?	query only STRING	Queries the analyzer's ethernet address.
SYSTem:COMMunicate:LAN:IPADdress <string>	STRING	Sets the analyzer's Internet Protocol address.
SYSTem:COMMunicate:LAN:LOGin :USER:ADD-? <string>,<string>	command only	Selects the login user/password pairs.
SYSTem:COMMunicate:LAN:LOGin :USER:DELETE-? <string>,<string>	command only	Deletes the login user/password pairs.
SYSTem:COMMunicate:LAN:LOGin :USER:LIST:COUNt?	query only NR1	Queries login user names.
SYSTem:COMMunicate:LAN:LOGin :USER:LIST:NAME? #1-7	query only STRING	Queries user name.
SYSTem:COMMunicate:LAN:NFS :AUTHentiation:ID:GROup #0~4.74836e+07#	NR2	Selects the NFS remote file system for entering a Group ID.

Table 11-24 SYSTem (3 of 8)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SYSTem:COMMunicate:LAN:NFS :AUTHentiation:ID:USER #0~4.74836e+07#	NR2	Selects the NFS remote file system for entering a User ID.
SYSTem:COMMunicate:LAN:NFS :AUTOmount:ADD-? <string>,<string>,<string>	command only	Sets the mounted NFS device to the automount device table.
SYSTem:COMMunicate:LAN:NFS :AUTOmount:LIST:COUNt?	query only NR1	Selects the number of NFS devices in Automount list.
SYSTem:COMMunicate:LAN:NFS :AUTOmount:LIST:LOCFilesys? #1-7	query only STRING	Selects the local file system name for an NFS device.
SYSTem:COMMunicate:LAN:NFS :AUTOmount:LIST:REMFilesys? #1-7	query only STRING	Queries the automount list.
SYSTem:COMMunicate:LAN:NFS :AUTOmount:LIST:REMHost? #1-7	query only STRING	Queries the remote host name (or host IP address) for an NFS device.
SYSTem:COMMunicate:LAN:NFS :AUTOmount:REMove-? <string>	command only	Removes device from automount list.
SYSTem:COMMunicate:LAN:NFS :MOUNT-? <string>,<string>,<string>	command only	Selects the remote host, file system, and local system name for an NFS device.
SYSTem:COMMunicate:LAN:NFS :MOUNT:LIST:COUNt?	query only NR1	Queries the number of currently mounted NFS devices.
SYSTem:COMMunicate:LAN:NFS :MOUNT:LIST:LOCFilesys? #1-7	query only STRING	Queries the local file system name.
SYSTem:COMMunicate:LAN:NFS :MOUNT:LIST:REMFilesys? #1-7	STRING	Selects the remote host file system name for an NFS device.

Table 11-23 SYSTem (4 of 8)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`SYSTem:COMMunicate:LAN:NFS :MOUNT:LIST:REMHost? #1-7`	STRING	Selects the remote host name (or host IP address) for an NFS device.
`SYSTem:COMMunicate:LAN:NFS :UNMount-? <string>`	NONE	Selects the NFS device table.
`SYSTem:COMMunicate:LAN:PRINter :HOSTname <string>`	STRING	Specifies the IP address of the LAN printer.
`SYSTem:COMMunicate:LAN:ROUTe :GATeway <string>`	STRING	Sets the IP address for a LAN gateway.
`SYSTem:COMMunicate:LAN:ROUTe :SMASk <string>`	STRING	Sets the subnet mask.
`SYSTem:COMMunicate:LAN:SCPI :SOCKet:PORT <num>`	NR1	Selects the port number for a socket connection to the analyzer for SCPI socket programming.
`SYSTem:COMMunicate:LAN:SICL :GPIB:ADDRess #0~20, 22~30#`	NR1	Sets the GPIB address for SICL LAN.
`SYSTem:COMMunicate:LAN:SICL :GPIB:LU #0~1024#`	NR1	Sets the GPIB logical unit number for SICL LAN.
`SYSTem:COMMunicate:LAN:SICL :GPIB:NAME <string>`	STRING	Sets the GPIB name for SICL LAN.

Table 11-23 SYSTem (5 of 8)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
SYSTem:COMMunicate:SERial :TRANsmit:BAUD <num>	NR1	Sets the baud rate for hardcopy output to a device on the serial port — choose from 1200|2400|4800|9600|19200.
SYSTem:COMMunicate:SERial :TRANsmit:HANDshake <char>	CHAR	Sets the handshake for communication to a hardcopy device on the serial port — choose XON or DTR.
SYSTem:COMMunicate:TTL:USER: FEED <char>	CHAR	Selects the function of the USER TTL IN/OUT port on the rear panel of the analyzer. Choose from DEFault|KEY|SWEep.
SYSTem:DATE <num1>,<num2>,<num3>	NR1, NR1, NR1	Sets the year (num1), month (num2) and day (num3) of the real time clock.
SYSTem:ERRor? [1]	query only NR1, STRING	Queries the error queue — returns the error number and message.
SYSTem:KEY <char>	command only	Sends key names[2] which execute the same functions as front panel keys.

1. For more information on errors, refer to Chapter 13, "SCPI Error Messages."
2. A list analyzer front panel key codes is provided in Chapter 8, "Front Panel Keycodes."

Table 11-23 SYSTem (6 of 8)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`SYSTem:KEY:MASK?`	query only NR1	Queries the mask (shift, ctrl, alt) associated with a keypress on an external keyboard.
`SYSTem:KEY:QUEue:CLEar`	command only	Clears the key queue.
`SYSTem:KEY:QUEue:COUNt?`	query only NR1	Queries the number of key codes in the queue.
`SYSTem:KEY:QUEue:MAXimum?`	query only NR1	Queries the size of the key queue (the maximum number of key codes it can hold).
`SYSTem:KEY:QUEue[:STATe] <ON|OFF>` [1]	NR1	Turns on/off the key queue.
`SYSTem:KEY:TYPE?`	query only CHAR	Queries the type of key that was pressed — returns `NONE`, `RPG`, `KEY` (front panel key) or `ASC` (external keyboard).
`SYSTem:KEY:USER`	command only	Sets the User Request bit of the Standard Event Status Register.
`SYSTem:KEY[:VALue]?`	query only NR1	Queries the key code value for the last key pressed — `RPG` type returns the knob count, positive for clockwise rotation, `KEY` type returns the front panel keycode, [2] and `ASC` type returns the ASCII code number.

1. Binary parameters accept the values of `1` (on) and `0` (off) in addition to `ON` and `OFF`.
2. A list of the analyzer's front panel keycodes and key names is provided in Chapter 8, "Front Panel Keycodes."

Table 11-23 SYSTem (7 of 8)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`SYSTem:PRESet`	command only	Performs a system preset — this is the same as the front panel (PRESET) key.
`SYSTem:SET <block>`	command only	Sends a learn string (obtained using `*LRN?`) to the analyzer — this command is included in the learn string.
`SYSTem:SET:LRN? [<USER>]` [1]	BLOCK	Queries or set the instrument state.
`SYSTem:SET:LRNLong? [<USER>]` [1]	BLOCK	Queries or set the instrument state, data, and calibration. Similar to save/recall.
`SYSTem:TIME <num1>,<num2>,<num3>`	NR1, NR1, NR1	Sets the hour (`num1`), minute (`num2`) and second (`num3`) of the real time clock.
`SYSTem:VERSion?`	query only NR2	Queries the SCPI version of the analyzer. See `*IDN?` to query the firmware revision.

1. Refer to "Automated Measurement Setup and Control" in Chapter 7 of the *User's Guide* for more information on using this command.

Table 11-24 SYSTem (8 of 8)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`TEST:RESult?`	query only CHAR	Queries the result of the selected adjustment or self-test — the response will be `NULL|PASS|FAIL`.
`TEST:SELect <num>`	NR1	Selects the adjustment or self-test to execute.
`TEST:STATe <char>`	CHAR	Selects the state of the active adjustment or self-test — choose from `RUN|CONTinue|STOP` for the command. Query returns `NULL|RUN|PAUS|DONE`.
`TEST:VALue <num>`	NR1	Sets or queries a value for an adjustment or self-test.

Table 11-25 TRACe (1 of 2)

SUBSYSTEM COMMANDS	FORM	DESCRIPTION	
TRACe[:DATA]? <char>	query only BLOCK or NR3[1]	Queries trace data — choose from the following array types:	
		CH<1|2>FDATA	Formatted data[2]
		CH<1|2>FMEM	Formatted memory data[2]
		CH<1|2>SDATA	Unformatted data[3]
		CH<1|2>SMEM	Unformatted memory data[3]
		CH<1|2><A|B|R>FWD	Raw data
		CH<1|2>SCORR<X>	Two-port error correction terms. Choose X from the list below:

Array choices for two-port error correction

1 Forward directivity	7 Reverse directivity
2 Forward source matching	8 Reverse source matching
3 Forward reflection tracking	9 Reverse reflection tracking
4 Forward transmission tracking	10 Reverse transmission tracking
5 Forward load matching	11 Reverse load matching
6 Forward isolation	12 Reverse isolation

1. The parameter type of the data is determined by the format selected — FORMat REAL uses BLOCK data, FORMat ASCii uses NR3 data separated by commas.
2. Single magnitude value for each measured point.
3. Corrected data in real/imaginary pairs for each measured point.

Table 11-25 TRACe (2 of 2)

<table>
<tr><th>SUBSYSTEM COMMANDS</th><th>FORM</th><th>DESCRIPTION</th></tr>
<tr><td>TRACe[:DATA] <char>,<data></td><td>command only</td><td>Inputs trace data — choose from the above list of arrays. The data can be either BLOCK or NR3 type.[1] Note: See Chapter 6 for more information.</td></tr>
<tr><td>TRACe[:DATA] <char1>,<char2></td><td>command only</td><td>Moves data from one internal array to another — char1 is the target array (CH<1|2>SMEM) while char2 is the source array (CH<1|2>SDATA). Note that the source and target arrays must be from the same measurement channel.</td></tr>
<tr><td>TRACe:CORRection:SIMulate [:DATA] {STD1|STD2|...|STD12}</td><td>command only</td><td>Reads simulated calibration standards. Loads pre-computed or stored measurements of calibration standards from a remote controller into the analyzer.[2]</td></tr>
<tr><td>TRACe:CORRection:SIMulate:SAVE <TRAN1|TRAN2|TRAN3|REFL3|TESTset|TESTset1|TESTset2|VERIFY||TWOPort |NONE></td><td>command only</td><td>Computes cal error correction arrays.</td></tr>
</table>

1. The parameter type of the data is determined by the format selected — FORMat REAL uses BLOCK data, FORMat ASCii uses NR3 data separated by commas.
2. See the example program titled "SIMCAL." This program demonstrates how to create 2-port correction arrays from measurements of the raw (uncorrected) calibration standards when using the analyzer's simcal command.
   Example programs can be found in the following four locations:
   • Example Programs Disk, 8712ET/ES and 8714ET/ES (DOS format): part number 08714-10003
   • Example Programs Disk, 8712ET/ES and 8714ET/ES (LIF format): part number 08714-10004
   • Web site **http://www.agilent.com**. Use the search function to find Web pages related to 8712 example programs.
   • *Example Programs Guide, 8712ET/ES and 8714ET/ES*: part number 08714-90016

Table 11-26 TRIGger

SUBSYSTEM COMMANDS	FORM	DESCRIPTION
`TRIGger[:SEQuence]:SOURce <char>`	CHAR	Sets the source for the sweep trigger signal — choose `IMMediate` or `EXTernal` (used in conjunction with `SENSe:SWEep:TRIGger:SOURce`).

12 SCPI Conformance Information

SCPI Conformance Information

The 8712ET/ES and 8714ET/ES RF Network Analyzers conform to the 1996.0 version of SCPI.

SCPI Standard Commands

The analyzer implements the following IEEE 488.2 standard commands:

- *CLS
- *ESE
- *ESE?
- *ESR?
- *IDN?
- *LRN?
- *OPC
- *OPC?
- *OPT?
- *PCB
- *PSC
- *RST
- *SRE
- *SRE?
- *STB?
- *TRG
- *TST?
- *WAI

The analyzer implements the following SCPI 1996.0 standard commands:

- ABORt
- CALCulate[1|2]:DATA?
- CALCulate[1|2]:FORMat
- CALCulate[1|2]:FORMat?
- CALCulate[1|2]:GDAPerture:APERture

- CALCulate[1|2]:GDAPerture:SPAN
- CALCulate[1|2]:LIMit:STATe
- CALCulate[1|2]:LIMit:STATe?
- CALCulate[1|2]:MATH[:EXPRession]
- CALCulate[1|2]:MATH[:EXPRession]?
- CALibration:ZERO:AUTO
- CALibration:ZERO:AUTO?
- DISPlay:CMAP:COLor[1|2|…16]:HSL
- DISPlay:CMAP:COLor[1|2|…16]:HSL?
- DISPlay:CMAP:COLor[1|2|…16]:RGB
- DISPlay:CMAP:COLor[1|2|…16]:RGB?
- DISPlay:CMAP:DEFault
- DISPlay:MENU[1|2]:KEY[1|2|…7]?
- DISPlay:WINDow[1|2|10]:GEOMetry:LLEFT?
- DISPlay:WINDow[1|2|10]:GEOMetry:SIZE?
- DISPlay:WINDow[1|2|10]:GEOMetry:URIGHT?
- DISPlay:WINDow[1|2|10]:GRAPhics:CLEar
- DISPlay:WINDow[1|2|10]:GRAPhics:COLor
- DISPlay:WINDow[1|2|10]:GRAPhics:COLor?
- DISPlay:WINDow[1|2|10]:GRAPhics[:DRAW]
- DISPlay:WINDow[1|2|10]:GRAPhics:LABel
- DISPlay:WINDow[1|2|10]:GRAPhics:MOVE
- DISPlay:WINDow[1|2|10]:GRAPhics:MOVE?
- DISPlay:WINDow[1|2|10]:GRAPhics:STATe?
- DISPlay:WINDow[1|2]:TRACe:GRATicule:GRID[:STATe]
- DISPlay:WINDow[1|2]:TRACe:GRATicule:GRID[:STATe]?
- DISPlay:WINDow[1|2]:TRACe[1|2][:STATe]
- DISPlay:WINDow[1|2]:TRACe[1|2][:STATe]?

- DISPlay:WINDow[1|2]:TRACe:Y[:SCALe]:AUTO
- DISPlay:WINDow[1|2]:TRACe:Y[:SCALe]:PDIVision
- DISPlay:WINDow[1|2]:TRACe:Y[:SCALe]:PDIVision?
- DISPlay:WINDow[1|2]:TRACe:Y[:SCALe]:RLEVel
- DISPlay:WINDow[1|2]:TRACe:Y[:SCALe]:RLEVel?
- DISPlay:WINDow[1|2]:TRACe:Y[:SCALe]:RPOSition
- DISPlay:WINDow[1|2]:TRACe:Y[:SCALe]:RPOSition?
- FORMat:BORDer
- FORMat:BORDer?
- FORMat[:DATA]
- FORMat[:DATA]?
- HCOPy:ABORt
- HCOPy:DEVice[1|2|3]:COLor
- HCOPy:DEVice[1|2|3]:COLor?
- HCOPy:DEVice[1|2|3]:LANGuage
- HCOPy:DEVice[1|2|3]:LANGuage?
- HCOPy:DEVice[1|2|3]:MODE
- HCOPy:DEVice[1|2|3]:MODE?
- HCOPy:DEVice[1|2|3]:RESolution
- HCOPy:DEVice[1|2|3]:RESolution?
- HCOPy[:IMMediate]
- HCOPy:ITEM:ANNotation:STATe
- HCOPy:ITEM:ANNotation:STATe?
- HCOPy:ITEM[1|2|3]:FFEed:STATe
- HCOPy:ITEM[1|2|3]:FFEed:STATe?
- INITiate[1|2]:CONTinuous
- INITiate[1|2]:CONTinuous?
- INITiate[1|2][:IMMediate]

- MMEMory:CATalog?
- MMEMory:CDIRectory
- MMEMory:CDIRectory?
- MMEMory:COPY
- MMEMory:DELete
- MMEMory:FILE:INFO?
- MMEMory:INITialize
- MMEMory:LOAD:STATe
- MMEMory:MOVE
- MMEMory:MSIS
- MMEMory:MSIS?
- MMEMory:STORe:STATe
- MMEMory:STORe:TRACe
- MMEMory:TRANsfer:BDAT
- MMEMory:TRANsfer[:HFS]
- OUTPut[:STATe]
- OUTPut[:STATe]?
- PROGram:CATalog?
- PROGram[:SELected]:DEFine
- PROGram[:SELected]:DEFine?
- PROGram[:SELected]:DELete:ALL
- PROGram[:SELected]:DELete[:SELected]
- PROGram[:SELected]:EXECute
- PROGram[:SELected]:MALLocate
- PROGram[:SELected]:MALLocate?
- PROGram[:SELected]:NAME
- PROGram[:SELected]:NAME?
- PROGram[:SELected]:NUMBer

- PROGram[:SELected]:NUMBer?
- PROGram[:SELected]:STATe
- PROGram[:SELected]:STATe?
- PROGram[:SELected]:STRing
- PROGram[:SELected]:STRing?
- PROGram[:SELected]:WAIT
- PROGram[:SELected]:WAIT?
- SENSe[1|2]:AVERage:COUNt
- SENSe[1|2]:AVERage:COUNt?
- SENSe[1|2]:AVERage[:STATe]
- SENSe[1|2]:AVERage[:STATe]?
- SENSe[1|2]:BWIDth[:RESolution]
- SENSe[1|2]:BWIDth[:RESolution]?
- SENSe[1|2]:CORRection:COLLect[:ACQuire]
- SENSe[1|2]:CORRection:COLLect:METHod
- SENSe[1|2]:CORRection:COLLect:SAVE
- SENSe[1|2]:CORRection:CSET[:SELect]
- SENSe[1|2]:CORRection:CSET[:SELect]?
- SENSe[1|2]:CORRection:EDELay:TIME
- SENSe[1|2]:CORRection:IMPedance:INPut:MAGNitude
- SENSe[1|2]:CORRection:OFFSet:PHASe
- SENSe[1|2]:CORRection:RVELocity:COAX
- SENSe[1|2]:CORRection[:STATe]
- SENSe[1|2]:CORRection[:STATe]?
- SENSe[1|2]:DETector[:FUNCTION]
- SENSe[1|2]:FREQuency:CENTer
- SENSe[1|2]:FREQuency:CENTer?
- SENSe[1|2]:FREQuency:SPAN

- SENSe[1|2]:FREQuency:SPAN?
- SENSe[1|2]:FREQuency:STARt
- SENSe[1|2]:FREQuency:STARt?
- SENSe[1|2]:FREQuency:STOP
- SENSe[1|2]:FREQuency:STOP?
- SENSe[1|2]:FUNCtion
- SENSe[1|2]:FUNCtion?
- SENSe:ROSCillator:SOURce
- SENSe:ROSCillator:SOURce?
- SENSe[1|2]:SWEep:POINts
- SENSe[1|2]:SWEep:POINts?
- SENSe[1|2]:SWEep:TIME
- SENSe[1|2]:SWEep:TIME?
- SENSe[1|2]:SWEep:TIME:AUTO
- SENSe[1|2]:SWEep:TIME:AUTO?
- SOURce[1|2]:POWer[:LEVel][:IMMediate][:AMPLitude]
- SOURce[1|2]:POWer[:LEVel][:IMMediate][:AMPLitude]?
- SOURce[1|2]:POWer:RANGe
- SOURce[1|2]:POWer:STARt
- SOURce[1|2]:POWer:STOP
- STATus:OPERation:CONDition?
- STATus:OPERation:ENABle
- STATus:OPERation:ENABle?
- STATus:OPERation[:EVENt]?
- STATus:OPERation:NTRansition
- STATus:OPERation:NTRansition?
- STATus:OPERation:PTRansition
- STATus:OPERation:PTRansition?

- STATus:QUEStionable:CONDition?
- STATus:QUEStionable:ENABle
- STATus:QUEStionable:ENABle?
- STATus:QUEStionable[:EVENt]?
- STATus:QUEStionable:NTRansition
- STATus:QUEStionable:NTRansition?
- STATus:QUEStionable:PTRansition
- STATus:QUEStionable:PTRansition?
- SYSTem:BEEPer[:IMMediate]?
- SYSTem:BEEPer:VOLume
- SYSTem:BEEPer:VOLume?
- SYSTem:COMMunicate:GPIB[:SELF]:ADDRess
- SYSTem:COMMunicate:GPIB[:SELF]:ADDRess?
- SYSTem:COMMunicate:SERial:TRANsmit:BAUD
- SYSTem:COMMunicate:SERial:TRANsmit:BAUD?
- SYSTem:DATE
- SYSTem:DATE?
- SYSTem:ERRor?
- SYSTem:KEY[:VALue]?
- SYSTem:PRESet
- SYSTem:SET
- SYSTem:SET:LRN?
- SYSTem:TIME
- SYSTem:TIME?
- SYSTem:VERSion?
- TRACe[:DATA]
- TRACe[:DATA]?
- TRIGger[:SEQuence]:SOURce
- TRIGger[:SEQuence]:SOURce?

Instrument Specific Commands

The following are instrument specific commands implemented by the 8712ET/ES and 8714ET/ES RF Network Analyzers which are not part of the present SCPI 1996.0 definition.

- CALCulate[1|2]:FORMat:UNIT:MLIN
- CALCulate[1|2]:FORMat:UNIT:MLIN?
- CALCulate[1|2]:FORMat:UNIT:MLOG
- CALCulate[1|2]:FORMat:UNIT:MLOG?
- CALCulate[1|2]:LIMit:DISPlay
- CALCulate[1|2]:LIMit:DISPlay?
- CALCulate[1|2]:LIMit:MARKer:FLATness:MAXimum
- CALCulate[1|2]:LIMit:MARKer:FLATness:MINimum
- CALCulate[1|2]:LIMit:MARKer:FLATness[:STATe]
- CALCulate[1|2]:LIMit:MARKer:FREQuency:MAXimum
- CALCulate[1|2]:LIMit:MARKer:FREQuency:MINimum
- CALCulate[1|2]:LIMit:MARKer:FREQuency[:STATe]
- CALCulate[1|2]:LIMit:MARKer:STATistic:MEAN:MAXimum
- CALCulate[1|2]:LIMit:MARKer:STATistic:MEAN:MINimum
- CALCulate[1|2]:LIMit:MARKer:STATistic:MEAN[:STATe]
- CALCulate[1|2]:LIMit:MARKer:STATistic:PEAK:MAXimum
- CALCulate[1|2]:LIMit:MARKer:STATistic:PEAK:MINimum
- CALCulate[1|2]:LIMit:MARKer:STATistic:PEAK[:STATe]
- CALCulate[1|2]:LIMit:MARKer:TILT:MAXimum
- CALCulate[1|2]:LIMit:MARKer:TILT:MINimum
- CALCulate[1|2]:LIMit:MARKer:TILT[:STATe]
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:AMPLitude:STARt
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:AMPLitude:STARt?
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:AMPLitude:STOP

- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:AMPLitude:STOP?
- CALCulate[1|2]:LIMit:SEGMent:AOFF
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:FREQuency:STARt
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:FREQuency:STARt?
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:FREQuency:STOP
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:FREQuency:STOP?
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:POWer:STOP
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:POWer:STOP?
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:STATe
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:STATe?
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:TYPE
- CALCulate[1|2]:LIMit:SEGMent[1|2|...12]:TYPE?
- CALCulate[1|2]:MARKer:AOFF
- CALCulate[1|2]:MARKer:BWIDth
- CALCulate[1|2]:MARKer:BWIDth?
- CALCulate[1|2]:MARKer:FUNCtion:RESult?
- CALCulate[1|2]:MARKer:FUNCtion[:SELect]
- CALCulate[1|2]:MARKer:FUNCtion[:SELect]?
- CALCulate[1|2]:MARKer:FUNCtion:TRACking
- CALCulate[1|2]:MARKer:FUNCtion:TRACking?
- CALCulate[1|2]:MARKer[1|2|...8]:GDELay?
- CALCulate[1|2]:MARKer[1|2|...8]:MAXimum
- CALCulate[1|2]:MARKer[1|2|...8]:MAXimum:LEFT
- CALCulate[1|2]:MARKer[1|2|...8]:MAXimum:RIGHt
- CALCulate[1|2]:MARKer[1|2|...8]:MINimum
- CALCulate[1|2]:MARKer[1|2|...8]:MINimum:LEFT
- CALCulate[1|2]:MARKer[1|2|...8]:MINimum:RIGHt
- CALCulate[1|2]:MARKer:MODE

- CALCulate[1|2]:MARKer:MODE?
- CALCulate[1|2]:MARKer:NOTCh
- CALCulate[1|2]:MARKer[1|2|...8]:POINt
- CALCulate[1|2]:MARKer[1|2|...8]:POINt?
- CALCulate[1|2]:MARKer:REFerence:X?
- CALCulate[1|2]:MARKer:REFerence:Y?
- CALCulate[1|2]:MARKer[1|2|...8][:STATe]
- CALCulate[1|2]:MARKer[1|2|...8][:STATe]?
- CALCulate[1|2]:MARKer[1|2|...8]:TARGet
- CALCulate[1|2]:MARKer[1|2|...8]:TARGet?
- CALCulate[1|2]:MARKer[1|2|...8]:X
- CALCulate[1|2]:MARKer[1|2|...8]:X?
- CALCulate[1|2]:MARKer[1|2|...8]:X:ABS
- CALCulate[1|2]:MARKer[1|2|...8]:Y?
- CALCulate[1|2]:MARKer[1|2|...8]:Y:INDuctance?
- CALCulate[1|2]:MARKer[1|2|...8]:Y:MAGNitude?
- CALCulate[1|2]:MARKer[1|2|...8]:Y:PHASe?
- CALCulate[1|2]:MARKer[1|2|...8]:Y:REACtance?
- CALCulate[1|2]:MARKer[1|2|...8]:Y:RESistance?
- CALibration:SELF
- CALibration:SELF:TIMER
- CALibration:SELF:ALL
- CONFigure
- CONFigure?
- CONTrol[1|2]:MULTiport:STATE
- DIAGnostic:CCONstants:INSTalled?
- DIAGnostic:CCONstants:LOAD
- DIAGnostic:CCONstants:STORe:DISK

- DIAGnostic:CCONstants:STORe:EEPRom
- DIAGnostic:COMMunicate:LAN:PING:IMM
- DIAGnostic:COMMunicate:LAN:PING:IPADress
- DIAGnostic:COMMunicate:LAN:SEND
- DIAGnostic:MDISplay[1|2]:CORRection C_DIRECT
- DIAGnostic:MDISplay[1|2]:CORRection C_ISOLATION
- DIAGnostic:MDISplay[1|2]:CORRection C_LDMATCH
- DIAGnostic:MDISplay[1|2]:CORRection C_RTRACKING
- DIAGnostic:MDISplay[1|2]:CORRection C_SRCMATCH
- DIAGnostic:MDISplay[1|2]:CORRection C_TTRACKING
- DIAGnostic:MDISplay[1|2]:CORRection I_DIRECTivity
- DIAGnostic:MDISplay[1|2]:CORRection I_RESPONSE
- DIAGnostic:MDISplay[1|2]:CORRection I_SRCMATCH
- DIAGnostic:MDISplay[1|2]:CORRection I_TRACKING
- DIAGnostic:MDISplay[1|2]:CORRection M_DIRECTivity
- DIAGnostic:MDISplay[1|2]:CORRection M_RESPONSE
- DIAGnostic:MDISplay[1|2]:CORRection M_SRCMATCH
- DIAGnostic:MDISplay[1|2]:CORRection M_TRACKING
- DIAGnostic:MDISplay[1|2]:CORRection M_XSCALAR
- DIAGnostic:MDISplay[1|2]:REST
- DIAGnostic:DITHer
- DIAGnostic:DITHer?
- DIAGnostic:SNUMber
- DIAGnostic:SNUMber?
- DIAGnostic:SPUR:AVOid
- DIAGnostic:SPUR:AVOid?
- DISPlay:ANNotation:CHANnel[1|2][:STATe]
- DISPlay:ANNotation:CHANnel[1|2]:USER:LABel[:DATA]

- DISPlay:ANNotation:CHANnel[1|2]:USER[:STATe]
- DISPlay:ANNotation:CLOCk:DATE:FORMat
- DISPlay:ANNotation:CLOCk:DATE:FORMat?
- DISPlay:ANNotation:CLOCk:DATE:MODE
- DISPlay:ANNotation:CLOCk:DATE:MODE?
- DISPlay:ANNotation:CLOCk:MODE
- DISPlay:ANNotation:CLOCk:MODE?
- DISPlay:ANNotation:CLOCk:SEConds[:STATe]
- DISPlay:ANNotation:CLOCk:SEConds[:STATe]?
- DISPlay:ANNotation:FREQuency[1|2]:MODE
- DISPlay:ANNotation:FREQuency[1|2]:MODE?
- DISPlay:ANNotation:FREQuency:RESolution
- DISPlay:ANNotation:FREQuency:RESolution?
- DISPlay:ANNotation:FREQuency[1|2][:STATe]
- DISPlay:ANNotation:FREQuency[1|2]:USER:LABel[:DATA]
- DISPlay:ANNotation:FREQuency[1|2]:USER:STARt
- DISPlay:ANNotation:FREQuency[1|2]:USER[:STATe]
- DISPlay:ANNotation:FREQuency[1|2]:USER:STOP
- DISPlay:ANNotation:FREQuency[1|2]:USER:SUFFIX
- DISPlay:ANNotation:LIMit:ICON[1|2]:FLAG
- DISPlay:ANNotation:LIMit:ICON[1|2]:POS:X
- DISPlay:ANNotation:LIMit:ICON[1|2]:POS:Y
- DISPlay:ANNotation:LIMit:ICON[1|2]:TEXT
- DISPlay:ANNotation:LIMit:ICON[1|2]:STATe
- DISPlay:ANNotation:MARKer[1|2]:NUMBers[:STATe]
- DISPlay:ANNotation:MARKer[1|2][:STATe]
- DISPlay:ANNotation:MARKer[1|2][:STATe]?
- DISPlay:ANNotation:MESSage:AOFF

- DISPlay:ANNotation:MESSage:CLEar
- DISPlay:ANNotation:MESSage[:DATA]?
- DISPlay:ANNotation:MESSage:STATe
- DISPlay:ANNotation:MESSage:STATe?
- DISPlay:ANNotation:TITLe[1|2]:DATA
- DISPlay:ANNotation:TITLe[1|2]:DATA?
- DISPlay:ANNotation:TITLe[:STATe]
- DISPlay:ANNotation:TITLe[:STATe]?
- DISPlay:ANNotation:YAXis:MODE
- DISPlay:ANNotation:YAXis:MODE?
- DISPlay:ANNotation:YAXis[:STATe]
- DISPlay:ANNotation:YAXis[:STATe]?
- DISPlay:CMAP:COLor[1|2|...16]:GREYscale
- DISPlay:CMAP:SCHeme
- DISPlay:FORMat
- DISPlay:FORMat?
- DISPlay:FORMat:EXPAND
- DISPlay:MENU:RECall:FAST[:STATe]
- DISPlay:PROGram[:MODE]
- DISPlay:PROGram[:MODE]?
- DISPlay:WINDow:GRAPhics:BUFFer[:STATe]
- DISPlay:WINDow:GRAPhics:BUFFer[:STATe]?
- DISPlay:WINDow[1|2|10]:GRAPhics:CIRCle
- DISPlay:WINDow[1|2|10]:GRAPhics:LABel:FONT
- DISPlay:WINDow[1|2|10]:GRAPhics:LABel:FONT?
- DISPlay:WINDow[1|2|10]:GRAPhics:RECTangle
- DISPlay:WINDow[1|2|10]:TRACe[1|2]:Y:TRACk
- HCOPy:DEVice:PAGE:MARGin:LEFT

- HCOPy:DEVice:PAGE:MARGin:TOP
- HCOPy:DEVice:PAGE:ORIentation
- HCOPy:DEVice:PAGE:WIDTh
- HCOPy:DEVice:PORT
- HCOPy:DEVice:PORT?
- HCOPy:ITEM:GRATicule:STATe
- HCOPy:ITEM:GRATicule:STATe?
- HCOPy:ITEM:MARKer:STATe
- HCOPy:ITEM:MARKer:STATe?
- HCOPy:ITEM:TITLe:STATe
- HCOPy:ITEM:TITLe:STATe?
- HCOPy:ITEM:TRACe:STATe
- HCOPy:ITEM:TRACe:STATe?
- HCOPy:PAGE:MARGin:LEFT
- HCOPy:PAGE:MARGin:LEFT?
- HCOPy:PAGE:MARGin:TOP
- HCOPy:PAGE:MARGin:TOP?
- HCOPy:PAGE:ORIentation
- HCOPy:PAGE:ORIentation?
- HCOPy:PAGE:WIDTh
- HCOPy:PAGE:WIDTh?
- INPut:GAIN:AUTO
- INPut:GAIN:SETTing
- MMEMory:MDIRectory
- MMEMory:RDIRectory
- MMEMory:STORe:STATe:CORRection
- MMEMory:STORe:STATe:CORRection?
- MMEMory:STORe:STATe:ISTate

- MMEMory:STORe:STATe:ISTate?
- MMEMory:STORe:STATe:TRACe
- MMEMory:STORe:STATe:TRACe?
- MMEMory:STORe:STATe:TSCAL
- MMEMory:STORe:TRACe
- MMEMory:STORe:TRACe:FORMat
- MMEMory:TRANsfer:BDAT
- MMEMory:TRANsfer[:HFS]
- POWer\[1|2]:MODE
- ROUTe[1|2]:REFLection:PATH:DEFine:PORT
- ROUTe[1|2]:TRANsmission:PATH:DEFine:PORT
- SENSe[1|2]:AVERage:CLEar
- SENSe[1|2]:CORRection:CAPacitance:CONNector **(Option 100 only)**
- SENSe[1|2]:CORRection:CAPacitance:CONNector? **(Option 100 only)**
- SENSe[1|2]:CORRection:COLLect:ABORt
- SENSe[1|2]:CORRection:COLLect:CKIT[:SELect]
- SENSe[1|2]:CORRection:COLLect:CKIT[:SELect]?
- SENSe[1|2]:CORRection:COLLect:ISTate[:AUTO]
- SENSe[1|2]:CORRection:COLLect:ISTate[:AUTO]?
- SENSe[1|2]:CORRection:COLLect:PORTS
- SENSe[1|2]:CORRection:COLLect:MP:OPEN
- SENSe[1|2]:CORRection:COLLect:MP:SHORT
- SENSe[1|2]:CORRection:COLLect:MP:LOAD
- SENSe[1|2]:CORRection:COLLect:MP:THRU
- SENSe[1|2]:CORRection:COLLect:VERify:TRANsmission
- SENSe[1|2]:CORRection:COLLect:VERify:REFLection
- SENSe[1|2]:CORRection:EXTension[:STATe]
- SENSe[1|2]:CORRection:EXTension:REFLection[:TIME]

- SENSe[1|2]:CORRection:EXTension:TRANsmission[:TIME]
- SENSe[1|2]:CORRection:IMPedance:INPut:MAGNitude:SELect
- SENSe[1|2]:CORRection:LENGth:COAX **(Option 100 only)**
- SENSe[1|2]:CORRection:LENGth:COAX? **(Option 100 only)**
- SENSe[1|2]:CORRection:LENGth:CONNector **(Option 100 only)**
- SENSe[1|2]:CORRection:LENGth:CONNector? **(Option 100 only)**
- SENSe[1|2]:CORRection:LOSS:COAX **(Option 100 only)**
- SENSe[1|2]:CORRection:LOSS:COAX? **(Option 100 only)**
- SENSe[1|2]:CORRection:MODel:CONNector[:IMMediate] **(Option 100 only)**
- SENSe[1|2]:CORRection:PEAK:COAX **(Option 100 only)**
- SENSe[1|2]:CORRection:PEAK:COAX? **(Option 100 only)**
- SENSe[1|2]:CORRection:RVELocity[:IMMediate] **(Option 100 only)**
- SENSe[1|2]:CORRection:TESTSET
- SENSe[1|2]:CORRection:THReshold:COAX **(Option 100 only)**
- SENSe[1|2]:CORRection:THReshold:COAX? **(Option 100 only)**
- SENSe:COUPle
- SENSe:COUPle?
- SENSe[1|2]:DETector[:FUNCtion]
- SENSe[1|2]:DETector[:FUNCtion]?
- SENSe:DISTance:STARt **(Option 100 only)**
- SENSe:DISTance:STARt? **(Option 100 only)**
- SENSe:DISTance:STOP **(Option 100 only)**
- SENSe:DISTance:STOP? **(Option 100 only)**
- SENSe:DISTance:UNITs **(Option 100 only)**
- SENSe:DISTance:UNITs? **(Option 100 only)**
- SENSe:FREQuency:MODE **(Option 100 only)**
- SENSe:FREQuency:MODE? **(Option 100 only)**
- SENSe:FREQuency:SPAN:MAXimum? **(Option 100 only)**

- SENSe:FREQuency:SPAN:MAXimum **(Option 100 only)**
- SENSe:FREQuency:ZSTop **(Option 100 only)**
- SENSe:FREQuency:ZSTop? **(Option 100 only)**
- SENSe:FUNCtion:SRL:IMPedance **(Option 100 only)**
- SENSe:FUNCtion:SRL:IMPedance? **(Option 100 only)**
- SENSe:FUNCtion:SRL:MODE **(Option 100 only)**
- SENSe:FUNCtion:SRL:MODE? **(Option 100 only)**
- SENSe:FUNCtion:SRL:SCAN[:IMMediate] **(Option 100 only)**
- SENSe[1|2]:STATe
- SENSe[1|2]:STATe?
- SENSe:SWEep:TRIGger:SOURce
- SENSe:SWEep:TRIGger:SOURce?
- SENSe:WINDow[:TYPE] **(Option 100 only)**
- SENSe:WINDow[:TYPE]? **(Option 100 only)**
- STATus:DEVice:CONDition?
- STATus:DEVice:ENABle
- STATus:DEVice:ENABle?
- STATus:DEVice[:EVENt]?
- STATus:DEVice:NTRansition
- STATus:DEVice:NTRansition?
- STATus:DEVice:PTRansition
- STATus:DEVice:PTRansition?
- STATus:OPERation:AVERaging:CONDition?
- STATus:OPERation:AVERaging:ENABle
- STATus:OPERation:AVERaging:ENABle?
- STATus:OPERation:AVERaging[:EVENt]?
- STATus:OPERation:AVERaging:NTRansition
- STATus:OPERation:AVERaging:NTRansition?

- STATus:OPERation:AVERaging:PTRansition
- STATus:OPERation:AVERaging:PTRansition?
- STATus:OPERation:MEASuring:CONDition?
- STATus:OPERation:MEASuring:ENABle
- STATus:OPERation:MEASuring:ENABle?
- STATus:OPERation:MEASuring[:EVENt]?
- STATus:OPERation:MEASuring:NTRansition
- STATus:OPERation:MEASuring:NTRansition?
- STATus:OPERation:MEASuring:PTRansition
- STATus:OPERation:MEASuring:PTRansition?
- STATus:PRESet
- STATus:QUEStionable:LIMit:CONDition?
- STATus:QUEStionable:LIMit:ENABle
- STATus:QUEStionable:LIMit:ENABle?
- STATus:QUEStionable:LIMit[:EVENt]?
- STATus:QUEStionable:LIMit:NTRansition
- STATus:QUEStionable:LIMit:NTRansition?
- STATus:QUEStionable:LIMit:PTRansition
- STATus:QUEStionable:LIMit:PTRansition?
- SYSTem:COMMunicate:GPIB:CONTroller[:STATe]
- SYSTem:COMMunicate:GPIB:CONTroller[:STATe]?
- SYSTem:COMMunicate:GPIB:ECHO
- SYSTem:COMMunicate:GPIB:ECHO?
- SYSTem:COMMunicate:GPIB:HCOPy:ADDRess
- SYSTem:COMMunicate:GPIB:HCOPy:ADDRess?
- SYSTem:COMMunicate:GPIB:MMEMory:ADDRess
- SYSTem:COMMunicate:GPIB:MMEMory:ADDRess?
- SYSTem:COMMunicate:GPIB:MMEMory:UNIT

- SYSTem:COMMunicate:GPIB:MMEMory:UNIT?
- SYSTem:COMMunicate:GPIB:MMEMory:VOLume
- SYSTem:COMMunicate:GPIB:MMEMory:VOLume?
- SYSTem:COMMunicate:GPIB:MMEMory:VOLume?
- SYSTem:COMMunicate:LAN:EADDress?
- SYSTem:COMMunicate:LAN:IPADdress
- SYSTem:COMMunicate:LAN:IPADdress?
- SYSTem:COMMunicate:LAN:PRINter:HOSTname
- SYSTem:COMMunicate:LAN:PRINter:HOSTname?
- SYSTem:COMMunicate:LAN:ROUTe:GATeway
- SYSTem:COMMunicate:LAN:ROUTe:GATeway?
- SYSTem:COMMunicate:LAN:ROUTe:SMASk
- SYSTem:COMMunicate:LAN:ROUTe:SMASk?
- SYSTem:COMMunicate:LAN:STATe
- SYSTem:COMMunicate:LAN:STATe?
- SYSTem:COMMunicate:SERial:TRANsmit:HANDshake
- SYSTem:COMMunicate:SERial:TRANsmit:HANDshake?
- SYSTem:COMMunicate:TTL:USER:FEED
- SYSTem:COMMunicate:TTL:USER:FEED?
- SYSTem:KEY:MASK?
- SYSTem:KEY:QUEue:CLEar
- SYSTem:KEY:QUEue:COUNt?
- SYSTem:KEY:QUEue:MAXimum?
- SYSTem:KEY:QUEue[:STATe]
- SYSTem:KEY:QUEue[:STATe]?
- SYSTem:KEY:TYPE?
- SYSTem:KEY:USER
- SYSTem:SET:LRNLong

- `TEST:RESult?`
- `TEST:SELect`
- `TEST:SELect?`
- `TEST:STATe`
- `TEST:STATe?`
- `TEST:VALue`
- `TEST:VALue?`

13 SCPI Error Messages

SCPI Error Messages

This chapter contains the same error message information that can be found in the *SCPI 1994 Volume 2: Command Reference*. There are four sections in this chapter:

- "Command Errors" on page 13-3
- "Execution Errors" on page 13-8
- "Device-Specific Errors" on page 13-15
- "Query Errors" on page 13-17

NOTE Your analyzer does not use all of the error messages listed in this chapter.

Command Errors

An error/event number in the range –199 to –100 indicates that an IEEE 488.2 syntax error has been detected by the instrument's parser. The occurrence of any error in this class shall cause the command error bit (bit 5) in the event status register (IEEE 488.2, section 11.5.1) to be set. One of the following events has occurred:

- An IEEE 488.2 syntax error has been detected by the parser. That is, a controller-to-device message was received which is in violation of the IEEE 488.2 standard. Possible violations include a data element which violates the device listening formats or whose type is unacceptable to the device.
- An unrecognized header was received. Unrecognized headers include incorrect device-specific headers and incorrect or unimplemented IEEE 488.2 common commands.
- A Group Execute Trigger (GET) was entered into the input buffer inside of an IEEE 488.2 program message.

Events that generate command errors shall not generate execution errors, device-specific errors, or query errors; see the other error definitions in this chapter.

Table 13-1 SCPI Command Errors

Error Number	Error Description
–100	Command error — This is the generic syntax error for devices that cannot detect more specific errors. This code indicates only that a Command Error has occurred.
–101	Invalid character — A syntactic element contains a character which is invalid for that type; for example, a header containing an ampersand, SETUP&. This error might be used in place of errors –114, –121, –141, and perhaps some others.
–102	Syntax error — An unrecognized command or data type was encountered; for example, a string was received when the device does not accept strings.
–103	Invalid separator — The parser was expecting a separator and encountered an illegal character; for example, the semicolon was omitted after a program message unit, `*EMC 1:CH1:VOLTS 5`.
–104	Data type error — The parser recognized a data element different than one allowed; for example, numeric or string data was expected but block data was encountered.
–105	GET not allowed — A Group Execute Trigger was received within a program message.
–108	Parameter not allowed — More parameters were received than expected for the header; for example, the `*EMC` common command only accepts one parameter, so receiving `*EMC 0,1` is not allowed.
–109	Missing parameter — Fewer parameters were received than required for the header; for example, the `*EMC` common command requires one parameter, so receiving `*EMC` is not allowed.
–110	Command header error — An error was detected in the header. This error message should be used when the device cannot detect the more specific errors described for errors –111 through –119.
–111	Header separator error — A character which is not a legal header separator was encountered while parsing the header; for example, no white space followed the header, thus `*GMC"MACRO"` is an error.

Error Number	Error Description
–112	Program mnemonic too long — The header contains more that twelve characters.
–113	Undefined header — The header is syntactically correct, but it is undefined for this specific device; for example, *XYZ is not defined for any device.
–114	Header suffix out of range — The value of a numeric suffix attached to a program mnemonic makes the header invalid.
–120	Numeric data error — This error, as well as errors –121 through –129, are generated when parsing a data element which appears to be numeric, including the nondecimal numeric types. This particular error message should be used if the device cannot detect a more specific error.
–121	Invalid character in number — An invalid character for the data type being parsed was encountered; for example, an alpha in a decimal numeric or a "9" in octal data.
–123	Exponent too large — The magnitude of the exponent was larger than 32000.
–124	Too many digits — The mantissa of a decimal numeric data element contained more than 255 digits excluding leading zeros.
–128	Numeric data not allowed — A legal numeric data element was received, but the device does not accept one in this position for the header.
–130	Suffix error — This error, as well as errors –131 through –139, are generated when parsing a suffix. This particular error message should be used if the device cannot detect a more specific error.
–131	Invalid suffix — The suffix does not follow the correct syntax, or the suffix is inappropriate for this device.
–134	Suffix too long — The suffix contained more than 12 characters.
–138	Suffix not allowed — A suffix was encountered after a numeric element which does not allow suffixes.
–140	Character data error — This error, as well as errors –141 through –149, are generated when parsing a character data element. This particular error message should be used if the device cannot detect a more specific error.

Error Number	Error Description
–141	Invalid character data — Either the character data element contains an invalid character or the particular element received is not valid for the header.
–144	Character data too long — The character data element contains more than twelve characters.
–148	Character data not allowed — A legal character data element was encountered where prohibited by the device.
–150	String data error — This error, as well as errors –151 through –159, are generated when parsing a string data element. This particular error message should be used if the device cannot detect a more specific error.
–151	Invalid string data — A string data element was expected, but was invalid for some reason. For example, an END message was received before the terminal quote character.
–158	String data not allowed — A string data element was encountered but was not allowed by the device at this point in parsing.
–160	Block data error — This error, as well as errors –161 through –169, are generated when parsing a block data element. This particular error message should be used if the device cannot detect a more specific error.
–161	Invalid block data — A block data element was expected, but was invalid for some reason. For example, an END message was received before the length was satisfied.
–168	Block data not allowed — A legal block data element was encountered but was not allowed by the device at this point in parsing.
–170	Expression error — This error, as well as errors –171 through –179, are generated when parsing an expression data element. This particular error message should be used if the device cannot detect a more specific error.
–171	Invalid expression — The expression data element was invalid (for example, unmatched parentheses or an illegal character).
–178	Expression data not allowed — A legal expression data was encountered but was not allowed by the device at this point in parsing.

Error Number	Error Description
–180	Macro error — This error, as well as errors –181 through –189, are generated when defining or executing a macro. This particular error message should be used if the device cannot detect a more specific error.
–181	Invalid outside macro definition — Indicates that a macro parameter placeholder ($<number>) was encountered outside of a macro definition.
–183	Invalid inside macro definition — Indicates that the program message unit sequence, sent with a `*DDT` or `*DMC` command, is syntactically invalid.
–184	Macro parameter error — Indicates that a command inside the macro definition had the wrong number or type of parameters.

Execution Errors

An error/event number in the range –299 to –200 indicates that an error has been detected by the instrument's execution control block. The occurrence of any error in this class shall cause the execution error bit (bit 4) in the event status register to be set. One of the following events has occurred:

- A program data element following a header was evaluated by the device as outside of its legal input range or is otherwise inconsistent with the device's capabilities.
- A valid program message could not be properly executed due to some device condition.

Execution errors shall be reported by the device after rounding and expression evaluation operations have taken place. Rounding a numeric data element, for example, shall not be reported as an execution error. Events that generate execution errors shall not generate Command Errors, device-specific errors, or Query Errors; see the other error definitions in this section.

Table 13-2 SCPI Execution Errors

Error Number	Error Description
–200	Execution error — This is the generic syntax error for devices that cannot detect more specific errors. This code indicates only that an Execution Error has occurred.
–201	Invalid while in local — Indicates that a command is not executable while the device is in local due to a hard local control; for example, a device with a rotary switch receives a message which would change the switches state, but the device is in local so the message can not be executed.
–202	Settings lost due to rtl — Indicates that a setting associated with a hard local control was lost when the device changed to LOCS from REMS or to LWLS from RWLS.
–203	Command protected — Indicates that a legal password-protected program command or query could not be executed because the command was disabled.
–210	Trigger error
–211	Trigger ignored — Indicates that a `GET`, `*TRG`, or triggering signal was received and recognized by the device but was ignored because of device timing considerations; for example, the device was not ready to respond. [1]
–212	Arm ignored — Indicates that an arming signal was received and recognized by the device but was ignored.
–213	Init ignored — Indicates that a request for a measurement initiation was ignored as another measurement was already in progress.
–214	Trigger deadlock — Indicates that the trigger source for the initiation of a measurement is set to `GET` and subsequent measurement query is received. The measurement cannot be started until a `GET` is received, but the `GET` would cause an `INTERRUPTED` error.
–215s	Arm deadlock — Indicates that the arm source for the initiation of a measurement is set to `GET` and subsequent measurement query is received. The measurement cannot be started until a `GET` is received, but the `GET` would cause an `INTERRUPTED` error.

Error Number	Error Description
–220	Parameter error — Indicates that a program data element related error occurred. This error message should be used when the device cannot detect the more specific errors –221 through –229.
–221	Settings conflict — Indicates that a legal program data element was parsed but could not be executed due to the current device state.
–222	Data out of range — Indicates that a legal program data element was parsed but could not be executed because the interpreted value was outside the legal range as defined by the device.
–223	Too much data — Indicates that a legal program data element of block, expression, or string type was received that contained more data than the device could handle due to memory or related device-specific requirements.
–224	Illegal parameter value — Used where an exact value, from a list of possible values, was expected.
–225	Out of memory — The device has insufficient memory to perform the requested operation.
–226	Lists not same length — Attempted to use LIST structure having individual LIST's of unequal lengths.
–230	Data corrupt or stale — Possibly invalid data; new reading started but not completed since last access.
–231	Data questionable — Indicates that measurement accuracy is suspect.
–232	Invalid format — Indicates that a legal program data element was parsed but could not be executed because the data format or structure is inappropriate, such as when loading memory tables or when sending a `SYSTem:SET` parameter from an unknown instrument.
–233	Invalid version — Indicates that a legal program data element was parsed but could not be executed because the version of the data is incorrect to the device. This particular error should be used when file or block data formats are recognized by the instrument but cannot be executed for reasons of version incompatibility. For example, an unsupported file version, or an unsupported instrument version.

Error Number	Error Description
–240	Hardware error — Indicates that a legal program command or query could not be executed because of a hardware problem in the device. Definition of what constitutes a hardware problem is completely device-specific. This error message should be used when the device cannot detect the more specific errors described for errors –241 through –249.
–241	Hardware missing — Indicates that a legal program command or query could not be executed because of missing device hardware; for example, an option was not installed. Definition of what constitutes missing hardware is completely device-specific.
–250	Mass storage error — Indicates that a mass storage error occurred. This error message should be used when the device cannot detect the more specific errors described for errors –251 through –259.
–251	Missing mass storage — Indicates that a legal program command or query could not be executed because of missing mass storage; for example, an option that was not installed. Definition of what constitutes missing massstorage is device-specific.
–252	Missing media — Indicates that a legal program command or query could not be executed because of a missing media; for example, no disk. The definition of what constitutes missing media is device-specific.
–253	Corrupt media — Indicates that a legal program command or query could not be executed because of corrupt media; for example, bad disk or wrong format. The definition of what constitutes corrupt media is device-specific.
–254	Media full — Indicates that a legal program command or query could not be executed because the media was full; for example, there is no room on the disk. The definition of what constitutes a full media is device-specific.
–255	Directory full — Indicates that a legal program command or query could not be executed because the media directory was full. The definition of what constitutes a full media directory is device-specific.
–256	File name not found — Indicates that a legal program command or query could not be executed because the file name on the device media was not found; for example, an attempt was made to read or copy a nonexistent file. The definition of what constitutes a file not being found is device-specific.

Error Number	Error Description
–257	File name error — Indicates that a legal program command or query could not be executed because the file name on the device media was in error; for example, an attempt was made to copy to a duplicate file name. The definition of what constitutes a file name error is device-specific.
–258	Media protected — Indicates that a legal program command or query could not be executed because the media was protected; for example, the write-protect tab on a disk was present. The definition of what constitutes protected media is device-specific.
–260	Expression error — Indicates that an expression program data element related error occurred. This error message should be used when the device cannot detect the more specific errors described for errors –261 through –269.
–261	Math error in expression — Indicates that a syntactically legal expression program data element could not be executed due to a math error; for example, a divide-by-zero was attempted. The definition of math error is device-specific.
–270	Macro error — Indicates that a macro-related execution error occurred. This error message should be used when the device cannot detect the more specific errors –271 through –279.
–271	Macro syntax error — Indicates that a syntactically legal macro program data sequence could not be executed due to a syntax error within the macro definition.
–272	Macro execution error — Indicates that a syntactically legal macro program data sequence could not be executed due to some error in the macro definition.
–273	Illegal macro label — Indicates that the macro label defined in the `*DMC` command was a legal string syntax, but could not be accepted by the device; for example, the label was too long, the same as a common command header, or contained invalid header syntax.
–274	Macro parameter error — Indicates that the macro definition improperly used a macro parameter placeholder.

Error Number	Error Description
–275	Macro definition too long — Indicates that a syntactically legal macro program data sequence could not be executed because the string or block contents were too long for the device to handle.
–276	Macro recursion error — Indicates that a syntactically legal macro program data sequence could not be executed because the device found it to be recursive.
–277	Macro redefinition not allowed — Indicates that a syntactically legal macro label in the `*DMC` command could not be executed because the macro label was already defined.
–278	Macro header not found — Indicates that a syntactically legal macro label in the `*GMC?` query could not be executed because the header was not previously defined.
–280	Program error — Indicates that a downloaded program-related execution error occurred. This error message should be used when the device cannot detect the more specific errors –281 through –289. A downloaded program is used to add algorithmic capability to a device. The syntax used in the program and the mechanism for downloading a program is device-specific.
–281	Cannot create program — Indicates that an attempt to create a program was unsuccessful. One reason for failure might include not enough memory.
–282	Illegal program name — The name used to reference a program was invalid; for example, redefining an existing program, deleting a nonexistent program, or in general, referencing a nonexistent program.
–283	Illegal variable name — An attempt was made to reference a nonexistent variable in a program.
–284	Program currently running — Certain operations dealing with programs may be illegal while the program is running; for example, deleting a running program might not be possible.
–285	Program syntax error — Indicates that a syntax error appears in a downloaded program. The syntax used when parsing the downloaded program is device-specific.
–286	Program runtime error

Error Number	Error Description
–290	Memory use error — Indicates that a user request has directly or indirectly caused an error related to memory or data_handles (this is not the same as "bad" memory).
–291	Out of memory
–292	Referenced name does not exist
–293	Referenced name already exists
–294	Incompatible type — Indicates that the type or structure of a memory item is inadequate.

1. A DT0 device always ignores GET and treats *TRG as a Command Error.

Device-Specific Errors

An error/event number in the range –399 to –300 or 1 to 32767 indicates that the instrument has detected an error which is not a command error, a query error, or an execution error. It indicates that some device operations did not properly complete, possibly due to an abnormal hardware or firmware condition. These codes are also used for self-test response errors. The occurrence of any error in this class should cause the device-specific error bit (bit 3) in the event status register to be set.

The meaning of positive error codes is device-dependent and may be enumerated or bit mapped; the error message string for positive error codes is not defined by SCPI and available to the device designer. Note that the string is not optional; if the designer does not wish to implement a string for a particular error, the null string should be sent (for example, 42). The occurrence of any error in this class should cause the device-specific error bit (bit 3) in the event status register to be set. Events that generate device-specific errors shall not generate command errors, execution errors, or query errors; see the other error definitions in this section.

Table 13-3 SCPI Device-Specific Errors

Error Number	Error Description
–300	Device-specific error — This is the generic device-dependent error for devices that cannot detect more specific errors. This code indicates only that a Device-Dependent Error has occurred.
–310	System error — Indicates that some error, termed "system error" by the device, has occurred. This code is device-dependent.
–311	Memory error — Indicates that an error was detected in the device's memory. The scope of this error is device-dependent.
–312	PUD memory lost — Indicates that the protected user data saved by the `*PUD` command has been lost.
–313	Calibration memory lost — Indicates that nonvolatile calibration data used by the `*CAL?` command has been lost.
–314	Save/recall memory lost — Indicates that the nonvolatile data saved by the `*SAV?` command has been lost.
–315	Configuration memory lost — Indicates that nonvolatile configuration data saved by the device has been lost. The meaning of this error is device-specific.
–330	Self-test failed.
–350	Queue overflow — A specific code entered into the queue in lieu of the code that caused the error. This code indicates that there is no room in the queue and an error occurred but was not recorded.
–360	Communication error — This is the generic communication error for devices that cannot detect the more specific errors –361 through –363.
–361	Parity error in program message — Parity bit not correct when data received, for example, on a serial port.
–362	Framing error in program message — A stop bit was not detected when data was received, for example, on a serial port (for example, a baud rate mismatch).
–363	Input buffer overrun — Software or hardware input buffer on serial port overflows with data caused by improper or nonexistent pacing.

Query Errors

An error/event number in the range –499 to –400 indicates that the output queue control of the instrument has detected a problem with the message exchange protocol. The occurrence of any error in this class shall cause the query error bit (bit 2) in the event status register to be set. These errors correspond to message exchange protocol errors. One of the following is true:

- An attempt is being made to read data from the output queue when no output is either present or pending;
- Data in the output queue has been lost.

Events that generate query errors shall not generate command errors, execution errors, or device-specific errors; see the other error definitions in this section.

Table 13-4 **SCPI Query Errors**

Error Number	Error Description
–400	Query error — This is the generic query error for devices that cannot detect more specific errors. This code indicates only that a Query Error has occurred.
–410	Query `INTERRUPTED` — Indicates that a condition causing an `INTERRUPTED` Query error occurred; for example, a query followed by `DAB` or `GET` before a response was completely sent.
–420	Query `UNTERMINATED` — Indicates that a condition causing an `UNTERMINATED` Query error occurred; for example, the device was addressed to talk and an incomplete program message was received.
–430	Query `DEADLOCKED` — Indicates that a condition causing a `DEADLOCKED` Query error occurred; for example, both input buffer and output buffer are full and the device cannot continue.
–440	Query `UNTERMINATED` after indefinite response — Indicates that a query was received in the same program message after an query requesting an indefinite response was executed.

Symbols

Numerics

A

B

C

D

M

S