Errata

Title & Document Type: 35665A Dynamic Signal Analyzer HP-IB Programming Manual

Manual Part Number: 35665-90004

Revision Date: May 1991

HP References in this Manual

This manual may contain references to HP or Hewlett-Packard. Please note that Hewlett-Packard's former test and measurement, semiconductor products and chemical analysis businesses are now part of Agilent Technologies. We have made no changes to this manual copy. The HP XXXX referred to in this document is now the Agilent XXXX. For example, model number HP8648A is now model number Agilent 8648A.

About this Manual

We've added this manual to the Agilent website in an effort to help you support your product. This manual provides the best information we could find. It may be incomplete or contain dated information, and the scan quality may not be ideal. If we find a better copy in the future, we will add it to the Agilent website.

Support for Your Product

Agilent no longer sells or supports this product. You will find any other available product information on the Agilent Test & Measurement website:

www.tm.agilent.com

Search for the model number of this product, and the resulting product page will guide you to any available information. Our service centers may be able to perform calibration if no repair parts are needed, but no other support from Agilent is available.

HP 35665A Dynamic Signal Analyzer

HP-IB Programming with the HP 35665A

HEWLETT PACKARD

HP Part Number 35665-90004
Microfiche Part Number 35665-90204
Printed in U.S.A.

Print Date: May 1991

8600 Soper Hill Road Everett, Washington 98205-1298 U.S.A.

Guide to HP 35665A Documentation

If you are thinking about...	And you want to...	Then read...
♦ Unpack and install the HP 35665A	Install the HP 35665A Dynamic Signal Analyzer	*HP 35665A Installation and Verification Guide*
	Do operation verification or performance verification tests	*HP 35665A Installation and Verification Guide*
♦ Getting started	Make your first measurements with your new analyzer	*HP 35665A Quick Start Guide*
	Review measurement basics	*HP 35665A Concepts Guide*
	Learn what each key does	*HP 35665A Operator's Reference* or use the analyzer's **[Help]** key
♦ Making measurements	Learn how to make typical measurements with the HP 35665A	*HP 35665A Operator's Guide*
	Understand each of the analyzer's instrument modes	*HP 35665A Concepts Guide*
♦ Creating automated measurements	Learn the HP Instrument BASIC interface	*Using HP Instrument BASIC with the HP 35665A*
	Record keystrokes for a particular measurement	*HP 35665A Operator's Guide*
	Program with HP Instrument BASIC	*HP Instrument BASIC User's Handbook*
♦ Remote operation	Learn about the HP-IB	*HP–IB Programming with the HP 35665A*
	Learn how to program with HP-IB	*HP–IB Programming with the HP 35665A*
	Find specific HP-IB commands	*HP 35665A HP-IB Commands: Quick Reference*
♦ Servicing the analyzer	Adjust, troubleshoot, or repair the analyzer	*HP 35665A Service Guide*

Table of Content

Chapter 1: Introduction to HP-IB

Chapter 2: Behavior in an HP-IB System

Chapter 3: Programming with HP-IB Commands

Chapter 4: Transferring Data

Chapter 5: Using Status Registers

Chapter 6: Example Programs

Chapter 7: Introduction to the Command Reference

Chapter 8: Common Commands

Chapter 9: ABORt

Chapter 10: ARM

Chapter 11: CALCulate

Chapter 12: CALibration

Table of Content (Continued)

Chapter 29: TRIGger

Appendix A: HP 35665A Command Summary

Appendix B: Cross-Reference from Front-Panel Keys to HP-IB Commands

Appendix C: Error Messages

Appendix D: Instrument Modes

Appendix E: Determining Units

Index

1

Introduction to HP-IB

Notice to Experienced HP-IB Programmers

The HP 35665A's HP-IB command set is derived from the Standard Commands for Programmable Instruments—SCPI. The SCPI command set which is described later in this chapter, differs from the traditional HP-IB command set in the following ways:

- A traditional HP-IB command typically consists of a single mnemonic. A SCPI command typically consists of a series of mnemonics separated by colons. The mnemonics are selected from a command hierarchy, which organizes commands into related groups. These multi-mnemonic commands are less cryptic than single-mnemonic commands. They can help you make your programs self-documenting. Chapter 3 tells you how to use the SCPI command hierarchy.

- A traditional HP-IB command set contains mnemonics that correspond directly to an instrument's front-panel keys. The HP 35665A's command set gives you HP-IB access to all front-panel functions. However there is no one-to-one correspondence between the SCPI command tree and the softkey menu tree. This results from the fact that SCPI command hierarchy is organized differently than the front-panel key hierarchy. A special feature allows the HP 35665A to echo equivalent HP-IB command mnemonics when you press a series of front-panel keys. You can enable this feature under the [**Local/HP-IB**] hardkey. Appendix B cross-references HP 35665A hardkeys and softkeys to their equivalent HP-IB commands.

Manual Overview

This manual is organized into six major parts:

- Programming Fundamentals.
- Example Programs.
- Introduction to the Command Reference.
- Command Reference.
- Appendixes.
- Index.

Programming Fundamentals

This part of the manual contains five chapters, each of which discusses some aspect of programming the HP 35665A via the HP-IB:

- Chapter 1 introduces you to HP-IB and SCPI concepts. It also tells you how to configure the HP 35665A in an HP-IB system.
- Chapter 2 tells you how the analyzer interacts with the controller and other devices on the HP-IB.
- Chapter 3 describes SCPI command hierarchy.
- Chapter 4 tells you how data is transferred between the analyzer and a controller.
- Chapter 5 describes the analyzer's register structure and tells you how the analyzer uses registers to generate service requests.

Example Programs

This part of the manual lists example programs for the HP 35665A. It is a good place to start if you are an experienced HP-IB programmer and are already familiar with SCPI concepts. These example programs are available on the HP 35665A Example Programs Disk which is included with HP Instrument BASIC (Option 1C2).

Introduction to the Command Reference

This chapter describes the conventions and syntax descriptions used in the Command Reference chapters. It includes a section on finding commands.

Command Reference

This part of the manual contains a detailed description of each HP-IB command. The command descriptions are organized alphabetically.

Appendixes

This part of the manual contains five appendixes:

- Appendix A provides a quick reference to the HP 35665A's HP-IB command set.
- Appendix B provides a cross reference of the HP 35665A's hardkeys and softkeys and their equivalent HP-IB commands.
- Appendix C provides a complete listing of the HP 35665A's error messages.
- Appendix D provides a list of valid HP-IB commands for each of the major instrument modes.
- Appendix E explains how to determine the Y-axis units you send with certain commands.

Index

This part of the manual references the page numbers where different subjects are discussed. It can be especially useful for determining which command you should use to access a particular analyzer function.

Additional Help

Included in the documentation set is the "HP 35665A HP-IB Commands: Quick Reference." This card provides quick and convenient access to command syntax and structure.

HP-IB Overview

What is HP-IB?

HP-IB—the Hewlett-Packard Interface Bus—is a high-performance bus that allows you to build integrated test systems from individual instruments and computers. The bus and its associated interface operations are defined by the IEEE 488.1 standard. This standard is described later in this chapter.

HP-IB cables provide the physical link between devices on the bus. There are eight data lines on each cable that are used to send data from one device to another. Devices that can be addressed to send data over these lines are called "talkers," and those that can be addressed to receive data are called "listeners." 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. Devices that can use these control lines to specify the talker and listener in a data exchange are called "controllers."

When an HP-IB system contains more than 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 system controller is the one device that can take control of the bus even if it is not the active controller. The HP 35665A can act as a talker, listener, active controller, or system controller at different times.

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

Sending Commands Over the HP-IB

Commands are sent over the HP-IB via a controller's language system, such as BASIC, C or Pascal. As a result, you need to determine which keywords your controller's language system uses to send HP-IB commands. When looking for keywords, keep in mind that there are two different kinds of HP-IB commands:

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

Language systems usually deal differently with these two kinds of HP-IB commands. For example, HP BASIC uses a unique keyword to send each bus management command, but always uses the keyword OUTPUT to send device commands. For more information on the differences between bus management commands and device commands, see chapter 2, "Behavior in an HP-IB System."

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

```
OUTPUT 711;"DISPLAY:CONTENTS TRACE"
```

This sends the command within the quotes (DISPLAY:CONTENTS TRACE) to the HP-IB device at address 711. If the device is an HP 35665A, the command instructs the analyzer to display trace data.

Note Most of the examples in this manual are written for HP BASIC running on an HP Series 200/300 computer. There are two example programs in chapter 6 written in Microsoft® QuickC and Microsoft® Quick Basic.

Microsoft® is registered trademarks of Microsoft Corp.

SCPI Overview

What is SCPI?

SCPI—the Standard Commands for Programmable Instruments—is a programming language designed specifically for controlling instruments. It defines how you communicate with these instruments from an external controller (computer).

History

Computer-controlled test instruments that were introduced in the 1960s used a wide variety of non-standard interfaces and communication protocols. During this time, Hewlett-Packard developed the HP-IB as an internal standard. For connectors and cables, HP-IB defined an electrical and mechanical interface. For transmitting individual bytes of data between instruments and computers, it defined handshaking, addressing, and general protocol.

IEEE 488.1

The Institute of Electrical and Electronic Engineers (IEEE) in 1975, approved IEEE 488-1975, which was based on Hewlett-Packard's internal HP-IB standard. They updated this standard which is IEEE 488.1-1987. Hewlett-Packard uses HP-IB to indicate that an instrument or controller has conforms to the IEEE 488.1 standard.

Although it solved the problem of how to send bytes of data between instruments and computers, IEEE 488 did not specify the data bytes' meanings. Instrument manufacturers freely invented new commands as they developed new instruments. The format of data returned from instruments varied as well. By the early 1980s, work began on additional standards that specified how to interpret data sent via the 488 bus.

IEEE 488.2

In 1987, the IEEE approved IEEE 488.2-1987. This standard defined the interface capabilities of instruments and controllers in a measurement system connected by the 488 bus (HP-IB). In particular, IEEE 488.2 described how to send commands to instruments and how to send responses to controllers. Although it explicitly defined some frequently used commands, it still left the naming of most commands to instrument manufacturers. This made it possible for two similar instruments to conform to 488.2, yet have entirely different command sets.

SCPI

SCPI goes beyond 488.2 by defining a standard set of programming commands. For a given measurement function (such as frequency), SCPI defines the specific commands used to access that function via the 488 bus. If two analyzers both conform to the SCPI standard, for example, you would use the same command to set each analyzer's center frequency.

Standard commands provide two advantages:

- If you know how to control functions on one SCPI instrument, you know how to control the same functions on any SCPI instrument.
- Programs written for a particular SCPI instrument are easily adapted to work with a similar SCPI instrument.

Figure 1-1 shows you how SCPI builds on the 488 standards.

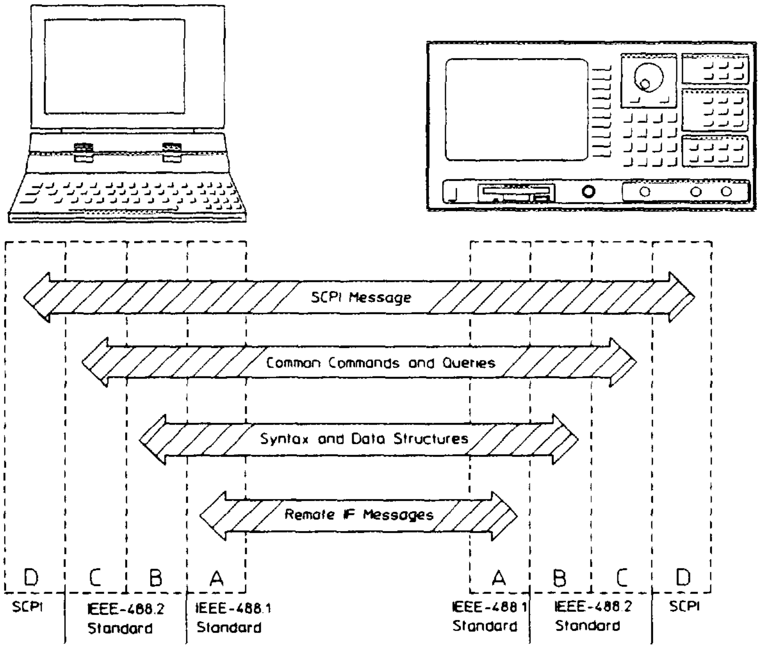


Figure 1-1. SCPI and Related Standards

The standards are layered to define different aspects of communication between devices:

- Layer A (IEEE 488.1) defines the physical and electrical connection between devices. It also defines how a byte of data is transmitted and how devices are instructed to talk and listen.
- Layer B (IEEE 488.2) defines the syntax and data formats used to send data between devices. It also defines the structure of status registers.
- Layer C (IEEE 488.2) defines the commands used for common tasks (such as resetting the device and reading the Status Byte).
- Layer D (SCPI) defines the commands used to control device-specific functions (such as setting frequency and amplitude). It also defines the parameters accepted by these functions and the values they return.

SCPI Compliance

Many of the HP 35665A's HP-IB commands comply to SCPI. The attribute summary in the command reference section identifies these commands as follows:

- Confirmed commands comply to SCPI 1990.
- Approved commands will be added to SCPI 1991.
- Instrument-specific commands do not comply to SCPI.

HP-IB Setup

This section contains a procedure for configuring the HP 35665A and an external controller in a simple HP-IB system. Although an HP 9836 computer is the controller used in the system, other computers that support an HP-IB interface can also be used. If you are using one of those other computers, the configuration procedure can only be used as a general guide. You should consult your computer's documentation for more complete information.

This section also contains a procedure for verifying that commands can be sent over the HP-IB. HP BASIC is used for the verification procedure's test program. If your computer uses some other language, the keywords and syntax for the test program may be different. You will need to write a similar program using your language's keywords and syntax.

Configuring the HP-IB System

Equipment and Software

- HP 35665A Dynamic Signal Analyzer
- HP 9836 computer
- HP 10833A, B, C, or D HP-IB Cable
- HP BASIC

Procedure

1. Turn off the HP 35665A and the HP 9836, then connect them with the HP-IB cable as shown in figure 1-2.

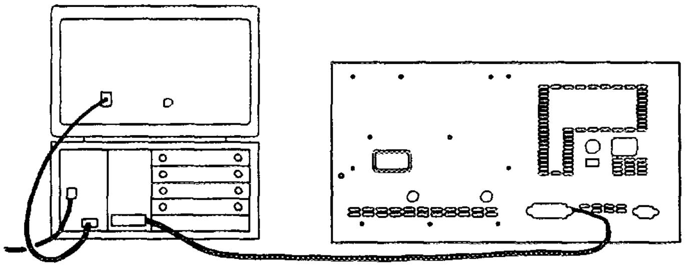


Figure 1-2. HP-IB Connections

2. Turn on the HP 9836. If necessary, load HP BASIC following the instructions in the computer's operating manual. Note that the following language extensions must be installed for the verification program to work:

- CRTA.
- HPIB.
- IO.
- EDIT.

   Programs that are more complex than the verification program will probably require more language extensions.

3. Turn on the HP 35665A. When the softkey labels appear, press the [**Local/HP-IB**] hardkey. See figure 1-3.

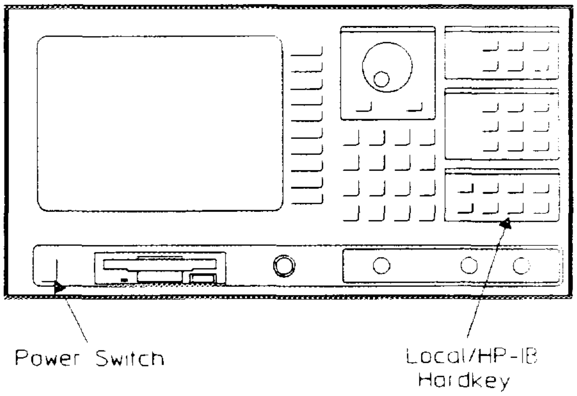
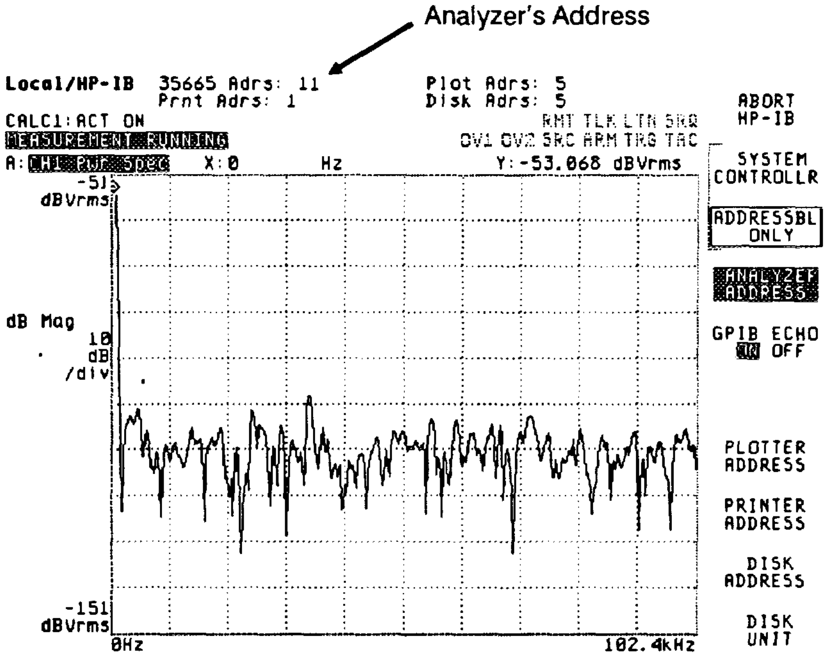


Figure 1-3. HP 35665A Front Panel

4. Verify that the analyzer's address is set to 11. The current address setting is displayed when you press the [ANALYZER ADDRESS] softkey (see figure 1-4). You can change the address by pressing [ANALYZER ADDRESS], then using the numeric keypad and the [ENTER] softkey to enter a new value. However, the instructions in the verification procedure assume that the analyzer address is set to 11.



Analyzer's Address

```
Local/HP-IB  35665 Adrs: 11        Plot Adrs: 5
             Prnt Adrs: 1          Disk Adrs: 5
CALC1:ACT ON                              RMT TLK LTN SRQ
MEASUREMENT RUNNING                       OV1 OV2 SRC ARM TRG TRC
A: CH1 Pwr Spec    X: 0        Hz         Y: -53.068 dBVrms
```

Figure 1-4. HP 35665A Screen After Pressing Local/HP-IB

5. Verify that the analyzer is set to the addressable-only mode. The softkey labels that appear when you press the [**Local/HP-IB**] hardkey include [SYSTEM CONTROLLR] and [ADDRESSBL ONLY]. Only one of these two softkeys can be selected at a time, and the one that is selected will have a box around it. Press [ADDRESSBL ONLY] if it is not selected.

Note

In any HP-IB system there can be more than one device with controller capabilities. But at any given time, only one device on the bus can be designated as the system controller.

Quick Verification

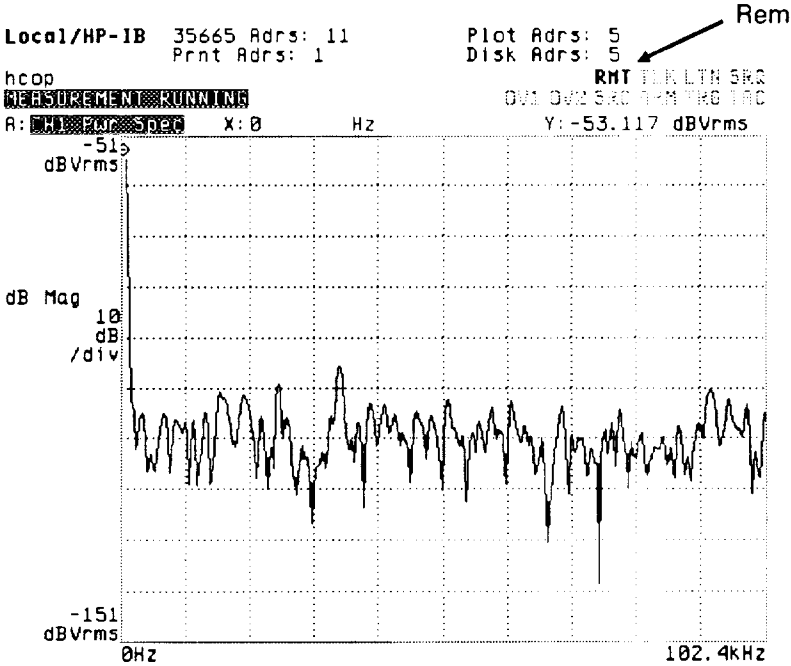
Having just completed all the steps in the preceding section, you are ready to verify that commands can be sent over the HP-IB. In this quick verification, you are going to enter an HP BASIC keyword that should place the HP 35665A under remote control.

Procedure

1. Type the following on the computer:

   ```
   REMOTE 711
   ```

   then press the computer's ENTER key. The RMT indicator should appear highlighted at the top of the HP 35665A's screen (see figure 1-5). This tells you that the analyzer is under remote control of the computer.



Local/HP-IB 35665 Adrs: 11
Prnt Adrs: 1
Plot Adrs: 5
Disk Adrs: 5
hcop
MEASUREMENT RUNNING
A: CH1 Pwr Spec X: 0 Hz Y: -53.117 dBVrms

Remote Indicator

Figure 1-5. RMT (Remote) Indicator

2. Now type the following on the computer:

   ```
   LOCAL 711
   ```

   then press the computer's ENTER key. The RMT indicator should become "ghosted." That is, the word is still readable, but no longer highlighted. This tells you that the analyzer has been returned to front-panel control.

Troubleshooting

If the RMT indicator does not perform as expected, check the following:

- Be sure that your HP-IB cable connections are secure and that the cable is free of defects.
- Verify that the analyzer is in addressable-only mode and that its address is set to 11.
- Be sure you are using the required equipment and software.
- Be sure you have loaded all the required language extensions into the computer. (For a list of loaded extensions, enter the following into the computer: LIST BIN)

If everything seems to be in order, but the RMT indicator still doesn't perform as expected, contact your local HP Sales/Service office.

Verification Program

The quick verification procedure confirmed that the computer could talk to the analyzer. However, you must write a short program to confirm that the analyzer can talk to the computer. If you enter the program correctly, the computer displays the following statement when you run the program:

```
FREQUENCY SPAN IS: 102400 HZ
```

Note The following procedure assumes that you have completed all the steps in "Configuring the HP-IB System" using all the required equipment and software.

Procedure

1. Enter the following program:

```
10 PRINTER IS 1
20  ASSIGN @Hp35665a TO 711
30  ABORT 7
40  CLEAR @Hp35665a
50  OUTPUT @Hp35665a;"*RST"
60  OUTPUT @Hp35665a;"SENS:FREQ:SPAN:FULL"
70  OUTPUT @Hp35665a;"SENS:FREQ:SPAN?"
80  ENTER @Hp35665a;A
90  PRINT "FREQUENCY SPAN IS:";A;"HZ"
100  END
```

2. See your computer and software documentation if you need help entering the program.
3. Press the computer's RUN key. The program tells the analyzer to preset. It then tells the analyzer to select its widest frequency span. Finally, the program asks the analyzer to return the value of the widest span and has the computer display the returned value as follows:

```
FREQUENCY SPAN IS: 102400 HZ
```

Troubleshooting

If the program does not run correctly, be sure you have entered the program exactly as listed. Then go back to "Quick Verification" for additional troubleshooting hints.

Need Assistance?

If you need assistance, contact your nearest Hewlett-Packard Sales and Service Office listed in the HP Catalog, or contact your nearest regional office listed at the back of this guide. If you are contacting Hewlett-Packard about a problem with your HP 35665A Dynamic Signal Analyzer, please provide the following information:

- Model number: HP 35665A
- Serial number and firmware version:
  (To locate the analyzer's serial number and firmware version,
  press [**System Utility**] [S/N VERSION].)
- Options:
- Date the problem was first encountered:
- Circumstances in which the problem was encountered:
- Can you reproduce the problem?
- What effect does this problem have on you?

Behavior in an HP-IB System

Introduction

This chapter explains how the HP 35665A behaves in an HP-IB system.

First it describes how to configure the HP 35665A as the controller of an HP-IB system or as an addressable device on the bus. Next, it explains how the HP 35665A functions in an HP-IB system and how the analyzer communicates with other devices. At the end of the chapter is a listing of the interface capabilities as defined by the IEEE 488.1 standard.

Controller Capabilities

The HP 35665A can be configured as an HP-IB system controller or as an addressable-only HP-IB device. To configure the analyzer, press the [**Local/HP-IB**] key on the front panel. To configure the analyzer as the HP-IB system controller press [SYSTEM CONTROLLR] which appears in the softkey menu. To configure the HP 35665A as an addressable-only device on the bus, press [ADDRESSBL ONLY].

Normally, the HP 35665A is not configured as the system controller unless it is the only controller on the bus. Such a setup would be likely if you only wanted to control printers or plotters with the analyzer. It might also be the case if you were using HP Instrument BASIC to control other test equipment.

When the analyzer is used with another controller on the bus, it is normally configured as an addressable-only HP-IB device. In this configuration, when the analyzer is passed control it can function as the active controller. It can also function as a talker or listener.

Bus Management Commands vs. Device Commands

The HP-IB contains an attention (ATN) line that determines whether the interface is in command mode or data mode. When the interface is in command mode (ATN TRUE), a controller can send bus management commands over the bus.

Bus management commands:

- Specify which devices on the interface can talk (send data) and which can listen (receive data).
- Instruct devices on the bus, either individually or collectively, to perform a particular interface operation.

The analyzer's responses to bus management commands are described in the next section.

When the interface is in data mode, device commands and data can be sent over the bus. Device commands are sent by the controller, but data can be sent either by the controller or a talker. The HP 35665A responds to two different kinds of device commands:

- Common commands which access device functions required by the IEEE 488.2 standard.
- Subsystem commands which access the bulk of the analyzer's functions.

The analyzer's responses to device commands are described in the Command Reference chapters 8 through 29.

Response to Bus Management Commands

This section describes how the HP 35665A responds to the HP-IB bus management commands. The commands themselves are defined by the IEEE 488.1 standard. Refer to the documentation for your controller's language system to determine how to send these commands.

Device Clear (DCL)

When the analyzer receives this command, it:

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

The command does not affect:

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

Group Execute Trigger (GET)

This command triggers the analyzer (causes it to start collecting measurement data) if the following two things are true:

- The trigger source is the HP-IB (TRIG:SOUR BUS).
- The analyzer is ready to trigger. (Bit 5 of the Operational Status condition register is set.)

Interface Clear (IFC)

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

Local Lockout (LLO)

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

Local lockout ensures that the analyzer's [**Local/HP-IB**] hardkey 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, thus enabling all other front-panel keys. However, when the key is disabled, it does not allow the front-panel operator to return the analyzer to local mode.

Parallel Poll

The HP 35665A ignores all of the following parallel poll commands:

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

Remote Enable (REN)

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

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 [**Local/HP-IB**] hardkey. See Local Lockout for more information.

Selected Device Clear (SDC)

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

When the analyzer receives this command, it:

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

The command does not affect:

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

Serial Poll

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

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

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

Take Control Talker (TCT)

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

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

Message Exchange

The analyzer communicates with the controller and other devices on the HP-IB via 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 3.

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. (Query termination is discussed in "Query Response Generation," later in this chapter.)
- Once it receives a terminated query, the analyzer expects to talk before it is told to do something else.

See figure 2-1.

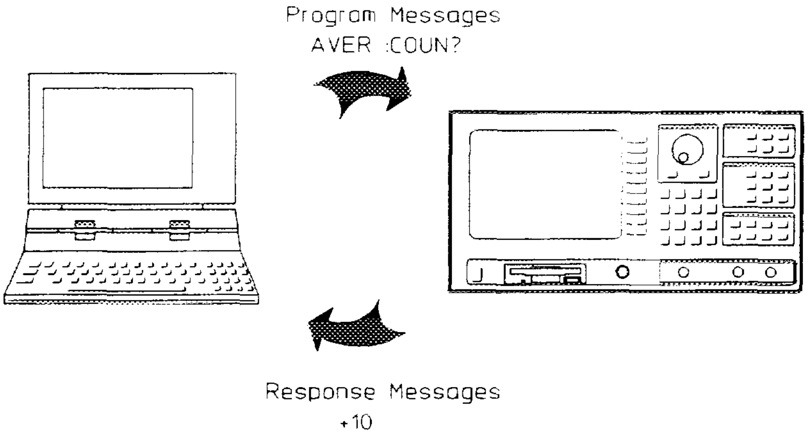


Figure 2-1. HP-IB Message Exchange

HP-IB Queues

Queues enhance the exchange of messages between the HP 35665A and other devices on the bus. The analyzer contains:

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

Input Queue

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

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

The input queue 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:

- You turn on the analyzer.
- You send the Device Clear (DCL) or the Selected Device Clear (SDC) command.

Error Queue

The error queue temporarily stores up to 5 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:

- You turn on the analyzer.
- You send the *CLS command.

Output Queue

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

- You turn on the analyzer.
- You send the Device Clear (DCL) or the Selected Device Clear (SDC) command.

Command Parser

The command parser reads program messages from the input queue in the order they were received from the bus. It analyzes the syntactic elements of 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. (For more information on the command tree, see chapter 3.) When the command parser is reset, the next syntactic element it receives is expected to arise from the base of the analyzer's command tree.

The parser is reset when:

- You turn on the analyzer.
- You send the Device Clear (DCL) or the Selected Device Clear (SDC) command.
- Follow a semicolon with a colon in a program message. (For more information, see "Sending Multiple Commands" in chapter 3.)
- A program message terminator is received.

Query Response Generation

When the HP 35665A parses a query, the response to that query is placed in the analyzer's output queue. You should read a query response immediately after sending the query. This ensures that the response is not cleared before it is read. The response is cleared if either of the following message exchange conditions occur:

- Unterminated condition — This results when you neglect to properly terminate the query with an ASCII line feed character or the HP-IB END message (EOI set true) before you read the response.

- Interrupted condition — This results when you send a second program message before reading the response to the first.

- Buffer deadlock — This results when you send a program message that exceeds the length of the input queue or generates more response data than fits in the output queue.

Synchronization

This section describes tools you can use 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 device commands that you send to the analyzer 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

The HP 35665A has the following overlapped commands:

- CALCulate[1|2]:CFIT[:IMMediate]
- CALCulate[1|2]:FEED
- CALCulate[1|2]:FORMat
- CALCulate[1|2]:SYNThesis[:IMMediate]
- CALCulate[1|2]:UNIT:AMPLitude
- CALCulate[1|2]:UNIT:ANGLe
- CALCulate[1|2]:UNIT:VOLTage
- HCOPy[:IMMediate]
- INITiate[:IMMediate]
- MMEMory:COPY
- MMEMory:DELete
- MMEMory:INITialize
- MMEMory:REName
- [SENSe:]TCAPture[:IMMediate]

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 at narrow spans or when averaging is enabled.

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

- *WAI — Holds off the processing of subsequent commands until the NPO flag is set to 1. Use this command to ensure that commands in the analyzer's input queue are processed in the order received.
- *OPC? — Places a 1 in the analyzer's output queue when the NPO flag is set to 1. Use this query to synchronize your controller to the completion of an overlapped command.
- *OPC — Sets bit 0 of the Standard Event event register to 1 when the NPO flag is set to 1. Use this command when you need to synchronize your controller to the completion of an overlapped command, but also want to leave the controller free to perform other tasks while the command is executing.

Each command requires a different amount of overhead in your program. *WAI requires the least overhead, *OPC requires the most.

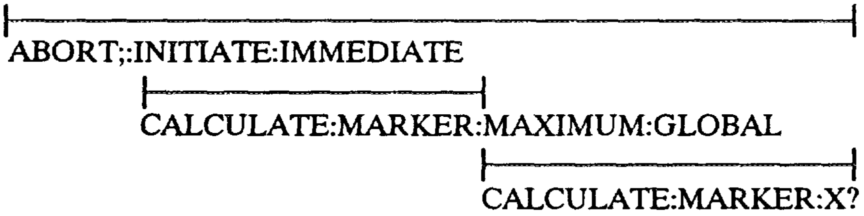
*WAI

This command holds off the processing of subsequent device commands until all overlapped commands are completed (the NPO flag is set to 1). The following example demonstrates the effect of the *WAI command.

To determine which frequency component of a signal contains the greatest amount of energy, you would send the following series of commands:

```
OUTPUT 711;"ABORT;:INITIATE:IMMEDIATE"                 !Restart the measurement.
OUTPUT 711;"CALCULATE:MARKER:MAXIMUM:GLOBAL"           !Search for max energy.
OUTPUT 711;"CALCULATE:MARKER:X?"                       !Which frequency?
ENTER 711; X
Print "MARKER at ';X;" Hz"
```

The following timeline shows how the processing times of the three commands relate to each other.

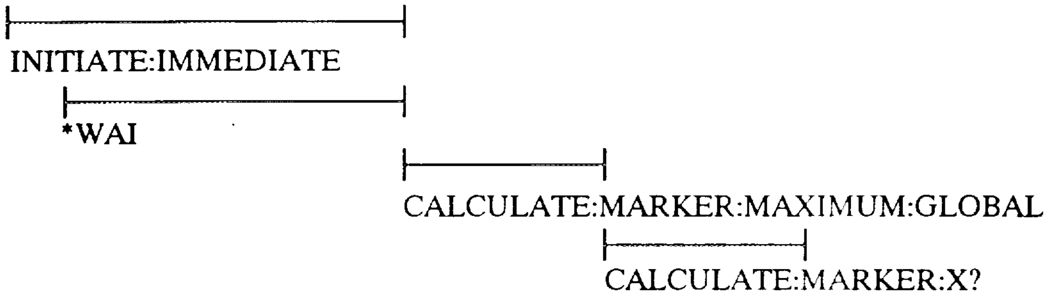


INITIATE:IMMEDIATE is an overlapped command because it does not hold off the processing of CALCULATE:MARKER:MAXIMUM:. Remember, INITIATE:IMMEDIATE is not considered complete until the measurement is complete. In this example, the marker searches for maximum energy before the measurement completes. The CALCULATE:MARKER:X? query could return an incorrect value.

To solve the problem, insert a *WAI command.

```
OUTPUT 711;"ABORT;:INITIATE:IMMEDIATE"                 !Restart the measurement.
OUTPUT 711;"*WAI"                                      !Wait until complete.
OUTPUT 711;"CALCULATE:MARKER:MAXIMUM:GLOBAL"           !Search for max energy.
OUTPUT 711;"CALCULATE:MARKER:X?"                       !Which frequency?
```

The timeline now looks like this:



The *WAI command keeps the search from taking place until the measurement is completed. The CALCULATE:MARKER:X? query returns the correct value.

***OPC? and *OPC**

If you send *OPC?, a 1 is placed in the analyzer's output queue when the NPO flag is set to 1. This allows you to effectively pause the controller until all pending overlapped commands are completed. Design your program so that it must read the queue before it continues.

If you send *OPC, bit 0 of the Standard Event register is set to 1 when the NPO flag is set to 1. This allows you to use the analyzer's register structure to generate a service request when all pending overlapped commands are completed. However, your program must also have enabled bit 0 of the Standard Event register and bit 5 of the Status Byte register. When you synchronize the analyzer and controller in this manner, the controller is free to perform some other task until the service request is generated.

*OPC only informs you when the NPO flag is set to 1. It does not hold off the processing of subsequent commands. As a result, you should not send any commands to the analyzer between the time you send *OPC and the time you receive a service request. Any command you send to the instrument is executed. However, it may affect how the instrument responds to the previously sent *OPC. If you send commands after an *OPC command, be especially careful which commands are sent!

Passing Control

The HP 35665A requires temporary control of the HP-IB to complete some commands. (If a command requires control of the bus, a special note appears in the command's reference description.) After sending such a command, the active controller must pass control to the analyzer. When the analyzer completes the command, it automatically passes control of the bus back to the controller.

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

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

Here is a procedure for passing control:

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

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

Chapter 6, "Example Programs," contains an example program that passes control to the analyzer. In the example program, "DTXFRB", control is passed so the analyzer can run an HP Instrument BASIC program which then acts as a device on the bus.

HP-IB Interface Capabilities

The HP 35665A has the following interface capabilities, as defined by the IEEE 488.1 standard:

SH1	full Source handshake capability
AH1	full Acceptor handshake capability
T6	basic Talker, Serial Poll, no Talk Only, unaddress if MLA
TE0	no Extended Talker capability
L4	basic Listener, no Listen Only, unaddress if MTA
LE0	no Extended Listener capability
SR1	full Service Request capability
RL1	full Remote/Local capability
PP0*	Parallel Poll capability
DC1	full Device Clear capability
DT1	full Device Trigger capability
C1	System Controller capability
C2	send IFC and take charge Controller capability
C3	send REN Controller capability
C4*	respond to SRQ
C6*	send IFC, receive control, pass control, parallel poll, pass control to self
C10*	send IFC, receive control, pass control, parallel poll
C12**	send IF messages, receive control, pass control
E2	tri-state drivers

* only when an HP Instrument BASIC program is running

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

3

Programming with HP-IB Commands

Introduction

This chapter will shows you how to create more efficient programs with HP-IB commands. It describes the general structure of the HP-IB command tree and the syntax rules for HP-IB program and response messages. It also explains how to:

- Send multiple commands.
- Shorten commands by abbreviating mnemonics.
- Shorten commands by omitting implied mnemonics.

The Command Tree

The HP-IB commands for the HP 35665A are based on the Standard Commands for Programming Instruments, known as SCPI. The SCPI standard organizes related instrument functions by grouping them together on a common branch of a command tree. Each branch is assigned a mnemonic to indicate the nature of the related functions. For example, the HP 35665A's trigger functions are grouped under the TRIGGER branch of the command tree. The TRIGGER branch is only one of 21 major SCPI branches—called subsystems—used by the HP 35665A. See figure 3-1 for a model of how the SCPI subsystems are organized to manage the measurement and data flow for the HP 35665A.

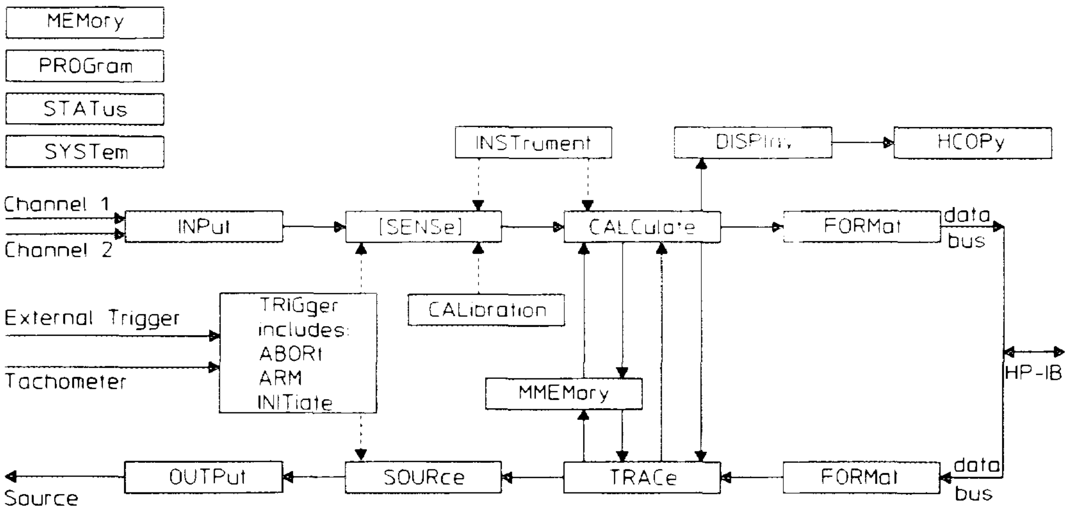


Figure 3-1. Measurement and Data Flow of the HP 35665A

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 CALCULATE branch serves as a good example.

The branching process continues until each analyzer function is assigned to its own branch. For example, the function that turns the analyzer's peak track marker on and off is assigned to the TRACK branch of the GLOBAL branch of the MAXIMUM branch of the MARKER branch. The command looks like this:

```
CALCULATE:MARKER:MAXIMUM:GLOBAL:TRACK ON
```

Note

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

Sending Multiple Commands

You can send multiple commands 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 711;"CALCULATE:MARKER:MODE RELATIVE;:CALCULATE:MARKER:MAXIMUM:GLOBAL"
```

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 you to simplify the previous program message. If you take advantage of this parser function, you create the equivalent, but simpler, program message:

```
OUTPUT 711;"CALCULATE:MARKER:MODE RELATIVE;MAXIMUM:GLOBAL"
```

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 preceding 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 711;"CALCULATE:MARKER:STATE ON"
    OUTPUT 711;"CALCULATE:MARKER:MODE RELATIVE"
    OUTPUT 711;"CALCULATE:MARKER:MAXIMUM:GLOBAL"
    OUTPUT 711;"CALCULATE:MARKER:MAXIMUM:RIGHT"
```

The single program message is:

```
OUTPUT 711;"CALCULATE:MARKER:STATE ON;MODE RELATIVE;MAXIMUM:GLOBAL;RIGHT"
```

Command Abbreviation

Each command mnemonic has a long form and a short form. Only the exact short form or the exact long form are accepted.

The short forms of the mnemonics allow you to send abbreviated commands. The mnemonics' short forms are created according to the following rules:

- If the long form of the mnemonic has less than four characters, the short form is the same as the long form. For example, ARM remains ARM.
- If the long form of the mnemonic has exactly four characters, the short form is the same as the long form. For example, MODE remains MODE.
- If the long form of 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 of 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, INPUT becomes INP.

Note

The syntax descriptions in the Command Reference chapters use upper-case characters to identify the short form of a particular mnemonic.

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

```
OUTPUT 711;"CALCULATE:MARKER:MODE RELATIVE;OFFSET ON;MAXIMUM:GLOBAL;RIGHT"
```

becomes:

```
    OUTPUT 711;"CALC:MARK:MODE REL;MAX:GLOB;RIGH"
```

Implied Mnemonics

You can omit some mnemonics from HP-IB commands without changing the effect of the command. These special mnemonics are called implied mnemonics, and they are used in many subsystems.

The OUTPUT subsystem contains the implied mnemonic STATE at its first branching point. As a result, you can send either of the following commands to the analyzer (using HP BASIC) to turn on the source:

```
OUTPUT 711;"OUTPUT:STATE ON"

OUTPUT 711;"OUTPUT ON"
```

The first mnemonic in the SENSE subsystem is also an implied mnemonic, so you can omit it from any SENSE command. These two commands are equivalent:

```
OUTPUT 711;"SENSE:FREQUENCY:SPAN:FULL"
OUTPUT 711;"FREQUENCY:SPAN:FULL"
```

and so are these:

```
OUTPUT 711;"SENSE:SWEEP:MODE AUTO"

OUTPUT 711;"SWEEP:MODE AUTO"
```

Implied mnemonics are identified by brackets [ ].

Message Syntax

As mentioned in chapter 2, the analyzer uses program messages and response messages to communicate with other devices on the HP-IB. This section uses syntax diagrams to describe the general syntax rules for both kinds of messages.

Conventions

The flow of syntax diagrams is generally from left to right. However, elements that repeat require a return path that goes from right to left. Any message that can be generated by following a diagram from its entry point to its exit point, in the direction indicated by the arrows, is valid.

Angle brackets < > enclose the names of syntactic items that need further definition. The definition is included either in the text accompanying the diagram, in a subsequent diagram, or in the next section, "Common Definitions."

The symbol ::= means "is defined as." When two items are separated by this symbol, the second item can replace the first in any statement that contains the first item.

Common Definitions

The syntax diagrams have the following definitions in common:

- <LF> is the line feed character (ASCII decimal 10).
- < ^ END> is assertion of the HP-IB END message (EOI set true) while the last byte of data is on the bus.
- <SP> is the space character (ASCII decimal 32).
- <WSP> is one or more white space characters (ASCII decimal 0-9 and 11-32).
- <digit> is one character in the range 0-9 (ASCII decimal 48-57).
- <alpha> is one character of the alphabet. The character can be either upper-case (ASCII decimal 65-90) or lower-case (ASCII decimal 97-122) unless otherwise noted.

Special Syntactic Elements

Several syntactic elements have special meanings:

- colon (:) — When a command or query contains a series of mnemonics, the mnemonics 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. For more information, see "The Command Tree" and "Sending Multiple Commands" at the beginning of this chapter.

- semicolon (;) — When a program message contains more than one command or query, a semicolon is used to separate them from each other. For example, if you want to autorange the analyzer's inputs and then start a measurement using one program message, the message would be:

  ```
  SENSE:VOLT:RANGE:AUTO ONCE;:ABORT;:INITIATE:IMMEDIATE
  ```

- comma (,) — A comma separates the data sent with a command or returned with a response. For example, the SYSTEM:TIME command requires three values to set the analyzer's clock: one for hours, one for minutes, and one for seconds. A message to set the clock to 8:45 AM would be:

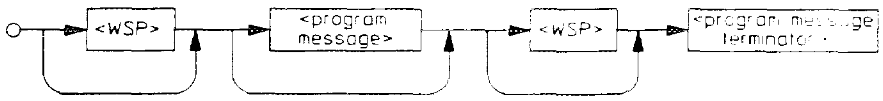
  ```
  SYSTEM:TIME 8,45,0
  ```

- <WSP> — One white space is required to separate a program header (the command or query) from its program data (the parameters). For example, the command "SYSTEM:TIME 8,45,0" contains a space between the program header (SYSTEM:TIME) and its program data (8,45,0). White space characters are not allowed within a program header.

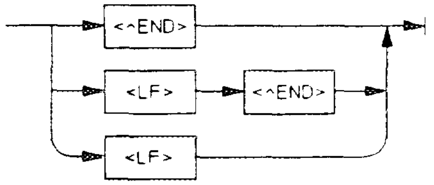
- <program message terminator> — A message terminator is required at the end of a program message or a response message. Program message terminators are described in "Program Message Syntax." Response terminators are described in "Response Message Syntax."

Program Message Syntax

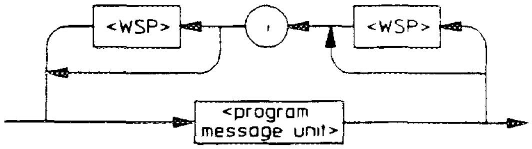
The syntax for a terminated program message is:



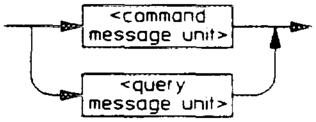
<program message terminator>::=



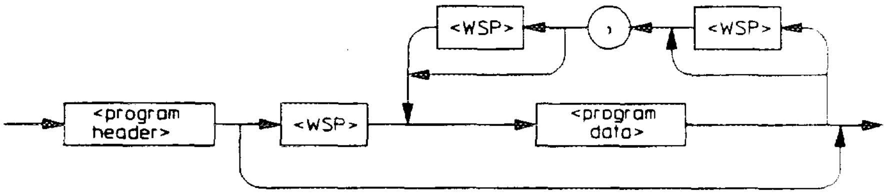
<program message>::=



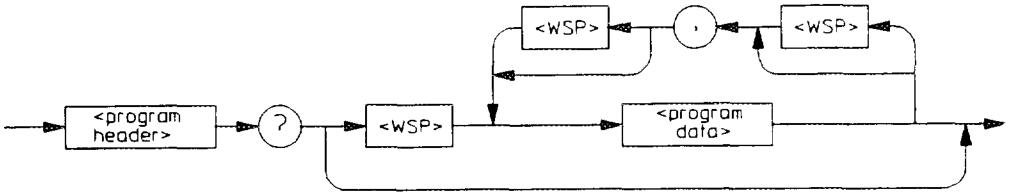
<program message unit>::=



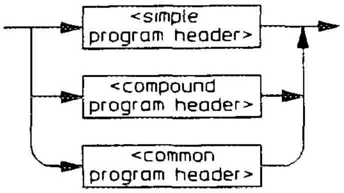
<command message unit>::=



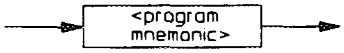
<query message unit>::=



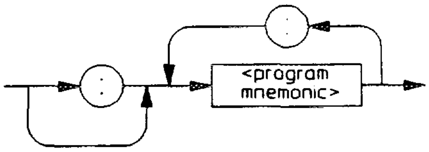
<program header>::=



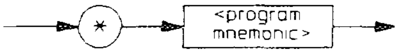
<simple program header>::=



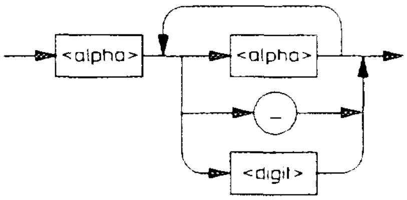
<compound program header>::=



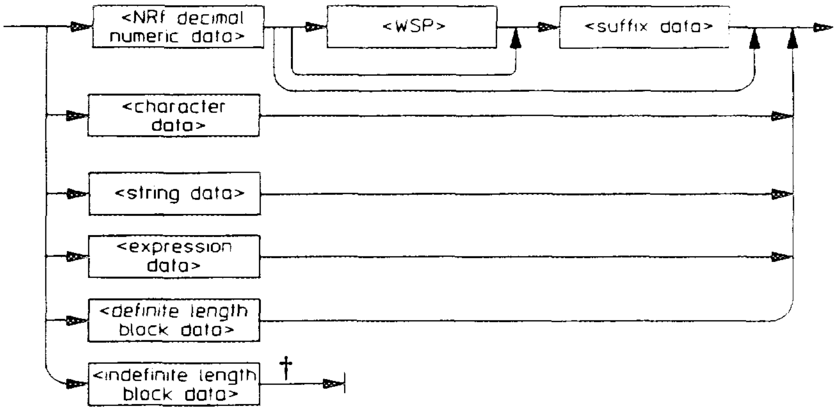
<common program header>::=



<program mnemonic>::=



<program data>::=

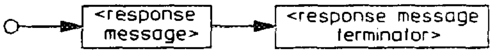


† The definition of indefinite length block data includes termination with <LF>< ^ END>. This serves the dual function of terminating the data and terminating the program message.

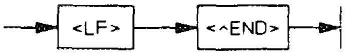
Program data and response data are described in chapter 4, "Transferring Data." <suffix data> is dependent on the command sent.

Response Message Syntax

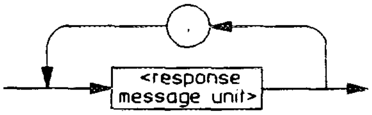
The syntax for a terminated response message is:



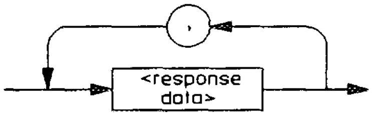
<response message terminator>::=



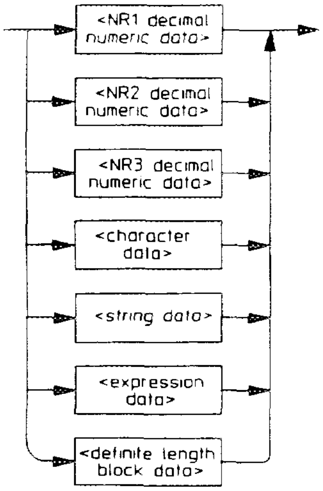
<response message>::=



<response message unit>::=



<response data>::=



Response data and program data are described in chapter4, "Transferring Data."

4

Transferring Data

Introduction

Data is transferred between the HP 35665A and a controller via the HP-IB data lines, DIO1 through DIO8. Such transfers occur in a byte-serial (one byte at a time), bit-parallel (8 bits at a time) fashion. This chapter discusses the following aspects of data transfer:

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

Data Types

The HP 35665A uses a number of different data types during data transfers. They are described in this section using syntax diagrams.

Conventions

The flow of syntax diagrams is generally from left to right. However, elements that repeat require a return path that goes from right to left. Any data you can generate by following a diagram from its entry point to its exit point, in the direction indicated by the arrows, is valid.

Angle brackets < > enclose the names of syntactic items that need further definition. The definition is included either in the text accompanying the diagram, or in the next section, "Common Definitions."

Common Definitions

The syntax diagrams have the following definitions in common:

- <LF> is the line feed character (ASCII decimal 10).
- <^END> is assertion of the HP-IB END message (EOI set true) while the last byte of data is on the bus.
- <SP> is the space character (ASCII decimal 32).
- <WSP> is one or more white space characters (ASCII decimal 0-9 and 11-32).
- <digit> is one character in the range 0-9 (ASCII decimal 48-57).
- <non-zero digit> is one character in the range 1-9 (ASCII decimal 49-57).
- <alpha> is one character of the alphabet. The character can be either upper-case (ASCII decimal 65-90) or lower-case (ASCII decimal 97-122) unless otherwise noted.

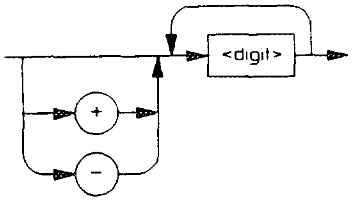
Decimal Numeric Data

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

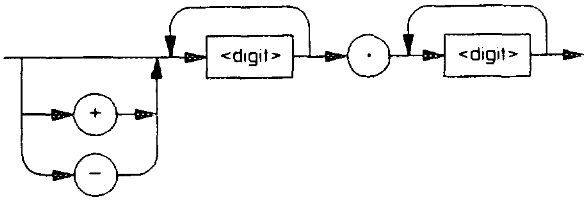
- Integers—returned as NR1 data.
- Fixed-point numbers—returned as NR2 data.
- Floating-point numbers—returned as NR3 data.

You can use the more flexible syntax of NRf data when sending any of the three decimal numeric data types to the analyzer. The NRx data syntax is described in the following four syntax diagrams.

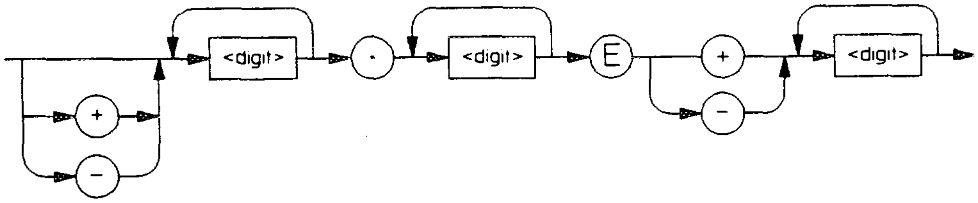
NR1 data:



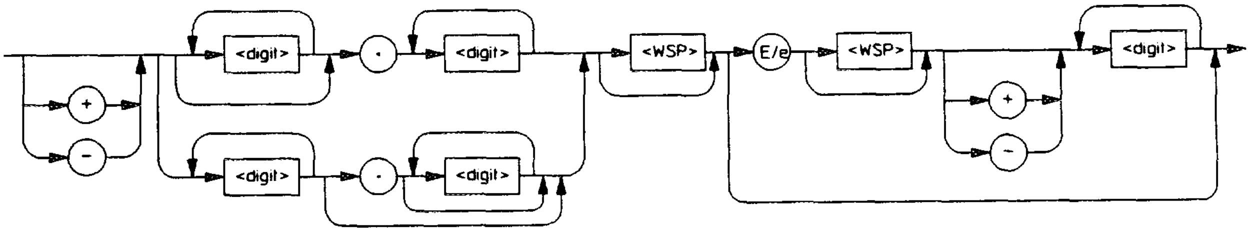
NR2 data:



NR3 data:

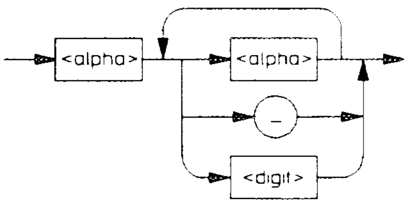


NRf data:



Character Data

To send character data use the following format:

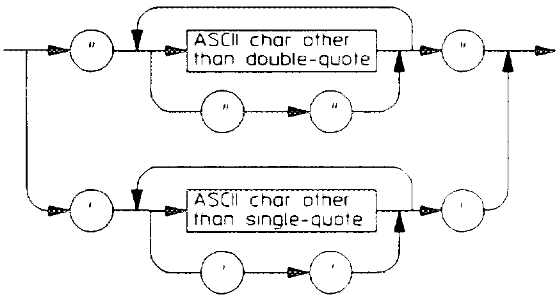


The "_" in the circle is the underscore character (ASCII decimal 95).

The analyzer returns character data in the format it uses to send character data, with one exception—the analyzer never returns lower-case alpha characters.

String Data

The format you use to send string data is:

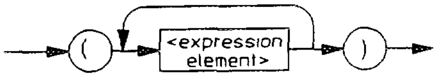

"
ASCII char other than double-quote
"
"
"
'
ASCII char other than single-quote
'
'
'

You must use two double-quote characters ("""") to represent one ("") in a string that is delimited by double-quote characters. You must use two single-quote characters ('') to represent one (') in a string that is delimited by single-quote characters.

The analyzer returns string data in the same format it uses to send string data, with one exception: the analyzer never returns string data using the single-quote path.

Expression Data

To send expression data use the following format:



Parameter values can be set with expression data. Commands that use expression data such as CALC:MATH:EXPR have syntax descriptions in the "Command Reference" chapters which list acceptable expression elements.

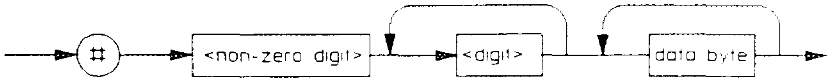
The analyzer returns expression data surrounded by double-quotes: "<expression data>".

Block Data

The analyzer returns definite length block data in response to queries. When you send block data to the analyzer, you can send it either as definite length or indefinite length block data.

Definite Length Block Data

To send definite length block data use the following format:



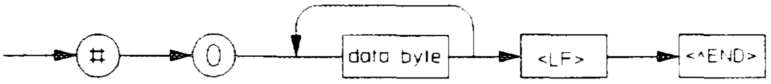
The elements #, <non-zero digit>, and <digit> make up a header for the block data. <non-zero digit> indicates how many times <digit> is repeated. The <digits> are interpreted as a single decimal number, which indicates how many bytes of data follow in the block.

Block Header				Block Data			
byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	...	byte 19
#	2	1	5	<data_byte_1>	<data_byte_2>	...	<data_byte_15>

In the example above, the element <non-zero digit> is 2. This indicates that the following two bytes are taken together as a single decimal number. In the example, the number is 15. The following 15 bytes are the 5th through 19th bytes of the data transfer. However, they are the 1st through 15th bytes of the data block.

Indefinite Length Block Data

To send indefinite length block data use the following format:



The first two bytes of the data transfer, # and 0, are the header for the block data. The data itself does not begin until the third byte of the data transfer.

Data Encoding for Block Data

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 an external controller. There are two specifiers:

- REAL specifies the block data type. You can use either the definite or indefinite length syntax. The block is transferred as a series of binary-encoded floating-point numbers.

- ASCii specifies the decimal numeric data type; either the NR1, NR2, NR3, or NRf syntax. The block is transferred as a series of ASCII-encoded NRx numbers separated by commas.

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

- CALCulate:MATH:DATA
- PROGram:EXPLicit:DEFine
- PROGRAM[:SELected]:DEFine
- SYSTem:SET

ASCII Encoding

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

When you use ASCII encoding for block data, you can specify the number of significant digits to be returned for each number in the block. For example, if you send the command FORMat:DATA ASCii,7 all numbers return as NR3 data with 7 significant digits.

Binary Encoding

When you use binary encoding for block data, all numbers in the block are transferred as 32-bit or 64-bit binary floating-point numbers. The binary floating-point formats are defined in the IEEE 754-1985 standard. Send FORMat:DATA REAL,32 to select the 32-bit format. Send FORMat:DATA REAL,64 to select the 64-bit format.

Many languages that run on controllers use the binary floating-point formats. Both of the formats have three fields in common. However, the length of the fields are different for each. The fields and their bit lengths appear in table 4-1.

Table 4-1. Fields in Binary Floating-point Numbers

Field	Width of Field	
	32-bit format	64-bit format
sign (s)	1 bit	1 bit
exponent (e)	8	11
fraction (f)	23	52

When the 32-bit format is used, the decimal value of the exponent field ranges from −126 to +127, with a bias of +127. When the 64-bit format is used, the decimal value of the exponent field ranges from − 1022 to +1023, with a bias of +1023.

You can use the following formulas to determine the value (x) of a 32-bit binary floating-point number. (s, e, and f must be converted from binary to decimal before using the formulas.)

If $e = 255$ and $f = 0$	then x is not a number
If $e = 255$ and $f = 0$	then $x = -1^{s}(\infty)$
If $0 < e < 255$	then $x = -1^{s}(2^{e-127})(1 + f)$
If $e = 0$ and $f = 0$	then $x = -1^{s}(2^{e-126})(0 + f)$
If $e = 0$ and $f = 0$	then $x = -1^{s}(0)$

32-bit binary floating-point numbers are sent over the bus as follows:

DIO	8	7	6	5	4	3	2	1
byte 1	s	e	e	e	e	e	e	e
byte 2	e	f	f	f	f	f	f	f
bytes 3 and 4	f	f	f	f	f	f	f	f

You can use the following formulas to determine the value (x) of a 64-bit binary floating-point number. (Again, s, e, and f must be converted from binary to decimal before using the formulas.)

If $e = 2047$ and $f = 0$	then x is not a number
If $e = 2047$ and $f = 0$	then $x = -1^{s}(\infty)$
If $0 < e < 2047$	then $x = -1^{s}(2^{e-1023})(1 + f)$
If $e = 0$ and $f = 0$	then $x = -1^{s}(2^{e-1022})(0 + f)$
If $e = 0$ and $f = 0$	then $x = -1^{s}(0)$

64-bit binary floating-point numbers are sent over the bus as follows:

DIO	8	7	6	5	4	3	2	1
byte 1	s	e	e	e	e	e	e	e
byte 2	e	e	e	e	f	f	f	f
bytes 3 through 8	f	f	f	f	f	f	f	f

Here is an example of a number encoded in the 32-bit binary floating-point format:

byte 1	byte 2	byte 3	byte 4
01000001	10010000	00000000	00000000
seeeeeee	efffffff	ffffffff	ffffffff

Where:

	binary	decimal
s =	0	= 0
e =	10000011	= 131
f =	.001	= .125

Therefore:

$$x = (-1)^{0}(2^{(131-127)})(1.125) = (2^{4})(1.125) = 18$$

5

Using Status Registers

Introduction

The HP 35665A's status registers contain information about various analyzer conditions. This chapter describes the registers and tells you how to use them in your HP-IB programs. The registers are explained in the following sections:

- General Status Register Model
- The Service Request Process
- The HP 35665A's Register Sets

General Status Register Model

Overview

The general status register model, shown in figure 5-1, is the building block of the HP 35665A's status system. Most register sets in the analyzer include all of the registers shown in the general 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). You control the flow by altering bits in the transition and enable registers.

Two register sets—Status Byte and Standard Event—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.

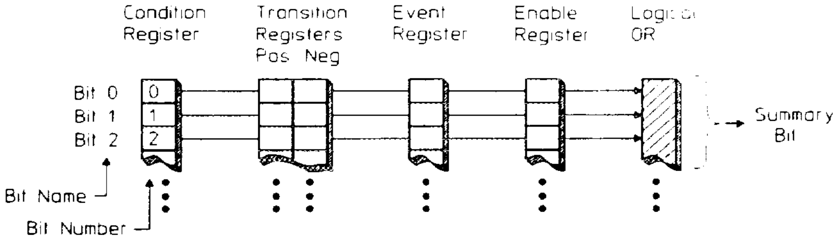


Figure 5-1. General Status Register Model

Condition Register

Condition registers continuously monitor hardware and firmware status. They represent the current state of the instrument. Bits in a condition register are not latched or buffered. They are updated in real time. When the condition monitored by a particular 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 the reporting of condition changes to the event registers. Positive changes in the state of a condition bit (0 to 1) are only reported to the event register if the corresponding positive transition bit is set to 1. Negative changes in the state of a condition bit (1 to 0) are only reported to the event register if the corresponding negative transition bit is set to 1. If you set both transition bits to 1, positive and negative changes are reported to the corresponding event bit. You can read and write most transition registers.

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— either when you read the register or when you send the *CLS (clear status) command. Event registers are read-only.

An event register is cleared when you read it. All event registers are cleared when you send the *CLS command.

Enable Register

Enable registers control the reporting of events (latched conditions) to the register summary bit. If an enable bit is set to one, the corresponding event bit 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.) You can read and write all enable registers.

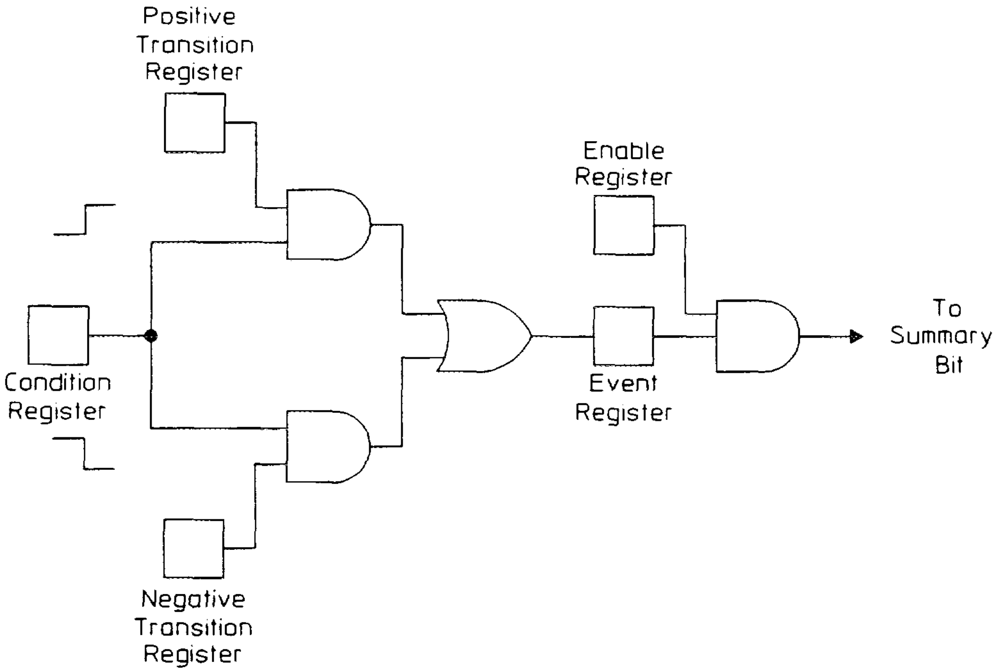


Figure 5-2. Flow of Information Within a Register Set

Note Reading the Event Register, clears the Event Register.

How to Use Registers

There are two methods you can use to access the information in status registers:

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

In the direct-read method, the analyzer has a passive role. It only tells the controller that conditions have changed when the controller asks the right question. In the SRQ method, the analyzer takes a more active role. 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.

When you monitor a condition with the direct-read method, you must:

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

The direct-read method works well if you do not need to know about changes the moment they occur. It does not work well if you must know when a condition changes immediately. To detect a change in a condition your program would need to continuously read the registers at very short intervals. This makes the program relatively inefficient. It is better to use the SRQ method.

The Service Request Process

When you monitor a condition with the SRQ method, you must:

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

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

Generating a Service Request

To use the SRQ method, you must understand how service requests are generated. As shown in figure 5-3, other register sets in the HP 35665A report to the Status Byte. Most of them report directly, but two report indirectly—via the Questionable Status register set.

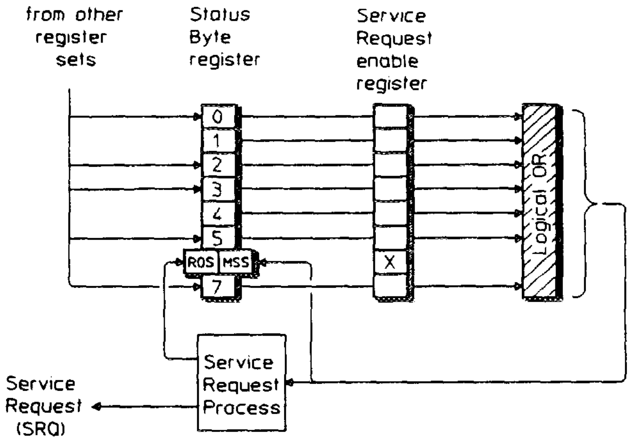


Figure 5-3. Generating a Service Request

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. However, the process is only initiated if both of the following conditions are true:

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

The SRQ process sets the HP-IB's SRQ line true. It also sets the Status Byte's request service (RQS) bit to 1. Both actions are necessary to inform the controller the HP 35665A requires service. Setting the SRQ line only informs the controller that some device on the bus requires service. Setting the RQS bit allows the controller to determine that the HP 35665A, in particular, requires service.

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

Note

When you read the analyzer's Status Byte 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.

Note

To avoid generating a service request when you send the INITiate command; pause the measurement with INIT:CONT OFF *before* setting or enabling the status registers.

The HP 35665A's Register Sets

Register Summary

The HP 35665A uses eight register sets to keep track of instrument status:

- Status Byte.
- Device State.
- Limit Fail.
- Questionable Status.
- Questionable Voltage.
- Standard Event.
- Operational Status.
- User Status.

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

Note

Register bits not explicitly presented in the following sections are not used by the HP 35665A. A query to one of these bits returns a value of 0.

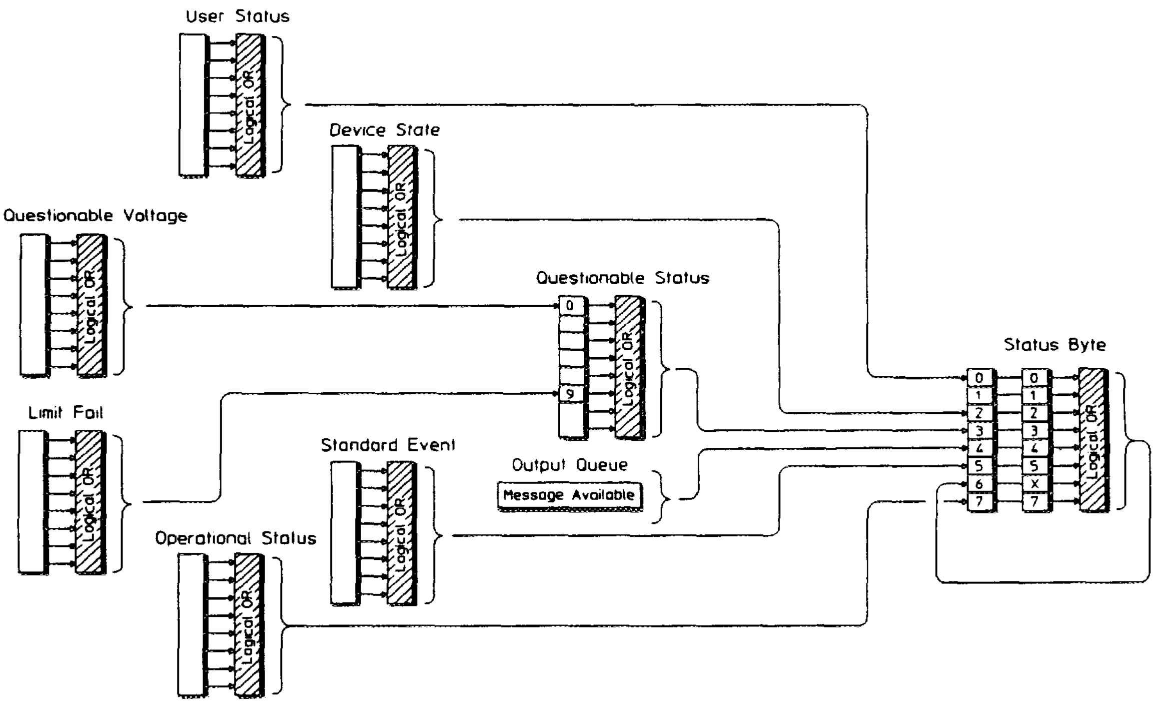


Figure 5-4. HP 35665A Register Sets

Status Byte Register Set

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

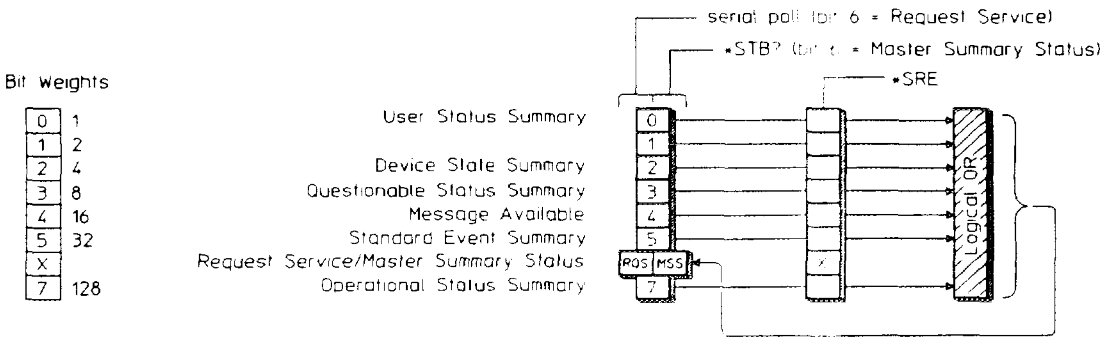

serial poll (bit 6 = Request Service)
*STB? (bit 6 = Master Summary Status)
*SRE
Bit Weights
0 1
1 2
2 4
3 8
4 16
5 32
X
7 128
User Status Summary
Device State Summary
Questionable Status Summary
Message Available
Standard Event Summary
Request Service/Master Summary Status
Operational Status Summary
RQS
MSS
Logical OR

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:

- User Status Summary (bit 0) is set to 1 when one or more enabled bits in the User Status event register are set to 1.
- Device State Summary (bit 2) is set to 1 when one or more enabled bits in the Device State 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 Summary (bit 5) is set to 1 when one or more enabled bits in the Standard Event event register are set to 1.
- Master Summary Status (bit 6, when read by *STB) is set to 1 when one or more enabled bits in the Status Byte register are set to 1.
- Request Service (bit 6, when read by serial poll) is set to 1 by the service request process (see "Generating a Service Request" earlier in this chapter).
- Operational Status Summary (bit 7) is set to 1 when one or more enabled bits in the Operational Status event register are set to 1.

Figure 5-5 also shows the commands you use to read and write the Status Byte registers. See chapter 25 for more information about these commands.

Device State Register Set

The Device State register set monitors the states of three device specific parameters. See figure 5-6

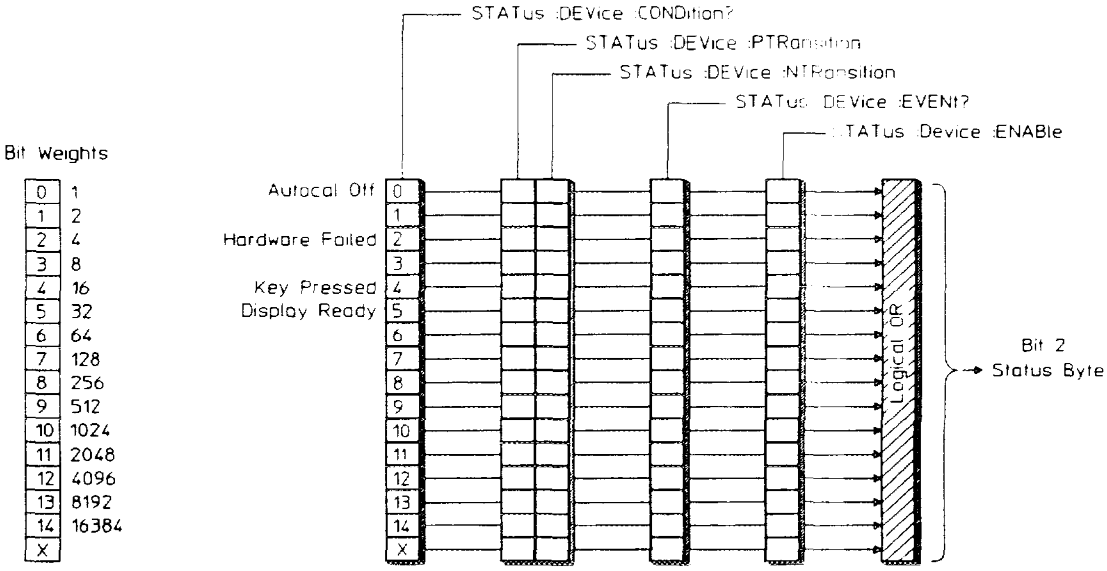


Figure 5-6. The Device State Register Set

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

- Autocal Off (bit 0) is set to 1 when the analyzer's autocalibration function is disabled (CAL:AUTO OFF).
- Hardware Failed (bit 2) is set to 1 when the analyzer detects a failure in its own hardware.
- Key Pressed (bit 4) is set to 1 when one of the front panel keys is pressed. This is an event. The condition register will always return 0 for this bit.
- Display Ready (bit 5) is set to 1 when measurement results are available. This is an event. The condition register will always return 0 for this bit.

Figure 5-6 also shows the commands you use to read and write the Device State registers. See chapter 25 for more information about these commands.

Limit Fail Register Set

The Limit Fail register set monitors limit test results for both traces. See figure 5-7.

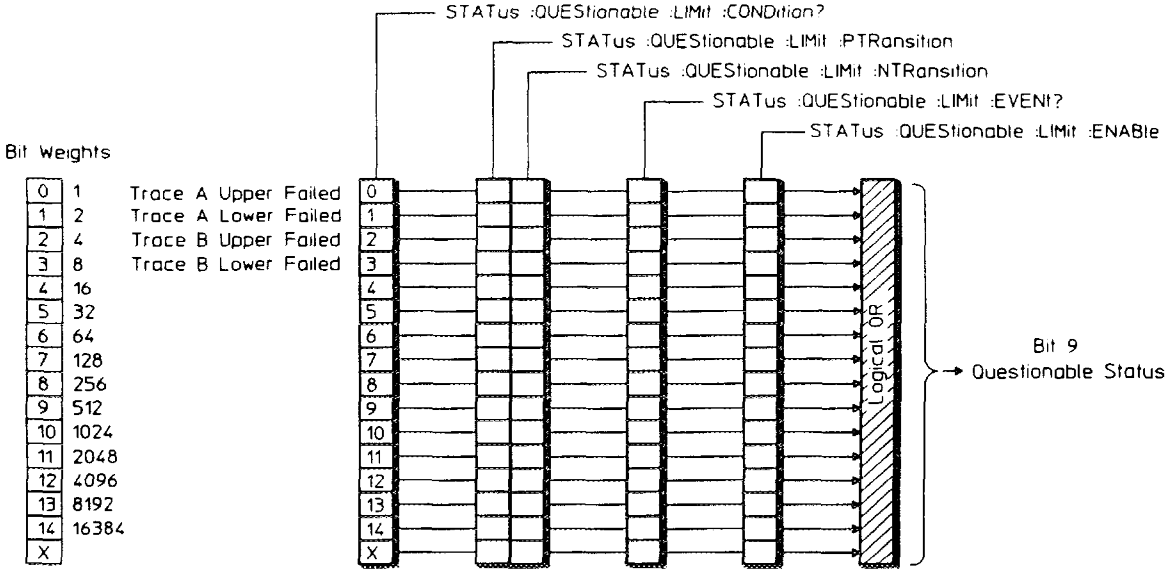


Figure 5-7. The Limit Fail Register Set

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

- Trace A Upper Failed (bit 0) is set to 1 when limit testing is enabled and any point on trace A exceeds its upper limit.
- Trace A Lower Failed (bit 1) is set to 1 when limit testing is enabled and any point on trace A falls below its lower limit.
- Trace B Upper Failed (bit 2) is set to 1 when limit testing is enabled and any point on trace B exceeds its upper limit.
- Trace B Lower Failed (bit 3) is set to 1 when limit testing is enabled and any point on trace B falls below its lower limit.

Figure 5-7 also shows the commands you use to read and write the Limit Fail registers. See chapter 25 for more information about these commands.

Questionable Status Register Set

The Questionable Status register set monitors conditions that affect the quality of measurement data. See figure 5-8.

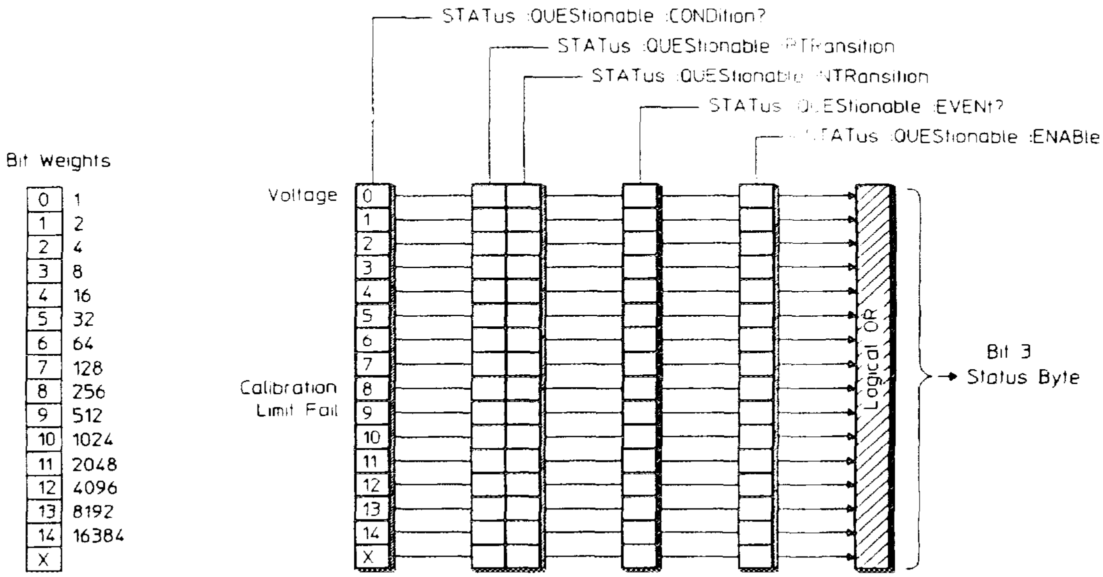

STATus :QUEStionable :CONDition?
STATus :QUEStionable :PTRansition
STATus :QUEStionable :NTRansition
STATus :QUEStionable :EVENt?
STATus :QUEStionable :ENABle
Bit Weights
0 1
1 2
2 4
3 8
4 16
5 32
6 64
7 128
8 256
9 512
10 1024
11 2048
12 4096
13 8192
14 16384
X
Voltage
Calibration
Limit Fail
Logical OR
Bit 3
Status Byte

Figure 5-8. The Questionable Status Register Set

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

- Voltage (bit 0) is set to 1 when one or more enabled bits in the Questionable Voltage event register are set to 1.
- Calibration (bit 8) is set to 1 when the last self-calibration attempted by the analyzer failed.
- Limit Fail (bit 9) is set to 1 when one or more enabled bits in the Limit Fail event register are set to 1.

Figure 5-8 also shows the commands you use to read and write the Questionable Status registers. See chapter 25 for more information about these commands.

Questionable Voltage Register Set

The Questionable Voltage register set monitors conditions that affect the amplitude accuracy of measurement data. See figure 5-9.

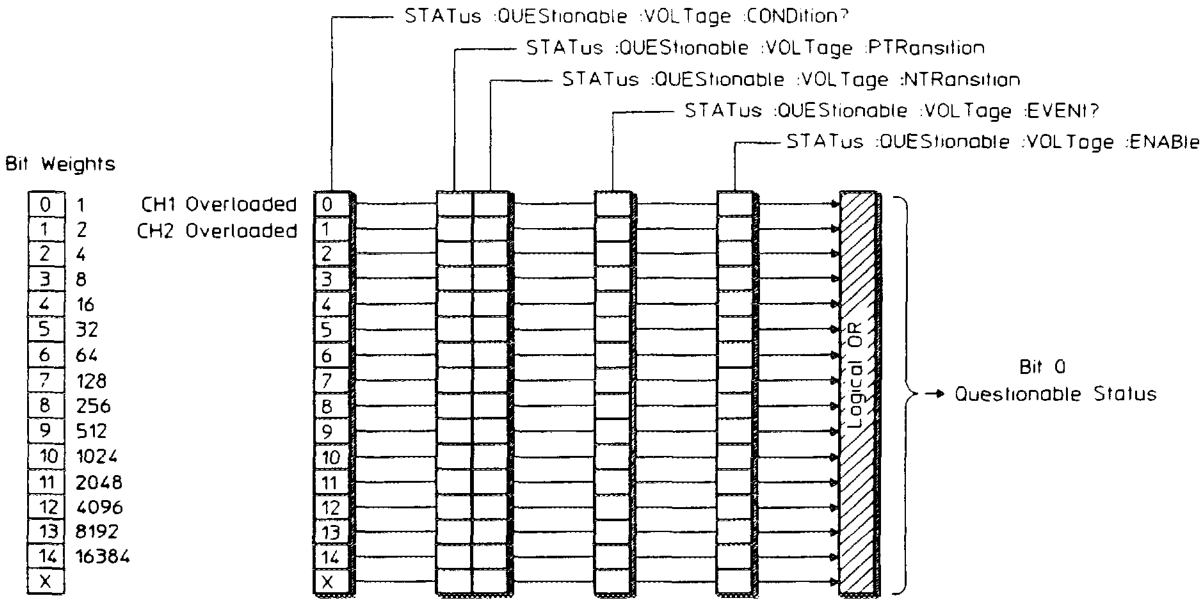

STATus :QUEStionable :VOLTage :CONDition?
STATus :QUEStionable :VOLTage :PTRansition
STATus :QUEStionable :VOLTage :NTRansition
STATus :QUEStionable :VOLTage :EVENt?
STATus :QUEStionable :VOLTage :ENABle
Bit Weights
0 1
1 2
2 4
3 8
4 16
5 32
6 64
7 128
8 256
9 512
10 1024
11 2048
12 4096
13 8192
14 16384
X
CH1 Overloaded
CH2 Overloaded
Logical OR
Bit 0
Questionable Status

Figure 5-9. The Questionable Voltage Register Set

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

- Channel 1 Overload (bit 0) is set to 1 when any input signal exceeds the current channel 1 input range.
- Channel 2 Overload (bit 1) is set to 1 when any input signal exceeds the current channel 2 input range.

Figure 5-9 also shows the commands you use to read and write the Questionable Voltage registers. See chapter 25 for more information about these commands.

Standard Event Register Set

The Standard Event register set monitors HP-IB errors and synchronization conditions. See figure 5-10.

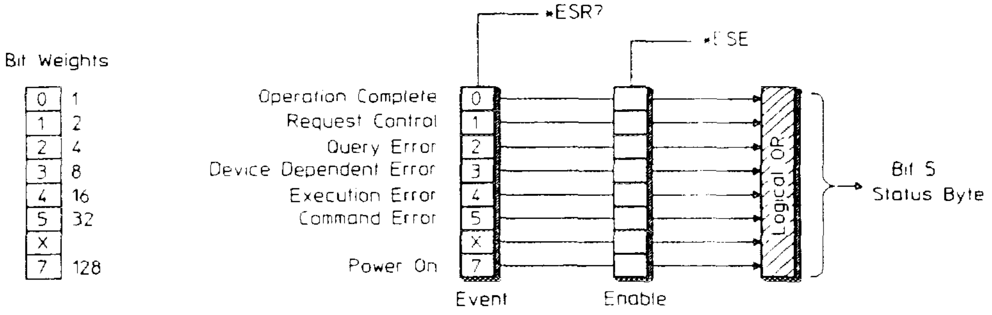


Figure 5-10. The Standard Event Register Set

The Standard Event register set does not conform to the general status register model described at the beginning of this chapter. It contains only two registers: the Standard Event event register and the Standard Event enable register. The Standard Event 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 enable register is the same as other enable registers.

Bits in the Standard Event event register are set to 1 under the following conditions:

- Operation Complete (bit 0) is set to one when the following two events occur (in the order listed):
  - You send the *OPC command to the analyzer.
  - The analyzer completes all pending overlapped commands (see "Synchronization" in chapter 2).

- Request Control (bit 1) is set to 1 when both of the following conditions are true:
  - The analyzer is configured as an addressable-only HP-IB device (see "Controller Capabilities" in chapter 2).
  - 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 to 1 when the command parser detects a query error.

- Device Dependent Error (bit 3) is set to 1 when the command parser detects a device-dependent error.

- Execution Error (bit 4) is set to 1 when the command parser detects an execution error.

- Command Error (bit 5) is set to 1 when the command parser detects a command error.

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

Figure 5-10 also shows the commands you use to read and write the Standard Event registers. See chapter 25 for more information about these commands.

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 current HP Instrument BASIC program. See figure 5-11.

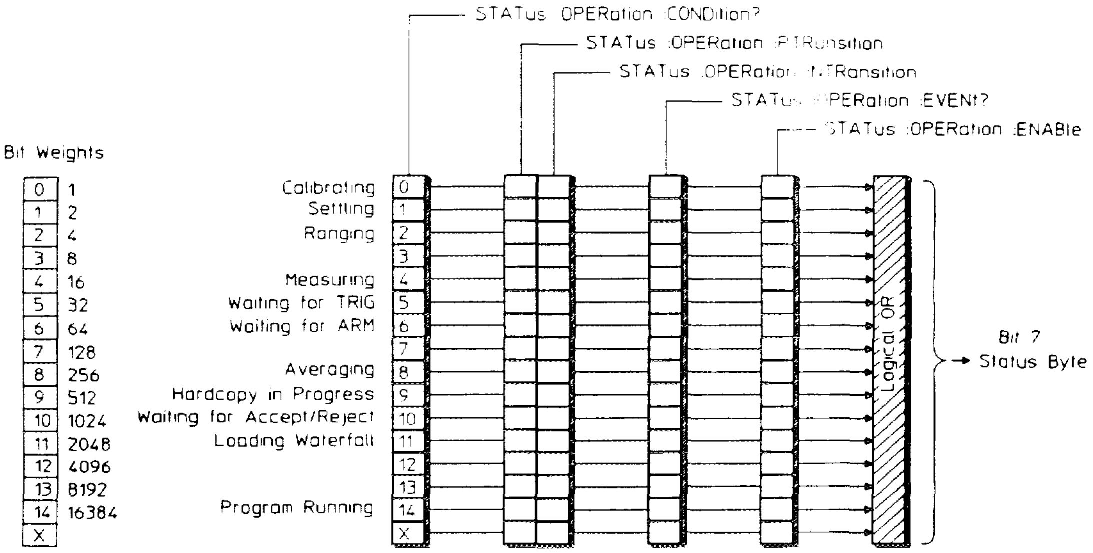


Figure 5-11. The Operational Status Register Set

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

- Calibrating (bit 0) is set to 1 while the self-calibration routine is running.
- Settling (bit 1) is set to 1 while the measurement hardware is settling.
- Ranging (bit 2) is set to 1 while the input range is changing.
- Measuring (bit 4) is set to 1 while the analyzer is collecting data for a measurement.
- Waiting for TRIG (bit 5) is set to 1 when the analyzer is ready to accept a trigger signal from one of the trigger sources. (If a trigger signal is sent before this bit is set, the signal is ignored.)
- Waiting for ARM (bit 6) is set to 1 when both of the following conditions are true:
  - Manual arming is selected.
  - The analyzer is ready to be armed.

  (If you send the ARM:IMM command before this bit is set, the command is ignored.)

- Averaging (bit 8) is set to 1 while the analyzer is averaging measurement data. If averaging is disabled ([SENSe:]AVERage[:STATe] OFF) this bit is set to 1 whenever the Measuring bit (bit 4) is set to 1 during data collection.

- Hardcopy In Progress (bit 9) is set to 1 while the analyzer is performing a print or plot operation.

- Waiting for Accept/Reject (bit 10) is set to 1 while the analyzer is waiting for a response during preview averaging ([SENSe:]AVERage:PREView MANual).

- Loading Waterfall (bit 11) is set to 1 while the analyzer is collecting the specified number of traces for a waterfall display.

- Program Running (bit 14) is set to 1 while the current HP Instrument BASIC program is running.

Figure 23-1 under the [SENSe:]AVERage[:STATe] command illustrates the transition of the bits in the Operational Status condition register.

Figure 5-11 also shows the commands you use to read and write the Operational Status registers. See chapter 25 for more information about these commands.

User Status Register Set

The User Status register set detects STATus:USER:PULSe commands. See figure 5-12.

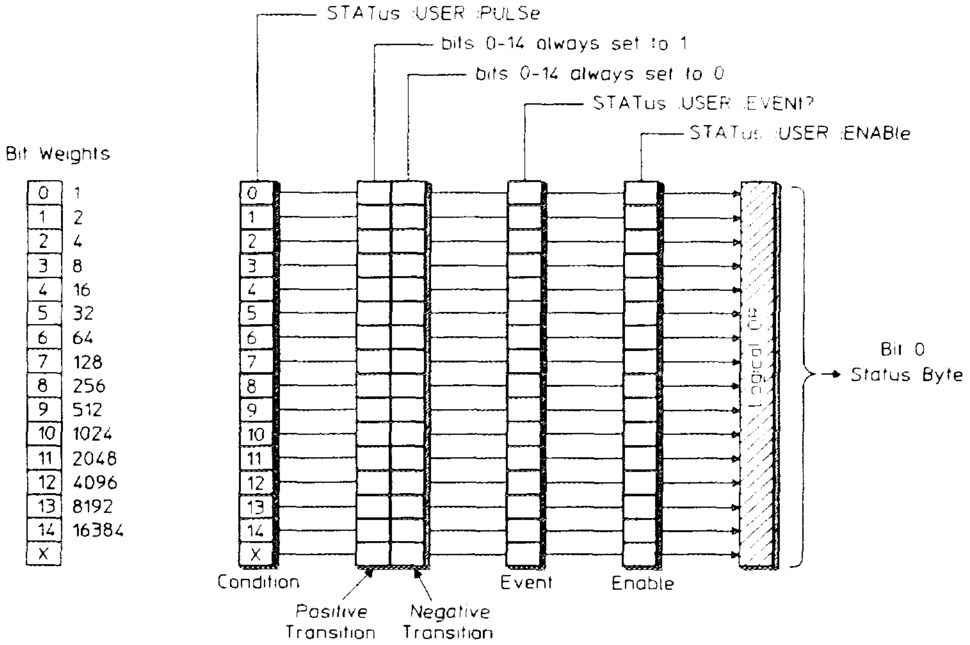


Figure 5-12. The User StatusRegister Set

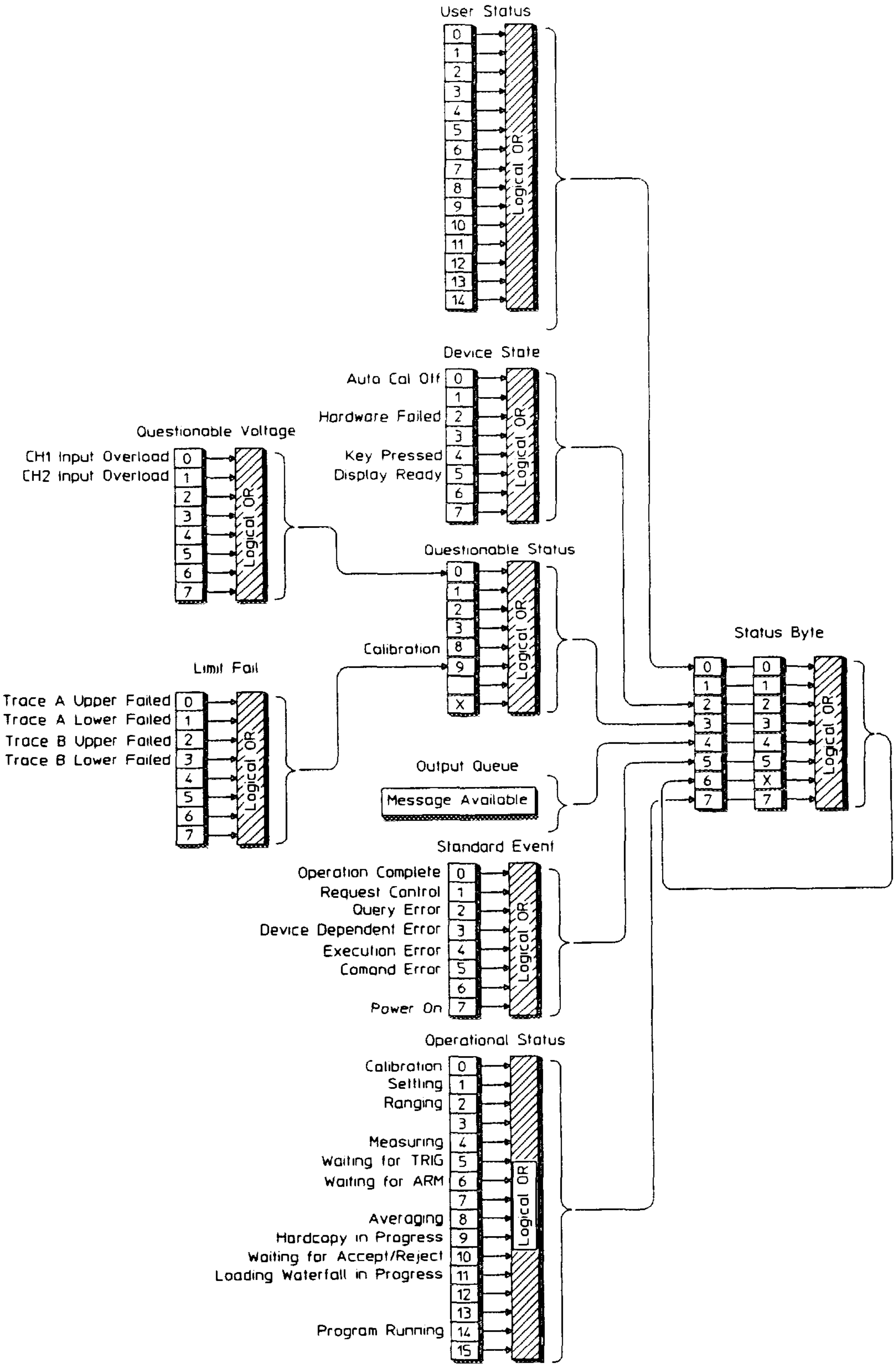
The User Status register set conforms to the general status register model (described at the beginning of this chapter) with the following exceptions:

- You can write (but not read) the condition register.
- You cannot write or read the transition registers.
- Bits in the positive transition register are always set to 1.
- Bits in the negative transition register are always set to 0.
- Bit 15 is not available. It is always set to 0.

Bits in the User Status condition register are normally set to 0, but are set to 1 (briefly) when you send a STAT:USER:PULS command. If you send STAT:USER:PULS 32, bit 5 of the condition register is pulsed high ($2^5 = 32$).

Figure 5-12 also shows the commands you use to read or write the User Status registers. See chapter 25 for more information about these commands.

HP 35665A Register Set Summary



6

Example Programs

This chapter contains listings of example programs written for the HP 35665A. They demonstrate many important programming concepts, including:

- Generating service requests (SRQs)
- Measurement synchronization
- Passing control
- Transferring data

Many of the programs are written in HP BASIC for use on an HP Series 200/300 computer. They contain numerous comments to make them easily adaptable to other languages or programs. The listings also contain example programs written in other languages; specifically MS QuickBASIC and MS QuickC.

The listings, which are organized alphabetically by name, are as follows:

ATEGRAPH.C

MS QuickC
Demonstrates using a PC with an HP-IB card as an external controller for the HP 35665A. The program runs an averaged measurement, uploads the trace results and displays them graphically on the CRT (VGA).

DOWNLOAD

Demonstrates downloading an HP Instrument BASIC ASCII program into the HP 35665A's HP Instrument BASIC memory and optionally storing it on the HP 35665A's default disk drive. This program resides on an external HP BASIC controller.

DTXFRA

Demonstrates transferring data between an HP Instrument BASIC program and a program running in an HP BASIC external controller. This program must reside on the HP 35665A's default drive. The controller contains the program DTXFRB which initiates the transfer.

DTXFRB

Demonstrates transferring data between an HP Instrument BASIC program and a program running in an HP BASIC external controller. This program runs in the controller to initiate the transfer. The program DTXFRA must reside on the HP 35665A's default drive.

EX35665.BAS

MS QuickBASIC 4.5
Demonstrates uploading pre- and post-coordinate transform data. The program manages an SRQ for the AVERAGE_COMPLETE bit, the MEASUREMENT_COMPLETE bit and the WAITING_FOR_ARM bit.

GETMATH

Demonstrates using the CALCulate:MATH:DATA query to transfer a complete set of math definitions.

LOADMATH

Demonstrates using the CALCulate:MATH:DATA command to transfer a complete set of math definitions.

MANARM

Demonstrates using the Standard Event Register to detect a WAITING_FOR_ARM event and generates an SRQ. The program handles the SRQ interrupt and allows you to arm the measurement.

OPC_SYNC

Demonstrates synchronizing a program and the HP 35665A using the *OPC statement. The *OPC statement sets the OPERATION_COMPLETE bit in the Standard Event Status Register when all pending operations have completed. The Status Register is masked to generate an SRQ. The program handles the service request interrupt.

OPCQ_SYNC

Demonstrates synchronizing a program and the HP 35665A. The program issues an *OPC? query and then pauses (on an ENTER statement) while it waits for the pending HP-IB operations to complete.

TWO_CTLR

Demonstrates using an external controller to download an HP Instrument BASIC program, execute it, and query variables in the HP Instrument BASIC program.

UPLOAD

Demonstrates uploading an HP Instrument BASIC program from the HP 35665A and storing it in an HP BASIC ASCII file. This program resides on an external HP BASIC controller.

WAI_SYNC

Demonstrates synchronizing program and instrument using the *WAI statement. The program terminates before the HP-IB statements finish executing.

ATEGRAPH.C

```
/***************************************************/
/* ATEGRAPH.C                                      */
/* HP 35665A ATE example graphics program          */
/*                                                 */
/* Demonstrates using PC as external controller    */
/* using Microsoft QuickC. Continuously uploads    */
/* and displays averaged measurement traces.       */
/*                                                 */
/* Run on IBM compatible with VGA monitor          */
/* Compile using Microsoft QuickC 2.0              */
/* Requires CLHPIB.LIB from HP-IB Command Library */
/* 82335 Release A.00.01 and HP-IB interface card */
/* Also requires QuickC library GRAPHICS.LIB       */
/***************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <graph.h>
#include <conio.h>
#include "CHPIB.H"
#include "CFUNC.H"

#define ISC 7L
#define HP35665A 711L
#define BSIZE 401


#define TRACE_COLOR 3       /* cyan   */
#define COLOR_OFF 0         /* black  */
#define GRID_COLOR 6        /* brown  */
#define BORDER_COLOR 4      /* red    */
#define TEXT_COLOR  6       /* yellow */
```

```
/**************************************************/
/* Global variable definitions                    */
/**************************************************/

/* dimension used for graphics functions */
int x0 = 0;
int y0 = 0;
int xLen = 400;
int yLen = 360;    /* VGA  display.
                                              (Use 260 for EGA)      */
float yMin, yMax ; /* minimum and maximum
                                              Y-axis display values */

struct videoconfig vc;

/* Trace display buffers */
double dispBuf[BSIZE-1], prevTrace[BSIZE-1];

void error_handle(error, routine)
int error;
char routine[80];
   {
     if (error != NOERR)
       {
                    printf("HPIB error in call to %s: %d, %s\n", routine,
                                         error, errstr(error));
                    IOLOCAL(HP35665A);
                    exit(1);
       }
     return;
   }

/**************************************************/
/* Simplified HP-IB output function               */
/**************************************************/
void outputcmd(char *strn)
   {
     int len;
     char s[80];
     strcpy(s,strn);
     len = strlen(s);
     error_handle(IOOUTPUTS(HP35665A,s,len),"IOOUTPUTS");
   }
```

```
/**************************************************/
/* Set up graphics display parameters            */
/**************************************************/
void InitGraphics()
  {
     _setvideomode(_DEFAULTMODE);
     _clearscreen( _GCLEARSCREEN);
/*   _setvideomode(_ERESCOLOR);            EGA color mode */
     _setvideomode(_VRES16COLOR);
     _getvideoconfig(&vc);
     _setviewport(120,40,120+xLen,40+yLen);
     _settextcolor(TEXT_COLOR);
     _setcolor(BORDER_COLOR);
     _rectangle(_GBORDER, x0,y0,xLen,yLen);
  }

/**************************************************/
/* Sets up a measurement on the HP 35665a        */
/**************************************************/
void SetupMeas()
    {
     outputcmd("SYST:PRES");            /* preset            */
     outputcmd("FORM:DATA REAL, 64");   /* definite length block data  */
     outputcmd("CAL:AUTO OFF");         /* autocal off       */
     outputcmd("SOUR:VOLT 1.0 Vrms");   /* source level      */
     outputcmd("SOUR:FREQ 48 Khz");     /* source frequency  */
     outputcmd("OUTP ON");              /* source on         */
     outputcmd("AVER:COUN 5");          /* 5 averages        */
     outputcmd("ABORT;:INIT; *WAI");    /* start measurement */
     outputcmd("DISP:WIND1:TRAC:Y:AUTO ONCE");  /* autoscale */

     outputcmd("AVER ON; *WAI");        /* averaging on      */
     _settextposition(28,1);
     printf("                           ");
     printf("Press [Enter] to Quit");
  }
```

```
/***************************************************/
/* Queries HP 35665a for units and prints correct  */
/* form of unit on screen at specified coordinates */
/***************************************************/
void PrintUnit(char *s, int x, int y)
   {
     int n = 20;
     char returnStr[20];
     char dbvrms[] = "\"DBVRMS\"";
     char hz[] = "\"HZ\"";
     char khz[] = "\"KHZ\"";

     outputcmd(s);
     error_handle(IOENTERS(HP35665A,returnStr,&n),"IOENTERS");
     _settextposition(x,y);
     if (!strncmp(returnStr,dbvrms,6)) printf("dBVrms\n");
     if (!strncmp(returnStr,hz,2)) printf("Hz\n");
     if (!strncmp(returnStr,khz,3)) printf("kHz\n");
  }

/***************************************************/
/* Queries HP 35665a and prints numeric results    */
/* on screen at specified coordinates              */
/***************************************************/
void PrintNumber(char *s, int y, int x)
{
     char returnStr[20];
     int n = 20;
     float f;

     outputcmd(s);
     error_handle(IOENTERS(HP35665A,returnStr,&n),"IOENTERS");
     f = atof(returnStr);
     _settextposition(y,x);
     printf("%6.0f\n",f);
}
```

```
/***************************************************/
/*  Labels graphic display with values from  the */
/*  current HP 35665a display                    */
/***************************************************/
void Label()
   (
   char s[20];
   int top - 2;                 /* top of graph       */
   int left - 9;                /* left edge of graph */
   int bottom - 27;             /* bottom of graph    */
  _settextposition(top,30);
   printf("ATE GRAPH DEMONSTRATION");
   PrintNumber("DISP:WIND1:TRAC:Y:TOP?", 4, left);
   PrintUnit("DISP:WIND1:TRAC:Y:TOP? UNIT", 5, left);

   PrintNumber("DISP:WIND1:TRAC:Y:PDIV?", 12, left);
   PrintUnit("DISP:WIND1:TRAC:Y:TOP? UNIT", 13, left);
   _settextposition(14,left+2);
   printf("/Div");

   PrintNumber("DISP:WIND1:TRAC:Y:BOTT?", 24, left);
   PrintUnit("DISP:WIND1:TRAC:Y:BOTT? UNIT", 25, left);

   PrintNumber("DISP:WIND1:TRAC:X:LEFT?", bottom, 11 );
   PrintUnit("DISP:WIND1:TRAC:X:LEFT? UNIT", bottom, 17);

   PrintNumber("DISP:WIND1:TRAC:X:RIGHT?", bottom, 59 );
   PrintUnit("DISP:WIND1:TRAC:X:RIGHT? UNIT", bottom, 65 );

   )


/*****************************************************/
/*  Scales yin between yMax and yMmin global values */
/*****************************************************/
int ScalePoint(float yin)
   (
                    int lowerLimit - -1000;
                    float yvall;
                    int   yval2;

                    /* normalize y value */
                    if (yin > lowerLimit) (
                       yvall - (yin-yMin)/(yMax-yMin);
                       yval2 - (int)(yvall*yLen);
                       return(yval2);
      )
       /* otherwise yin - log(zero) so set to Ymin */
       else return(yMin);
  )
```

```
/*************************/
/* Draws a 10 * 10 grid  */

/*************************/
void drawGrid()
  {
       int i;
       int YDivs = 10;
       int XDivs = 10;
       int yRow = y0;

       _setcolor(GRID_COLOR);

       _setlinestyle(0x8888);        /* pattern = 1000100010001000 */
       /* draw vertical lines */
       for(i=x0+xLen/XDivs;i<xLen;i+=xLen/XDivs){
                      _moveto(i,y0);
                      _lineto(i,yLen);
       }
       /* draw horizontal lines */
       for(i=y0+yLen/YDivs;i<yLen;i+=yLen/YDivs){
                      _moveto(x0, i);
                      _lineto(xLen, i);
       }
       _setlinestyle(0xFFFF);
  }

/*******************************/
/* Reads a block from trace A  */
/*******************************/
void GetTrace()
  {
     char s[11];
     int j = 11;
     int i;
     int elements;
     int swap;

     swap = sizeof(double);
     elements = BSIZE * swap;
     /* save copy of current trace */
     for (i=0; i < BSIZE; ++i)
                        prevTrace[i] = dispBuf[i];
     /* get new trace */
     outputcmd("ABORT;:INIT;*WAI");
     outputcmd("CALC1:DATA?");
     error_handle(IOENTERAB(HP35665A,dispBuf,&elements,swap),"IOENTERAB");
     /* read LF character */
     error_handle(IOENTERS(HP35665A,s,&j),"IOENTERS");
  }
```

```
/*****************************************************/
/*  Plots the points in display buffer to the screen */
/*****************************************************/
void Graph()
 {
          int x1, x2;

          drawGrid();
          _moveto(0,yLen-ScalePoint(prevTrace[0]));
          for (x1 = 0, x2 = 0; x1 < BSIZE-3; ) {
            _setcolor(COLOR_OFF);     /* erase previous point */
            _moveto(x1++,yLen-ScalePoint(prevTrace[x1++]));
            _lineto(x1,yLen-ScalePoint(prevTrace[x1]));
            _setcolor(TRACE_COLOR);     /* set line color */
            _moveto(x2++,yLen-ScalePoint(dispBuf[x2++]));

            _lineto(x2,yLen-ScalePoint(dispBuf[x2]));
    }
          _setcolor(BORDER_COLOR);     /* redraw border */
          _rectangle(_GBORDER, x0,y0,xLen,yLen);
 }

/***************************************************/
/* Returns Ymin and Ymax into global variables   */
/***************************************************/
void GetYlimits()
  {
   char returnStr[20];
   int n = 20;

   outputcmd("DISP:WIND1:TRAC:Y:TOP?");
   error_handle(IOENTERS(HP35665A,returnStr,&n),"IOENTERS");
   yMax = atof(returnStr);
   outputcmd("DISP:WIND1:TRAC:Y:BOTT?");
   error_handle(IOENTERS(HP35665A,returnStr,&n),"IOENTERS");
   yMin = atof(returnStr);
  }

/****************************************************/
/*  Gets yMin and yMax values and draws first trace */
/****************************************************/
void InitTrace()
  {
   /* called first so that prevTrace has current data */

   int i;

   GetTrace();
   for (i=0; i < BSIZE; ++i)
            prevTrace[i] = dispBuf[i];
   GetYlimits();
   Graph();
  }
```

```
/********************************/
/*  Resets HP-IB and graphics   */
/********************************/
void Cleanup()
   {
     error_handle(IOCLEAR(HP35665A),"IOCLEAR");
     error_handle(IOLOCAL(HP35665A),"IOLOCAL");
     _setvideomode(_DEFAULTMODE);
   }

main()
  {
     char *a;
     printf("\n\nConnect HP 35665A source to channel 1 and press <Enter> \n");
     printf("to continue program... \n");

     scanf(a);
     InitGraphics();
     SetupMeas();
     Label();
     InitTrace();
     while(!kbhit()) {
                     GetTrace();
                     Graph();
     }
     Cleanup();
  }
```

DOWNLOAD

```
10  ! ------------------------------------------------------------
20  ! HP BASIC Program: DOWNLOAD
30  ! This program downloads a file into an HP 35665A Instrument
40  ! BASIC program from an external controller.
50  ! The downloaded program must be an HP BASIC ASCII type file.
60  ! It will NOT work with DOS or HP-UX (untyped) files.
70  ! ------------------------------------------------------------
80  !
90   DIM Load_file$[20],Prog_line$[256],Command$[80],Name$[10]
100  DIM Diskname$[20],Answer$[2]
110  ASSIGN @Hp35665a TO 711
120 !
130 ! The file of program to download must be an ASCII file.
140 !
150  INPUT "ENTER NAME OF FILE TO DOWNLOAD ",Load_file$
160  ASSIGN @File TO Load_file$
170  INPUT "WHAT PROGRAM [1..5] DO YOU WANT TO DOWNLOAD IT TO?",Prognumber
180  OUTPUT @Hp35665a;"PROG:NAME PROG"&VAL$(Prognumber)
190  OUTPUT @Hp35665a;"PROG:DEL:ALL"
200  ON ERROR GOTO End_load
210  OUTPUT @Hp35665a;"PROG:DEF #0";
220  LOOP
230    ENTER @File;Prog_line$
240    PRINT Prog_line$
250    OUTPUT @Hp35665a;Prog_line$
260  END LOOP
270 !
280 End_load: !
290  OUTPUT @Hp35665a;CHR$(10) END
300  INPUT "SAVE PROGRAM TO INSTRUMENT'S DEFAULT DRIVE? [Y/N]",Answer$
310  IF UPC$(Answer$)="Y" THEN
320    INPUT "ENTER NAME FOR DISK FILE",Diskname$
330    OUTPUT @Hp35665a;"MMEM:STORE:PROGRAM '"&Diskname$&"'"
340  END IF
350  END
```

DTXFRA

```
10   !BASIC program:  DTXFRA -- Data transfer BASIC to BASIC
15 !------------------------------------------------------------------
20 ! This program demonstrates how to transfer data to and from an
30 ! external controller.  In this example a catalog listing is transferred
40 ! from the HP35665A to the external controller.  For more information
50 ! look at the program listing for 'DTXFRB'
60 !
70 ! This program is intended to be executed with HP Instrument BASIC.
80 !------------------------------------------------------------------
90   DIM Directory$(1:100)[85] !Create string array for catalog
100 !
110   Host=721                  !Address for external controller
120 !
130   ON ERROR GOTO 140         !Loop until control is passed to the HP35665A
140   ENTER Host;Stor_dev$      !Address Host to talk, read device to catalog
150   OFF ERROR
160 !
170   DISP "Reading catalog..."
180   CAT Stor_dev$ TO Directory$(*)!Catalog into the string array
190 !
200   DISP "Transferring data..."
210   OUTPUT Host;Directory$(*)     !Address Host to listen, write array
220 !
230   PASS CONTROL Host             !Pass control back to host
240   DISP "DONE"
250   END
```

DTXFRB

```
10    !BASIC program:  DTXFRB -- Data transfer BASIC to BASIC
20    !---------------------------------------------------------------------
30    ! This program demonstrates how to transfer data from an HP Instrument
40    ! BASIC program.  This program, which runs on the computer, loads a
50    ! a program into the HP35665A, runs it, and then gives it control of
60    ! the bus. This program then acts as a device on the bus; sending and
70    ! receiving data. Before running this program, a disk with the program
80    ! 'DTXFRA' should be in the HP35665A's internal drive.
90    !---------------------------------------------------------------------
100   Scode=7                 !Select code for interface
110   Address=11              !Address for HP35665A
120   Hp35665a=Scode*100+Address
130   CLEAR Hp35665a
140   OUTPUT Hp35665a;"*CLS"  !Clear the EVENT registers
150   CLEAR SCREEN            !Clear the display
160   !
170   DIM Directory$(1:100)[85] !Array to hold catalog listing
180   !
190   INPUT "Put disk with program 'DTXFRA' into the HP35665A. Press <E>
                    ",A$
200   DISP "Loading program on HP35665A..."
210   OUTPUT Hp35665a;"MMEM:LOAD:PROG 'INT:DTXFRA'" !Load BASIC program from
                    disk
220   OUTPUT Hp35665a;"*OPC?"
230   ENTER Hp35665a;Opc     !Wait here until program loaded
240   OUTPUT Hp35665a;"*ESR?" !Read the EVENT STATUS reg
250   ENTER Hp35665a;Esr
260   IF Esr0 THEN           !Have any errors occurred
270     BEEP
280     DISP "Error occurred while loading 'DTXFRA'...Cannot continue
program."
290     STOP
300   END IF
310   !
320   OUTPUT Hp35665a;"*PCB 21" !Set pass control back address
330   !                                      to HP-IB address for controller
340   DISP "Running the program..."
350   OUTPUT Hp35665a;"PROG:STAT RUN"!Start the program
360   PASS CONTROL Hp35665a          !Give program control of bus
370   !
380   OUTPUT Scode;":INTERNAL"       !Wait until addressed to talk
390   DISP "Reading data..."
400   ENTER Scode;Directory$(*)      !Wait until addressed to listen
410   !
420   FOR I=1 TO 100                 !Print the catalog
430     IF LEN(Directory$(I)) > 0 THEN PRINT Directory$(I)
440   NEXT I
450   !
```

```
460   ON ERROR GOTO 470 !Loop until control passed back
470   LOCAL Hp35665a
480   DISP ""
490   END
```

EX35665.BAS

```
'************************************************************************
' HP 35665A Dynamic Signal Analyzer example program
' Uses QuickBASIC 4.5 and HP-IB Command Library 82335 Release A.00.01
' This program demonstrates masking HP 35665A status registers to
' generate an SRQ on one of three events.
' It also demonstrates handling the SRQ and uploading data from the
' analyzer to the QuickBASIC program for both display and trace data.
'*************************************************************************
DECLARE SUB SETUPMEAS (ADDR&, Event)
DECLARE SUB UPLOAD (ADDR&, Dtype%)
DECLARE SUB OUTPUTCMD (ADDR&, CODE$)
DECLARE FUNCTION WAITFORSRQ% (ISC&, ADDR&)
DECLARE SUB HANDLESRQ (ADDR&, Event, Dtype%)
COMMON MEASDATA#()

' $INCLUDE: 'QBSETUP'
' $DYNAMIC


CONST ADDR& = 711
CONST ISC& = 7

' measurement SRQ types
CONST measuring = 16       ' Generate SRQ at measurement complete state
CONST averaging = 256      ' Generate SRQ at averaging complete state
CONST waterfall = 2048     ' Generate SRQ at waterfall complete state

' upload data types
CONST RawData = 1         ' Upload data from data reg (raw trace data)
CONST DispData = 2        ' Upload display data (coordinate-transformed)
CONST WaterDisp = 3       ' Upload waterfall trace data

' default values
NumberOfMeas = 1
DataType% = 1
Event = measuring

' needed by IOENTERS function:
MAX.STR% = 5
ACTUAL% = 0
INFO$ = SPACE$(MAX.STR%)

CLS 0                          ' Clear screen

PRINT "Enter measurement mode: (1) No average (2) Averaged (3) Waterfall"
INPUT measmode
SELECT CASE measmode
  CASE 1: Event = measuring
  CASE 2: Event = averaging
  CASE 3: Event = waterfall
END SELECT


IF measmode < 3 THEN
  PRINT "Enter data type:(1) trace data (raw) (2) display data"
  INPUT DataType%
ELSE
  DataType% = 3
END IF

PRINT "Enter number of measurements to upload"
INPUT NumberOfMeas

' Preset analyzer and pause program until preset completed
CALL OUTPUTCMD(ADDR&, "SYST:PRES;*WAI")
PRINT "Waiting for initialization to conclude..."
CALL OUTPUTCMD(ADDR&, "*OPC?")
CALL IOENTERS(ADDR&, INFO$, MAX.STR%, ACTUAL%)

' Pause measurement to bring measuring, averaging and waterfall bits low
CALL OUTPUTCMD(ADDR&, "INIT:CONT OFF;*WAI")
' Clear status registers
CALL OUTPUTCMD(ADDR&, "*CLS")
CALL OUTPUTCMD(ADDR&, "*OPC?")
CALL IOENTERS(ADDR&, INFO$, MAX.STR%, ACTUAL%)

' Set data format to definite length block format with 64 bit word length
CALL OUTPUTCMD(ADDR&, "FORM:DATA REAL, 64")
' Enable status byte for operation status register events
CALL OUTPUTCMD(ADDR&, "*SRE 128")
' Mask specified operation status register event
CALL OUTPUTCMD(ADDR&, "STAT:OPER:ENAB" + STR$(Event))
' Enable operation status transition register for negative transition
CALL OUTPUTCMD(ADDR&, "STAT:OPER:NTR" + STR$(Event))
CALL OUTPUTCMD(ADDR&, "STAT:OPER:PTR 0")

CALL SETUPMEAS(ADDR&, Event)
PRINT "Starting measurement..."

' NOTE: The INIT command forces the averaging, measuring and waterfall bits
'        low then high. This negative transition can cause an early SRQ.
'        To avoid this issue an "INIT:CONT OFF" command before setting
'        up enable registers to make sure these bits are already low
'        when the INIT command is issued.
```

```
CALL OUTPUTCMD(ADDR&, "INIT;*WAI")

FOR I = 1 TO NumberOfMeas
  BEEP
  E = 0
  WHILE E < > Event
    E = WAITFORSRQ(ISC&, ADDR&)
    '  To increase update rates for non-averaged display data
    '  put this function inline
  WEND
  CALL UPLOAD(ADDR&, DataType%)
  IF (Event = 256) OR (Event = 2048) THEN
        ' average or waterfall complete so restart measurement
        CALL OUTPUTCMD(ADDR&, "INIT;*WAI")
  END IF
NEXT

END

REM $STATIC
SUB OUTPUTCMD (ADDR&, CODE$) STATIC
' ******************************************** '
Sends a string to the specified address '
*******************************************
SHARED PCIB.ERR, PCIB.BASERR, NOERR
   L% = LEN(CODE$)
   CALL IOOUTPUTS(ADDR&, CODE$, L%)
   IF PCIB.ERR < > NOERR THEN ERROR PCIB.BASERR
END SUB

SUB SETUPMEAS (ADDR&, Event) '

******************************************************* '
Sets up measurement parameters for selected measurement '
*******************************************************
SELECT CASE Event
     CASE waterfall
       CALL OUTPUTCMD(ADDR&, "CALC2:FEED 'XFR:POW 1';*WAI")
       CALL OUTPUTCMD(ADDR&, "CALC:WAT:COUNT 5;*WAI")
       ' when uploading waterfall trace data, this will produce 5
       ' Power Spectrum traces ( 5 * 512 = 2560 points )     CASE averaging
       CALL OUTPUTCMD(ADDR&, "FREQ:SPAN 12800")
       CALL OUTPUTCMD(ADDR&, "AVER ON")
     CASE measuring
       ' set to narrow span to demonstrate timing of handshake
       CALL OUTPUTCMD(ADDR&, "FREQ:SPAN 400")
       ' set to zoomed Linear measurement type:
       CALL OUTPUTCMD(ADDR&, "FREQ:STAR 1000")
       CALL OUTPUTCMD(ADDR&, "CALC:FEED 'XFR:POW:LIN 1'")
       ' if uploading raw trace data, this will produce 1024 points
       ' ( 512 complex pairs ), otherwise there will be 401 display points
   END SELECT
END SUB

SUB UPLOAD (ADDR&, DataType%)
' ******************************************
'Uploads data from analyzer to array
'******************************************
SHARED PCIB.ERR, PCIB.BASERR, NOERR
SHARED MEASDATA#()
   MAX% = 2
ACTUAL% = 0
INFO$ = SPACE$(MAX%)
   s% = 8

   SELECT CASE DataType%
     CASE 1
        PRINT "uploading trace data from D1..."
        CALL OUTPUTCMD(ADDR&, "INIT:CONT OFF;*WAI")  ' pause measurement
        CALL OUTPUTCMD(ADDR&, "TRAC D1, TRAC1;*WAI") ' copy trace to D1
        CALL OUTPUTCMD(ADDR&, "TRAC:DATA? D1;*WAI")  ' request D1 data
     CASE 2
        PRINT "uploading display data..."
        CALL OUTPUTCMD(ADDR&, "CALC:DATA?")          ' request display data
     CASE 3
        PRINT "uploading waterfall data from W1..."
        CALL OUTPUTCMD(ADDR&, "INIT:CONT OFF;*WAI")  ' pause measurement
        CALL OUTPUTCMD(ADDR&, "TRAC:WATERFALL W1, TRAC2;*WAI") ' copy to W1
        CALL OUTPUTCMD(ADDR&, "TRAC:WATERFALL:DATA? W1") ' request W1 data
   END SELECT

   ' Read # of characters in header
   CALL IOENTERS(ADDR&, INFO$, MAX%, ACTUAL%)
   MAX% = VAL(MID$(INFO$, 2, 1))
   INFO$ = SPACE$(MAX%)
   ACTUAL% = 0
   ' Read number of bytes in file from header
   CALL IOENTERS(ADDR&, INFO$, MAX%, ACTUAL%)
   ' convert string to number
   MAX.BYTE = VAL(LEFT$(INFO$, MAX%))

   ' IOENTERAB function only accepts integers for byte count parameter
   IF MAX.BYTE < 32768 THEN
      ACTUAL% = 0
      DIMSIZE% = (MAX.BYTE / 8) - 1
      ' dimension the MEASDATA# array to the proper size
      REDIM MEASDATA#(DIMSIZE%)
      ' convert MAX.BYTE to an integer
      MAX.BYTE% = MAX.BYTE
      CALL IOENTERB(ADDR&, SEG MEASDATA#(0), MAX.BYTE%, ACTUAL%, s%)
      PRINT ACTUAL% / 8; "  points uploaded"
      ACTUAL% = 0
      ' read termination character
      CALL IOENTERS(ADDR&, INFO$, MAX%, ACTUAL%)
   ELSE
      BEEP
      PRINT "DATA ARRAY TOO LARGE TO UPLOAD"
   END IF
END SUB

FUNCTION WAITFORSRQ% (ISC&, ADDR&) STATIC
' ******************************************
'
' ******************************************
SHARED PCIB.ERR, PCIB.BASERR, NOERR
   SRQ% = 1
   PRINT
   PRINT "Waiting for SRQ..."
CHECKSTAT:   CALL IOSTATUS(ISC&, SRQ%, STATUS%)
   IF PCIB.ERR < > NOERR THEN ERROR PCIB.BASERR
   IF STATUS% = 0 THEN GOTO CHECKSTAT
   PRINT "SRQ Received"
   CALL IOSPOLL(ADDR&, RESPONSE%)
   IF PCIB.ERR < > NOERR THEN ERROR PCIB.BASERR

   PRINT "Serial Poll Response = ", RESPONSE%
   IF RESPONSE% = 192 THEN
      CALL OUTPUTCMD(ADDR&, "STAT:OPER:EVEN?")
      CALL IOENTER(ADDR&, R)
      IF PCIB.ERR < > NOERR THEN ERROR PCIB.BASERR
      PRINT "STAT:OPER:EVEN? = ", R
   ELSE
      GOTO CHECKSTAT
   END IF
   WAITFORSRQ% = R
END FUNCTION
```

GETMATH

```
1 ! This is a short program to show how to use the CALC:MATH:DATA? query to
2 ! get an entire math setup from the analyzer
3 !
4 INTEGER Type    ! file type
5 DIM Junk$[6]
6 DIM F$(1:5)[270]    ! array of function definitions
7 DIM K(1:10)    ! constants
8 !
9 ASSIGN @Dut TO 800
10 OUTPUT @Dut;"FORM:DATA REAL,64"
11 OUTPUT @Dut;"CALC:MATH:DATA?" ! query for math definition structure
12 ENTER @Dut USING "#,6A";Junk$ ! get rid of the "#41432" at front of block
13 ASSIGN @Dut;FORMAT OFF
14 ENTER @Dut;Type    ! get 1503 file type
15 ASSIGN @Dut;FORMAT ON
16 !ENTER @Dut USING "#,1350A";func$     ! get function definitions
17 FOR I=1 TO 5
18   ENTER @Dut USING "#,270A";F$(I)
19   FOR J=1 TO 270      ! trim the trailing nulls
20     IF (F$(I)[J;1]=CHR$(0)) THEN
21       F$(I)=F$(I)[1,J]
22       GOTO Strip
23     END IF
24   NEXT J
25 Strip:     ! striped of trailing nulls
26   IF (LEN(F$(I))=0) THEN  ! print the function definitions
27     PRINT "F"&VAL$(I)&" is undefined"
28   ELSE
29     PRINT "F";VAL$(I);" = ";F$(I)
30   END IF
31 NEXT I
32 !
33 ASSIGN @Dut;FORMAT OFF
34 ENTER @Dut;K(*)    ! get constant definitions
35 !
36 !
37 FOR I=1 TO 5     ! print the constant definitions
38   PRINT "K";VAL$(I);" = ";VAL$(K(2*I-1));", ";VAL$(K(2*I))
39 NEXT I
40 !
41 ASSIGN @Dut;FORMAT ON
42 ENTER @Dut;Junk$    ! clean out the end of block stuff
43 END
```

LOADMATH

```
1 ! This short program illustrates how to use the CALC:MATH:DATA command to
2 ! download a complete math setup.
3 !
4 DIM Func$[1350] ! string to hold all 5 functions at their max length (270*5)
5 DIM F$(1:5)[80] ! array of example functions for F1 thru F5
6 DIM K(1:10)  ! array to hold values of 5 constants, K1 thru K4
7 !
8 F$(1)="LSPEC1/K2"
9 F$(2)="K1/FRES"
10 F$(3)="(FRES*K4)/K4-SQRT(FRES*FRES)+K1"
11 F$(4)="TIME1*(K1+K5)-XJOM(TIME1)-REAL(TIME1)"
12 F$(5)="PSPEC1-PSPEC2"
13 Func$=RPT$(CHR$(0),1350)   ! null out extire function array
14 Func$[1;LEN(F$(1))]=F$(1)   ! load each function in its proper position
15 Func$[271;LEN(F$(2))]=F$(2)
16 Func$[541;LEN(F$(3))]=F$(3)
17 Func$[811;LEN(F$(4))]=F$(4)
18 Func$[1081;LEN(F$(5))]=F$(5)
19 !
20 ! constants: K1 = 1+j0, K2 = 10+j0, K3 = 0+j1, K4 = -1+j0, K5 = 0-j1
21 DATA 1.0,0.0,10.0,0.0,0.0,1.0,-1.0,0.0,0.0,-1.0
22 READ K(*)
23 !
24 ! set math info
25 Dummy=FNSend_math_data(Func$,K(*))
26 !
27 END
28 !
29 DEF FNSend_math_data(Functions$,REAL Constant(*))
30   ASSIGN @Hp35665a TO 800
31   OUTPUT @Hp35665a;"FORM:DATA REAL,64"
32   OUTPUT @Hp35665a;"CALC:MATH:DATA ";       ! Space delimiter required.
33   OUTPUT @Hp35665a;"#41432";  ! 1432 byte block coming (2 + 5*270 + 5*2*8)
34   ASSIGN @Hp35665a;FORMAT OFF  ! send the integer with format off
35   OUTPUT @Hp35665a;1503;
36   ASSIGN @Hp35665a;FORMAT ON  ! send the string with format on
37   OUTPUT @Hp35665a;Functions$;
38   ASSIGN @Hp35665a;FORMAT OFF  ! send the REAL array with format off
39   OUTPUT @Hp35665a;Constant(*)
40   ASSIGN @Hp35665a;FORMAT ON
41   RETURN 1
42 FNEND
```

MANARM

```
10  ! HP Instrument BASIC example program: MANARM
20  ! -------------------------------------------------------------------
30   !
40  ! This program demonstrates using the instrument's
50  ! status registers to enable SRQs for event-
60  ! initiated program interrupts. In this case the
70  ! waiting_for_arm bit is detected.
80   !
90  ! -------------------------------------------------------------------
100  Sc=8
110  Addr=0
120  Device=(Sc*100)+Addr
130  ASSIGN @Hp35665a TO Device
140  CLEAR SCREEN
150  OUTPUT @Hp35665a;"SYST:PRES"
160  IF Sc=8 THEN
170    OUTPUT @Hp35665a;"DISP:FORM ULOW"
180    OUTPUT @Hp35665a;"DISP:PROG LOW"
190    CLEAR SCREEN
200  END IF
210 !
220 ! Setup registers to detect WAITING_FOR_ARM
230 !
240 ! clear any pending events
250  OUTPUT @Hp35665a;"*CLS"
260 ! allow SRQ from operation register
270  OUTPUT @Hp35665a;"*SRE 128"
280 ! allow SRQ from waiting_for_arm bit
290  OUTPUT @Hp35665a;"STAT:OPER:ENAB 64"
300 ! latch waiting_for_arm TRUE
310  OUTPUT @Hp35665a;"STAT:OPER:PTR 64"
320 ! do not latch waiting_for_arm FALSE
330  OUTPUT @Hp35665a;"STAT:OPER:NTR 0"
340 !
350 ! set up interrupts
360  ON INTR Sc GOSUB Check_srq
370  ENABLE INTR Sc;2
380 !
390 OUTPUT @Hp35665a;"FREQ:SPAN 100 Hz"
400 OUTPUT @Hp35665a;"ARM:SOUR MAN"
410 OUTPUT @Hp35665a;"ABOR;:INIT"
420  !
430  ! Wait for SRQ
440  !
450 Hang_out:GOTO Hang_out
460  !
470 Check_srq:    !
480  !
490    PRINT "SRQ Received"
500    Sb=SPOLL(Device)
```

```
510    PRINT "SPOLL(";Device;") = ";Sb
520    Queryarm(@Hp35665a)
530    ENABLE INTR Sc
540    RETURN
550  !
560  END
570  !**************************************************
580  ! Query Standard Event Status Register and arm
590  ! if waiting_for_arm event detected
600  !**************************************************
610  SUB Queryarm(@Device)
620   OUTPUT @Device;"STAT:OPER:EVEN?"
630   ENTER @Device;Resp
640   PRINT "STAT:OPER:EVEN?: ";Resp
650   IF Resp=64 THEN
660      INPUT "PRESS ENTER TO ARM (ENTER 'Q' TO QUIT)",A$
670      IF UPC$(A$)="Q" THEN STOP
680      OUTPUT @Device;"ARM"
690      PRINT "ARMED!"
700      PRINT
710   END IF
720   SUBEND
```

OPC_SYNC

```
10 ! HP Instrument BASIC program: OPCSYNC - Measurement synchronization
20 ! ------------------------------------------------------------------
30 ! This program demonstrates how to use the *OPC command to
40 ! allow an SRQ to interrupt program execution.  *OPC will set
50 ! the OPERATION_COMPLETE bit in the EVENT STATUS register
60 ! when all pending HP-IB commands have finished.  With the proper
70 ! register masks, this will generate a service request.
80 ! ------------------------------------------------------------------
90 !
100 Scode=8                            ! Interface select code
110 Address=0
120 Hp35665a=Scode*100+Address
130 !
140 OUTPUT Hp35665a;"FREQ:SPAN 50 HZ" !Measurement will take 8 seconds
150 OUTPUT Hp35665a;"*CLS"            !Clear the STATUS BYTE register
160 OUTPUT Hp35665a;"*ESE 1"          !Program the EVENT STATUS ENABLE reg.
170 OUTPUT Hp35665a;"*SRE 32"         !Program the STATUS BYTE ENABLE reg.
180 !
190 ON INTR Scode,2 GOTO Srq_handler  !Set up interrupt branching
200 ENABLE INTR Scode;2               !Allow SRQ to generate an interrupt
210 !
220 OUTPUT Hp35665a;"ABORT; INIT"     !Start the measurement
230 OUTPUT Hp35665a;"*OPC"            !Generate SRQ when all commands have
240                                   !finished.
250 Start_time=TIMEDATE
260 LOOP                              !Do something useful while waiting
270   DISP USING "14A, 2D.D";"Elapsed time :",TIMEDATE-Start_time
280   WAIT .1
290 END LOOP
300 !
310 Srq_handler:                       !Got an SRQ
320  Stb=SPOLL(Hp35665a)               !Read STATUS BYTE and clear SRQ
330  BEEP
340  OUTPUT Hp35665a;"*ESR?"           !Read and clear EVENT STATUS reg.
350  ENTER Hp35665a;Esr
360  DISP "Got the SRQ! SPOLL returns:";Stb;"   ESR returns:";Esr
370 END
```

OPCQ_SYNC

```
10 ! HP Instrument BASIC program: OPCQSYNC -  Measurement synchronization
20 ! ---------------------------------------------------------------------
30 ! This program demonstrates how to use the *OPC? HP-IB command
40 ! to hang the bus on a query before continuing on with the
50 ! program.  After all pending HP-IB commands have finished,
60 ! the HP 35665a will return a '1' in response to *OPC?.
70 ! ---------------------------------------------------------------------
80 !
90 Scode=8
100 Hp35665a=Scode*100
110 !
120 OUTPUT Hp35665a;"SYST:PRES"        !Preset the HP35665a
130 OUTPUT Hp35665a;"*OPC?"            !Pause on ENTER statement until
140 ENTER Hp35665a;Opc                 !'*RST' command has finished
150 !
160 OUTPUT Hp35665a;"FREQ:SPAN 50 Hz"  !Measurement will take 8 seconds
170 DISP "Measurement started ..."
180 OUTPUT Hp35665a;"ABOR; INIT"       !Start the measurement
190 OUTPUT Hp35665a;"*OPC?"            !Pause until all pending HP-IB commands
200 ENTER Hp35665a;Opc                 !have finished.
210 BEEP
220 DISP "Measurement done"
221 OUTPUT Hp35665a;"DISP:FORM ULOW"
223 OUTPUT Hp35665a;"INIT:CONT OFF"
230 END
```

TWO_CTLR

```
10  !HP BASIC program:  TWO_CTLR - Two controller operation
20  !------------------------------------------------------------
30  !This program demonstrates how an external controller
40  !and HP Instrument BASIC can work together.  This program
50  !will download a BASIC program to the HP 35665A and run it two
60  !times.  After each run, two BASIC program variables will
70  !will be read from the HP 35665A and displayed.
80  !------------------------------------------------------------
90  !
100   Scode=7                         !Select code for interface
110   Address=11                      !Address for HP 35665A
120   Hp35665a=Scode*100+Address
130 !
140   CLEAR Hp35665a
150   OUTPUT Hp35665a;"PROG:DEL:ALL"   !Scratch the program space
160 !
170   DISP "Downloading the program..."
180   ASSIGN @Prog TO Hp35665a;EOL CHR$(10) !Change EOL character
190   OUTPUT @Prog;"PROG:DEF #0";          !Send program
200   OUTPUT @Prog;"10 COM INTEGER Times_run,Test$[10]"
210   OUTPUT @Prog;"20 Times_run=Times_run +1"
220   OUTPUT @Prog;"30 IF Times_run=1 THEN Test$=""PASS"""
230   OUTPUT @Prog;"40 IF Times_run=2 THEN Test$=""FAIL"""
240   OUTPUT @Prog;"50 BEEP"
250   OUTPUT @Prog;"60 END"
260   OUTPUT @Prog;CHR$(10) END        !Terminate the data block
270 !
280 !Set up registers for interrupt on PROGRAM_RUNNING going false
290   OUTPUT Hp35665a;"*CLS"           !Clear the STATUS register
291 !Program NTR reg and OPERATION ENABLE reg for PROGRAM_RUNNING bit
300   OUTPUT Hp35665a;"STAT:OPER:NTR 16384"
310   OUTPUT Hp35665a;"STAT:OPER:ENAB 16384"
320   OUTPUT Hp35665a;"*SRE 128"    !Allow SRQ on bit 7 of STATUS reg
330 !
340   DISP "Running the program..."
350   OUTPUT Hp35665a;"PROG:STAT RUN"  !Run Program
360   Display_res(Hp35665a,Scode)      !Read and display variables
370   OUTPUT Hp35665a;"PROG:STAT RUN"  !Run Program  again
380   Display_res(Hp35665a,Scode)      !Read and display variables
390 !
400   END                              !End of this program
410 !
420   SUB Display_res(Hp35665a,Scode)
430 ! This subprogram waits for an SRQ interrupt to signal that a
440 ! BASIC program has finished.  It then clears the HP-IB registers
450 ! by reading them.  Once that is done, the values of two IBASIC
460 ! variables are read and displayed.
470 !
480     ON INTR Scode GOTO Read_results     !Set up interrupt branching
490     ENABLE INTR Scode;2                 !Allow interrupt on SRQ
```

```
500 Idle:GOTO Idle
510    !
520 Read_results:                           !Program has finished
530     A=SPOLL(Hp35665a)                   !Read and clear the SRQ
540     OUTPUT Hp35665a;"STAT:OPER?" !Read and clear OPERATION STATUS reg.
550     ENTER Hp35665a;Event
560     WAIT .5
570     !
580     OUTPUT Hp35665a;"FORM:DATA ASCII,3"
590     OUTPUT Hp35665a;"PROG:NUMB? 'Times_run'"!Read the first variable
600     ENTER Hp35665a;Times_run
610     !
620     OUTPUT Hp35665a;"PROG:STR? 'Test$'"     !Read the second variable
630     ENTER Hp35665a;Test$
640     !
650     PRINT "Times_run: ";Times_run,"Test$: ";Test$
660   SUBEND
```

UPLOAD

```
10 ! HP BASIC example program : UPLOAD
20 ! --------------------------------------------------------------
30 ! This program runs on an HP BASIC workstation connected to
40 ! the HP 35665A with HP Instrument BASIC installed.  The 35665A
50 ! must have its address set to 711 and must be set up as
60 ! ADDRESSABLE ONLY on the HP-IB. This program uploads the
70 ! current program in the HP 35665A's memory to an ASCII file
80 ! on the workstation's current MSI disk.
90 ! --------------------------------------------------------------
100   ASSIGN @Hp35665a TO 711
110   DIM Prog_line$[256]
120   INPUT "ENTER NAME OF FILE INTO WHICH TO UPLOAD PROGRAM ",Filename$
130   PRINT Filename$
140   CLEAR @Hp35665a
150   OUTPUT @Hp35665a;"PROG:DEF?"
160   ENTER @Hp35665a USING "#,A,D";Prog_line$,Ndigits
170   ENTER @Hp35665a USING "#,"&VAL$(Ndigits)&"D";Nbytes
180   PRINT Nbytes
190   Openfile(@File,Filename$,Nbytes)
200   ASSIGN @File TO Filename$
210   LOOP
220     ENTER @Hp35665a;Prog_line$
230   EXIT IF LEN(Prog_line$)=0
240     PRINT Prog_line$
250     OUTPUT @File;Prog_line$
260   END LOOP
270   ASSIGN @File TO *
290   END
300   SUB Openfile(@File,Filename$,Fisize)
310     ON ERROR GOTO Openerr
320     IF Fisize MOD 256 > 0 THEN Fisize=Fisize+256
330     CREATE ASCII Filename$,Fisize DIV 256
340 Openerr:  !
350     IF ERRN < >54 THEN
360       PRINT ERRM$
370     END IF
380   SUBEND
```

WAI_SYNC

```
10  ! HP Instrument BASIC program: WAI_SYNC - Measurement synchronization
11  ! ----------------------------------------------------------------------
12  ! This program demonstrates how to use the *WAI command to
13  ! prevent execution of an HP-IB command until all previous
14  ! commands have finished.  In this example, the trace display
15  ! measurement has finished.
16  ! The *WAI command does not affect program operation. The
17  ! program will run to completion, sending all of the commands to
18  ! to the HP35665A without waiting for them to be executed.
19  ! ----------------------------------------------------------------------
100 Scode=8                           !Interface select code
110 Address=0
120 Hp35665a=Scode*100+Address
160 !
170 DISP "Sending HP-IB commands..."
171 OUTPUT Hp35665a;"SYST:PRES"
172 OUTPUT Hp35665a;"AVER:COUN 1"
174 OUTPUT Hp35665a;"AVER ON"
190 OUTPUT Hp35665a;"FREQ:SPAN 50 HZ"!Set narrow span
200 OUTPUT Hp35665a;"ABORT; INIT"    !Start the measurement
210 OUTPUT Hp35665a;"*WAI"           !Tell analyzer to wait here until
220                                  !all HP-IB commands have finished
230 OUTPUT Hp35665a;"DISP:FORM ULOW" !Go to upper/lower after waiting
240 BEEP
250 DISP "Finished. Display will go to UPPER/LOWER when meas. done"
260 END
```

7

Introduction to the Command Reference

The Command Reference chapters describe all of the HP 35665A's HP-IB commands. Each command description has the following:

- A brief description of the command. This one- or two-line description appears just below the heading.
- A syntax description. This consists of one or two fields, depending on whether the command has only a command form, only a query form, or both. It shows you the syntax expected by the analyzer's HP-IB command parser. At the end of this chapter is a detailed description for the elements appearing in the syntax description.
- Example statements. This field appears at the end of syntax description. It contains two HP BASIC output statements that use the command.
- A returned format description. This field is only used if the command has a query form. It tells you how data is returned in response to the query.
- An attribute summary. This field indicates whether one of the analyzer's options must be installed; defines the command's preset state and specifies compliance with SCPI. A "confirmed" command complies with SCPI 1990. An "approved" command complies with SCPI 1991. An "instrument-specific" command does not conform to the SCPI standard. This field also identifies overlapped commands. (See "Synchronization" in chapter 2 for more information.)
- A detailed description. This field contains additional information about the command.

Finding the Right Command

- If you can not find a command you have seen in a program, remember that commands can omit implied mnemonics.

  For example, the command SENSe:FREQuency:CENTer 10000 HZ contains the implied mnemonic SENSe. SENSe can be omitted to create the equivalent command FREQuency:CENTer 10000 HZ. (See "Implied Mnemonics" in chapter 2.) You will not find an entry for FREQuency:CENTer—or any other command that omits an implied mnemonic—in the command reference. You will find the FREQuency:CENTer command in the [SENSe:] command reference chapter.

- If you do not find a command where you expect it, try scanning the command list in appendix A for the equivalent command that contains the implied mnemonic.

  Each command has a brief description. After you locate the equivalent command, you can find a more detailed description in one of the command reference chapters.

- If you are looking for a command that accesses a particular function, use the index.

  For example, if you want to find the command that changes the analyzer's center frequency, look for "center frequency" in the index. It sends you to the page that describes the SENSe:FREQuency:CENTer command.

- If you are familiar with front panel operation of the analyzer, use Appendix B.

  It provides a cross reference of the analyzer's hardkeys and softkeys and their equivalent HP-IB commands.

- If you have an analyzer, use the GPIB Echo facility.

  It displays the corresponding HP-IB command for front panel operations. GPIB Echo displays the most abbreviated form of the command. To turn on the facility, press
  [Local/HP-IB] [GPIB ECHO ON OFF]

Figure 7-1 shows the flow of measurement and signal data through the HP 35665A. The SCPI subsystems appear in the block diagram. Use figure 7-1 to help identify the subsystem containing a command. For example, the location of the SENSe subsystem indicates it contains commands which determine how measurement data is acquired. The use of the brackets ([ ]) indicates SENSe is an implied mnemonic.

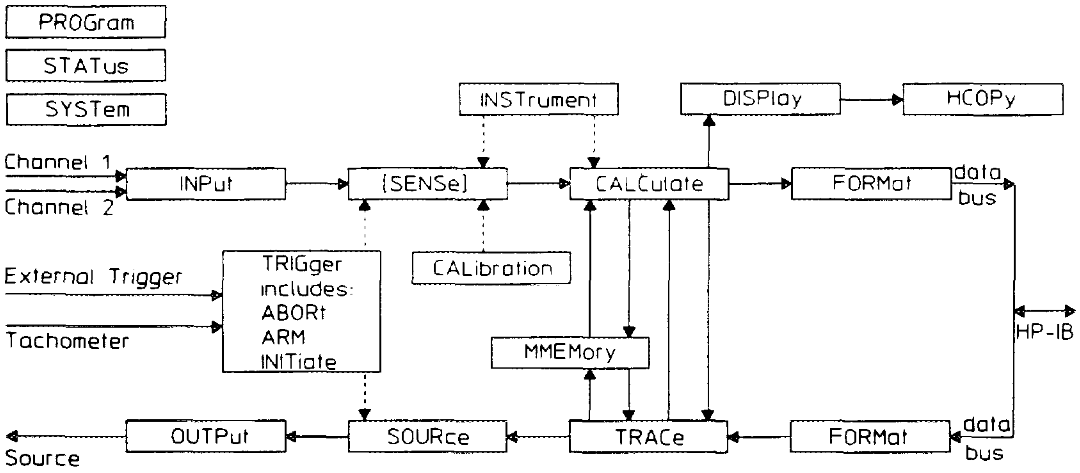


Figure 7-1. HP 35665A Measurement and Signal Flow

Conventions

Syntax and returned format descriptions use the following conventions:

- < > Angle brackets enclose the names of items that need further definition. The definition will be included in accompanying text. In addition, detailed descriptions of these elements appear at the end of this chapter.

- ::= "is defined as" When two items are separated by this symbol, the second item can replace the first in any statement that contains the first item. For example, A::=B indicates that B can replace A in any statement that contains A.

- | "or" When items in a list are separated by this symbol, one and only one of the items can be chosen from the list. For example, A|B indicates that A or B can be chosen, but not both.

- ... An ellipsis (trailing dots) is used to indicate that the preceding element may be repeated one or more times.

- [ ] Square brackets indicate that the enclosed items are optional.

- { } Braces are used to group items into a single syntactic element. They are most often used to enclose lists and to enclose elements that are followed by an ellipsis. One of the items must be sent with the command.

Although the analyzer is not case sensitive, the case of letters in the command mnemonics is significant in the command reference chapters. Mnemonics that are longer than four characters can have a short form or a long form. The analyzer accepts either form. Upper-case letters show the short form of a command mnemonic. For more information, see "Command Abbreviation" in chapter 3.

The analyzer is sensitive to white space characters. White space characters are not allowed within command mnemonics. They are only allowed when they are used to separate a command and a parameter.

Syntax Descriptions

Syntax descriptions in the command reference chapters use the following elements:

< BLOCK >

This item designates block data. There are three kinds of block data; binary definite length block data, binary indefinite length block data, and ASCII data. The analyzer always returns binary definite length block data or ASCII data in response to queries.

When you send block data to the analyzer, you can send it either as definite length block or indefinite length block data or as ASCII data.

If you are sending binary data using indefinite length syntax, < BLOCK > takes the following form:
< BLOCK > ::= #0 < data_byte > [, < data_byte >] . . . < LF > < ^END >
< data_byte > ::= unsigned 8-bit data
< LF > ::= line feed character, ASCII decimal 10
< ^END > ::= HP-IB END message (EOI set true)

If you are sending binary data using definite length syntax, < BLOCK > takes the following form:
< BLOCK > ::= # < byte > < length_bytes > < data_byte > [< data_byte >] . . .
< byte > ::= number of length bytes to follow (ASCII encoded)
< length_bytes > ::= number of data bytes to follow (ASCII encoded)
< data_byte > ::= unsigned 8-bit data

If you are sending ASCII data, < BLOCK > takes the following form:
< BLOCK > ::= NRf[,NRf]. . . < LF >

See chapter 4, “Transferring Data,” for more information about block data.

< CHAR >

This item designates a string of ASCII characters. There are no delimiters. Usually, the string is from an explicit set of responses. Maximum length is 12 characters.

< CMDSTR >

This string specifies the type of measurement data, where:
D[1|2| .. |8] selects a data register
TCAPture [1|2] selects a time capture buffer
W[1|2|...|8] selects a waterfall register
XFRequency:POWer [1|2] selects power spectrum
XFRequency:POWer:COHerence 1,2 selects coherence
XFRequency:POWer:COMPosite [1|2] selects composite power
XFRequency:POWer:CROSs 1,2 selects cross spectrum
XFRequency:POWer:LINear [1|2] selects linear spectrum
XFRequency:POWer:RATio 2,1 selects frequency response
XFRequency:POWer:VARiance [1|2] selects normalized variance
XORDer:TRACk [1|2][,1|2|3|4|5] selects order track
XRPM:PROFile selects RPM profile
XTIMe:CORRelation [1|2] selects auto correlation
XTIMe:CORRelation:CROSs 1,2 selects cross correlation
XTIMe:VOLTage [1|2] selects time
XTIMe:VOLTage:CDF [1|2] selects cumulative density function
XTIMe:VOLTage:HISTogram [1|2] selects histogram
XTIMe:VOLTage:PDF [1|2] selects probability density function
XTIMe:VOLTage:WINDow [1|2] selects windowed time
XVOLtage:VOLTage 1,2 selects orbit

<DEF_BLOCK>

This item designates definite block data which takes the following form if the data is binary encoded (FORMat:DATA REAL command):

```
<DEF_BLOCK> ::= #<byte><length_bytes><data_byte>[<data_byte>]...
<byte> ::= number of length bytes to follow (ASCII encoded)
<length_bytes> ::= number of data bytes to follow (ASCII encoded)
<data_byte> ::= unsigned 8-bit data
```

If the data is ASCII encoded (FORMat:DATA ASCii command):

```
<DEF_BLOCK> ::= NRf[,NRf]... <LF>
```

See chapter 4, “Transferring Data,” for more information about block data.

<EXPR>

This item designates an expression for a math function.

```
<EXPR> ::= ([<expr_element>]...)
<expr_element> ::= See the operations and operands listed below:
```

Operations:

AWEIGHT	Apply A-weight filter
BWEIGHT	Apply B-weight filter
CWEIGHT	Apply C-weight filter
CONJ	Complex Conjugate
DIFF	Differentiate
DJOM	Divide by j θ
EXP	Exponential
FFT	Fast Fourier Transform
IFFT	Inverse Fast Fourier Transform
INTEG	Integrate
IMAG	Imaginary Part
LN	Natural Logarithm
MAG	Magnitude
PSD	Power Spectral Density
REAL	Real Part
SQRT	Square Root
XJOM	multiply by j θ
+	Add
-	Subtract
*	Multiply
/	Divide

Operands:
D1 | D2 | . . . D8 Contents of data registers
F1 | F2 | . . . F5 Contents of function registers
K1 | K2 | . . . K5 Contents of constant registers

Measurement Data (depends on instrument mode)

[1 | 2] specifies which channel the data was taken from.
PSPEC[1 | 2] Power Spectrum
LSPEC[1 | 2] Linear Spectrum
TIME[1 | 2] Time Data
WTIME[1 | 2] Windowed Time Data
FRES Frequency Response
CSPEC Cross Spectrum
HIST[1 | 2] Histogram (INST:SEL HIST only)
PDF[1 | 2] Probability Density Function (INST:SEL HIST only)
CDF[1 | 2] Cumulative Density Function (INST:SEL HIST only)
TIME[1 | 2] Unfiltered Time (INST:SEL HIST only)
ACORR[1 | 2] Autocorrelation (INST:SEL CORR only)
XCORR[1 | 2] Cross Correlation (INST:SEL CORR only)
CPOW[1 | 2] Composite Power (INST:SEL ORD only; Option 1D0)
TRACK[1 | 2 | 3 | 4 | 5] (INST:SEL ORD only; Option 1D0)
[1 | 2 | 3 | 4 | 5] specifies order track
TIME[1 | 2] Resampled Time (INST:SEL ORD only; Option 1D0)
RPM RPM Profile (INST:SEL ORD only; Option 1D0)
NVAR[1 | 2] Normalized Variance (INST:SEL SINE only; Option **1D2)**

<FILE>

This string is used to describe the name of a file. It does not include any disk drive information.

The valid character set for <FILE> depends on the disk format.

DOS file names are limited to 8 ASCII characters followed by a period and a three character extension. The period and extension are not required. File names are *not* case sensitive.

LIF file names are limited to 10 character which may include all characters except ":" "<" and "|". The first character must be a letter. File names *are* case sensitive.

< FILENAME >

This string is used to describe the name of a file on the default disk or on the specified disk. The allowed form is:

"[MSIS:]filename"

where MSIS: must be replaced with:

RAM: which selects volatile RAM.
NVRAM: which selects non-volatile RAM.
INT: which selects the internal disk drive.
EXT[, < select_code > [, < unit_number >]]: which selects the external disk drive.

The brackets are not literal in the parameter. They indicate that the disk drive designation is optional. If the disk drive is not specified, the default disk drive is used. Select the default specifier with the MMEM:MSIS command.

The valid character set for < FILENAME > depends on the disk format.

DOS file names are limited to 8 ASCII characters followed by a period and a three character extension. The period and extension are not required. File names are *not* case sensitive.

LIF file names are limited to 10 character which may include all characters except ":" "<" and "|". The first character must be a letter. File names *are* case sensitive.

< MSINAME >

This item specifies the mass storage device. It is a < STRING > with one of the following values:

"RAM:"
"NVRAM:"
"INT:"
"EXT[, < select_code > [, < unit_number >]]:"

< MMEMNAME >

This item specifies a single file or an entire mass storage device. It takes the same form as < FILENAME > except either the name of the file, the name of the mass storage device or both must be present. It can take one of the following forms:

"[MSIS:]filename"
"MSIS:[filename]"
"MSIS:filename"

See the description for < FILENAME > for the valid mass storage device specifiers and the valid file name character set.

< PROGRAM >

This item designates an HP Instrument BASIC program.

Load and save HP Instrument BASIC using the indefinite length block data format. The simplest way to load an HP Instrument BASIC program into the analyzer is to send the command (PROG[:SELected]:DEFine) followed by #0, followed by all the characters making up the program (including line numbers and linefeeds at the end of each program statement).

Terminate the entire command with line feed character (ASCII decimal 10) and < ^END > (the HP-IB END message (EOI set true) while the last byte of data is on the bus).

< STRING >

This item specifies any 8-bit characters delimited by single quotes or double quotes. The beginning and ending delimiter must be the same. If the delimiter character is in the string, it must be entered twice.

8

Common Commands

This section describes all of the IEEE 488.2 common commands implemented by the HP 35665A. An important property of all common commands is that you can send them without regard to a program message's position in the HP-IB command tree.

For more information on the HP-IB command tree, see chapter 3, "Programming with HP-IB Commands."

*CAL? query

Calibrates the analyzer and returns the result.

Query Syntax: `*CAL?`

Example Statements:

```
OUTPUT 711;"*CAL?"
OUTPUT 711;"*cal?"
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The analyzer performs a full calibration when you send this query. If the calibration completes without error, the analyzer returns 0. If the calibration fails, the analyzer returns 1.

This query is the same as the CAL:ALL? query.

*CLS command

Clears the Status Byte by emptying the error queue and clearing all event registers.

Command Syntax:

```
*CLS
```

Example Statements:

```
OUTPUT 711;"*Cls"
OUTPUT 711;"*CLS"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command clears the Status Byte register. It does so by emptying the error queue and clearing (setting to 0) all bits in the event registers of the following register sets:

- User Status.
- Device State.
- Questionable Voltage.
- Limit Fail.
- Questionable Status.
- Standard Event.
- Operation Status.

In addition, *CLS cancels any preceding *OPC command or query. This ensures that bit 0 of the Standard Event register will not be set to 1 and that a response will not be placed in the analyzer's output queue when pending overlapped commands are completed.

*CLS does not change the current state of enable registers or transition filters.

Note To guarantee that the Status Byte's Message Available and Master Summary Status bits are cleared, send *CLS immediately following a Program Message Terminator.

See chapter 5 for more information on the Status Byte register.

*ESE command/query

Sets bits in the Standard Event enable register.

Command Syntax:

```
*ESE <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:255
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"*ese 101"
OUTPUT 711;"*Ese 214"
```

Query Syntax: `*ESE?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  dependent on setting of *PSC
SCPI Compliance:  confirmed
```

Description:

This command allows you to set bits in the Standard Event enable register. Assign a decimal weight to each bit you want set (to 1) according to the following formula:

$$2^{(\text{bit\_number})}$$

with acceptable values for bit_number being 0 through 7. Add the weights and then send the sum with this command.

When an enable register bit is set to 1, the corresponding bit of the Standard Event event register is enabled. All enabled bits are logically ORed to create the Standard Event summary, which reports to bit 5 of the Status Byte. Bit 5 is only set to 1 if both of the following are true:

- One or more bits in the Standard Event event register are set to 1.
- At least one set bit is enabled by a corresponding bit in the Standard Event enable register.

The setting last specified with *ESE is saved in nonvolatile memory. It can be recalled at power-up, depending on the setting of the Power-on Status Clear flag (set with *PSC). When the flag is 0 at power-up, all bits in the Standard Event enable register are set according to the saved *ESE value. When the flag is 1 at power-up, all bits in the Standard Event enable register are initialized to 0.

The query returns the current state of the Standard Event enable register. The state is returned as a sum of the decimal weights of all set bits.

For more information on the Standard Event register set, see chapter 5.

*ESR? query

Reads and clears the Standard Event event register.

Query Syntax: `*ESR?`

Example Statements:

```
OUTPUT 711;"*ESR?"
OUTPUT 711;"*esr?"
```

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

This query returns the current state of the Standard Event event register. The state is returned as a sum of the decimal weights of all set bits. The decimal weight for each bit is assigned according to the following formula:

$$2^{(\text{bit\_number})}$$

with acceptable values for bit_number being 0 through 7.

The query clears the register after it reads the register.

A bit in this register is set to 1 when the condition it monitors becomes true. A set bit remains set, regardless of further changes in the condition it monitors, until one of the following occurs:

- You read the register with this query.
- You clear all event registers with the *CLS command.

For more information on the Standard Event register set, see chapter 5.

*IDN? query

Returns a string that uniquely identifies the analyzer.

Query Syntax: `*IDN?`

Example Statements:

```
OUTPUT 711;"*Idn?"
OUTPUT 711;"*IDN?"
```

Return Format: `"HEWLETT-PACKARD,35665A,<serial_number><software_version>"`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  instrument dependent
SCPI Compliance:  confirmed
```

Description:

The response to this query uniquely identifies your analyzer.

The query returns:

- The name of the manufacturer, Hewlett-Packard.
- The product number.
- The serial number of the analyzer.
- The version of the software.

*OPC command/query

Sets or queries completion of all pending overlapped commands.

Command Syntax: `*OPC`

Example Statements:

```
OUTPUT 711;"*opc"
OUTPUT 711;"*Opc"
```

Query Syntax: `*OPC?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

Some commands are processed sequentially by the analyzer. A sequential command holds off the processing of subsequent commands until it has been completely processed. However, some commands do not hold off the processing of subsequent commands. These commands are called overlapped commands. The Attribute Summary for each command indicates whether or not it is an overlapped command.

The analyzer uses the No Pending Operation (NPO) flag to keep track of overlapped commands that are still pending (that is, not completed). The NPO flag is reset to 0 when an overlapped command is pending. It is set to 1 when no overlapped commands are pending. You cannot read the NPO flag directly, but you can use *OPC and *OPC? to tell when the flag is set to 1.

If you use *OPC, bit 0 of the Event Status event register is set to 1 when the NPO flag is set to 1. This allows the analyzer to generate a service request when all pending overlapped commands are completed (assuming you have enabled bit 0 of the Event Status register and bit 5 of the Status Byte register).

If you use *OPC?, +1 is placed in the output queue when the NPO flag is set to 1. This allows you to effectively pause the controller until all pending overlapped commands are completed. It must wait until the response is placed in the queue before it can continue.

Note The *CLS and *RST commands cancel any preceding *OPC command or query. Pending overlapped commands are still completed, but you can no longer determine when they are completed.

Two HP-IB bus management commands—Device Clear (DCL) and Selected Device Clear (SDC)—also cancel any preceding *OPC command or query.

See "Overlapped Commands" in chapter 2 for additional information about the *OPC command and the *OPC? query.

*OPT? query

Returns a string that identifies the analyzer's option configuration.

Query Syntax:

```
*OPT?
```

Example Statements:

```
OUTPUT 711;"*OPT?"
OUTPUT 711;"*opt?"
```

Return Format:

```
"1D0,1D1,1D2,1D3,1C2,ANA"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  instrument dependent
SCPI Compliance:  confirmed
```

Description:

The response to this query identifies the analyzer's option configuration. For example, if your analyzer has Computed Order Tracking installed, it returns 1D0 to this query. Options are identified by the following:

- 1D0 Computed Order Tracking
- 1D1 Real Time Octave Measurements
- 1D2 Swept Sine Measurements
- 1D3 Curve Fit / Synthesis
- 1D4 Arbitrary Source
- 1C2 HP Instrument BASIC
- 1C1 Add 2MBytes Memory
- ANA Add 6MBytes Memory

The query returns a null string ("") if special options are not installed in the analyzer.

*PCB command

Sets the pass-control-back address.

Command Syntax:

```
*PCB <primary_address>, [<secondary_address>]

  <primary_address> ::= <number>|<step>|<bound>
<secondary_address> ::= <number>|<step>|<bound>
           <number> ::= a real number (NRf data)
                        limits:  0:30
            <bound> ::= MAX|MIN
           <number> ::= a real number (NRf data)
                        limits:  0:30
            <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"*Pcb 11"
OUTPUT 711;"*PCB 19"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Use this command to specify the address of your controller before you pass control of the HP-IB to the analyzer. When the analyzer completes the operation that required it to have control of the bus, it automatically passes control back to the controller at the specified address.

The optional second number is only used for controllers that support extended addressing. It is ignored by the analyzer.

The address last specified with this command is saved in nonvolatile memory, so it is not affected when you turn the analyzer off and on. It is also not affected by the *RST command.

*PSC command/query

Sets the state of the Power-on Status Clear flag.

Command Syntax:

```
*PSC <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  -32767:32767

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"*psc 0"
OUTPUT 711;"*Psc 1"
```

Query Syntax: `*PSC?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This command lets you specify whether or not the Service Request enable register and the Event Status enable register should be cleared (all bits reset to 0) at power-up.

The settings of the Service Request enable register and the Event Status enable register are saved in nonvolatile memory when you turn the analyzer off. These settings can be recalled when you turn the analyzer on, but only if the Power-on Status Clear (PSC) flag is reset to 0. When the PSC flag is set to 1, the two enable registers are cleared at power-up. Use *PSC to specify the state of the PSC flag.

The number last specified with *PSC is saved in nonvolatile memory, so it is not affected when you turn the analyzer off and on. It is also not affected by the *RST command.

If you want the analyzer to generate a service request at power-up, bit 7 of the Event Status enable register and bit 5 of the Service Request enable register must be set. This is only possible if the PSC flag is reset to 0.

The query returns the current state of the PSC flag.

*RST command

Executes a device reset.

Command Syntax:

```
*RST
```

Example Statements:

```
OUTPUT 711;"*RST"
OUTPUT 711;"*rst"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command returns the analyzer to a reset state. In addition, *RST cancels any pending *OPC command or query.

The reset state is similiar to the preset state. The preset state of each command is listed in the Attribute Summary. In some cases, however, a command's reset state differs from its preset state. These commands (and their reset states) are listed below.

- CALibration:AUTO is set to 0 (OFF).
- FREQuency:SPAN:LINK is set to CENTer.
- FREQuency:RESolution:AUTO is ON.

Note This command is *not* equivalent to a front panel preset. Send SYST:PRES if you want to send a command that is equivalent to a front panel preset.

The following are *not* affected by this command:

- The state of the Power-on Status Clear flag.
- The state of all enable and transition registers.
- The HP-IB input and output queues.
- The time and date (SYST:TIME and SYST:DATE).
- The HP-IB address settings (SYST:COMM:GPIB:ADDR, HCOP:PLOT:ADDR, HCOP:PRIN:ADDR).
- The HP-IB controller capability setting.
- The default disk selection (MMEM:MSIS).
- Contents of limit and data registers.
- Contents of math function and constant registers.
- Contents of the RAM disks.
- Calibration constants.
- Contents of the time capture buffer.

*SRE — command/query

Sets bits in the Service Request enable register.

Command Syntax:

```
*SRE <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:255
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"*Sre 40"
OUTPUT 711;"*SRE 97"
```

Query Syntax: `*SRE?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  dependent on setting of *PSC
SCPI Compliance:  confirmed
```

Description:

This command allows you to set bits in the Service Request enable register. Assign a decimal weight to each bit you want set (to 1) according to the following formula:

$$2^{(bit\_number)}$$

with acceptable values for bit_number being 0 through 7. Add the weights and then send the sum with this command.

Note

The analyzer ignores the setting you specify for bit 6 of the Service Request enable register. This is because the corresponding bit of the Status Byte register is always enabled.

The analyzer requests service from the active controller when one of the following occurs:

- A bit in the Status Byte register changes from 0 to 1 while the corresponding bit of the Service Request enable register is set to 1.
- A bit in the Service Request enable register changes from 0 to 1 while the corresponding bit of the Status Byte register is set to 1.

The setting last specified with *SRE is saved in nonvolatile memory. It can be recalled at power-up, depending on the setting of the Power-on Status Clear flag (set with *PSC). When the flag is 0 at power-up, all bits in the Service Request enable register are set according to the saved *SRE value. When the flag is 1 at power-up, all bits in the Service Request enable register are initialized to 0.

The query returns the current state of the Service Request enable register. The state is returned as a sum of the decimal weights of all set bits.

*STB? query

Reads the Status Byte register.

Query Syntax:	`*STB?`

Example Statements:	`OUTPUT 711;"*stb?"`
	`OUTPUT 711;"*Stb?"`

Return Format:	`NR1`

Attribute Summary:	`Option: not applicable`
	`Overlapped: no`
	`Preset State: variable`
	`SCPI Compliance: confirmed`

Description:

This command allows you to set bits in the Status Byte register. The state is returned as a sum of the decimal weights of all set bits. The decimal weight for each bit is assigned according to the following formula:

$$2^{(\text{bit\_number})}$$

with acceptable values for bit_number being 0 through 7.

The register is not cleared by this query. To clear the Status Byte register, you must send the *CLS command.

Bits in the Status Byte register are defined as follows:

- Bit 0 summarizes all enabled bits of the User Status register.
- Bit 1 is reserved.
- Bit 2 summarizes all enabled bits of the Device State register.
- Bit 3 summarizes all enabled bits of the Questionable Status register.
- Bit 4 is the Message Available (MAV) bit. It is set whenever there is something in the analyzer's output queue.
- Bit 5 summarizes all enabled bits of the Standard Event Status register.
- Bit 6, when read with this query (*STB?), acts as the Master Summary Status (MSS) bit. It summarizes all enabled bits of the Status Byte register. (Bit 6 acts as the Request Service (RQS) bit when it is read by a serial poll.
- Bit 7 summarizes all enabled bits of the Operation Status register.

For more information on the Status Byte register, see chapter 5.

*TRG command

Triggers the analyzer if TRIG:SOUR is BUS.

Command Syntax:

```
*TRG
```

Example Statements:

```
OUTPUT 711;"*TRG"
OUTPUT 711;"*trg"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command triggers the analyzer if the following two conditions are met:

- The HP-IB is designated as the trigger source. (Send the TRIG:SOUR:BUS command.)
- The analyzer is waiting to trigger. (Bit 5 of the Operational Status condition register must be set.) It is ignored at all other times.

The *TRG command has the same effect as TRIG:IMM. It also has the same effect as the HP-IB bus management command Group Execute Trigger (GET).

This command triggers the analyzer when the following two conditions are met:

- The HP-IB is designated as the trigger source. (See the TRIG:SOUR BUS command.)
- The analyzer is waiting to trigger. (Bit 5 of the Operational Status register must be set).

The *TRG command has the same effect as TRIG:IMM. It also has the same effect as the HP-IB bus management command Group Execute Trigger (GET).

*TST? query

Tests the analyzer hardware and returns the results.

Query Syntax: *TST?

Example Statements:

```
OUTPUT 711;"*Tst?"
OUTPUT 711;"*TST?"
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The analyzer's self-test performs a full calibration and then compares the calibration results to specified limits. If the results are within specified limits, the analyzer returns 0. If the results exceed the specified limits, the analyzer returns 1.

*WAI command

Holds off processing of subsequent commands until all preceding commands have been processed.

Command Syntax:

```
*WAI
```

Example Statements:

```
OUTPUT 711;"*wai"
OUTPUT 711;"*Wai"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

Use *WAI to hold off the processing of subsequent commands until all pending overlapped commands have been completed.

Some commands are processed sequentially by the analyzer. A sequential command holds off the processing of any subsequent commands until it has been completely processed. However, some commands do not hold off the processing of subsequent commands; they are referred to as overlapped commands. *WAI ensures that overlapped commands are completely processed before subsequent commands (those sent after *WAI) are processed.

See "Synchronization" in chapter 2 for additional information about the use of *WAI and overlapped commands.

9

ABORt

This subsystem contains one command which aborts any measurement process.

ABORt command

Stops the current measurement in progress.

Command Syntax:

```
ABORt
```

Example Statements:

```
OUTPUT 711;"ABORT"
OUTPUT 711;"abor"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command aborts any measurement in progress and resets the trigger system. ABOR forces the trigger system to an idle state. ABOR forces the Measuring bit (bit 4) and the Averaging bit (bit 8) of the Operational Status register to 0. To restart a measurement you must send the INIT:IMM command.

The program message ABOR;INIT:IMM serves a special synchronization function. When you send this message to restart a measurement, the analyzer's No Pending Operation (NPO) flag is set to 1 until the measurement is complete. The two commands and the query that test the state of this flag—*WAI, *OPC, and *OPC?—allow you to hold off subsequent actions until the measurement is complete. See "Synchronization" in chapter 2 for more information on the NPO flag.

Note

When averaging is on ([SENSe:]AVERage[:STATe] ON) the NPO flag is not set to 1 until *n* measurements have been combined into one trace. You specify the value of *n* with the[SENSe:] AVER:COUNt command.

ARM

This subsystem contains commands that control the analyzer's trigger arming functions. See the TRIGger subsystem for commands related to other triggering functions.

Figure 10-1 shows the model for the HP 35665A's ARM-INITiate-TRIGger functions.

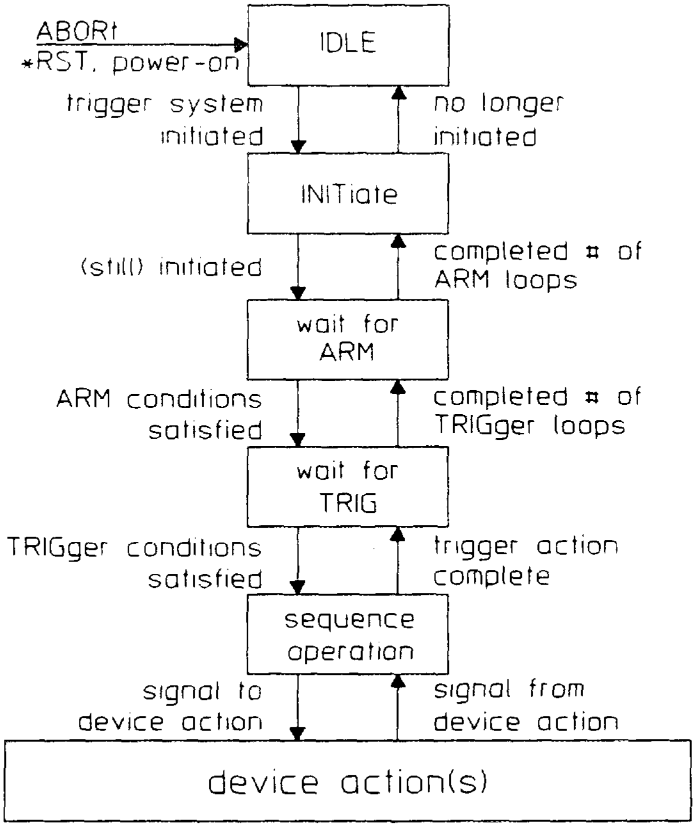


Figure 10-1. The ARM-INITiate-TRIGger Functions

ARM[:IMMediate] command

Arms the trigger if ARM:SOUR is MAN.

Command Syntax: `ARM[:IMMediate]`

Example Statements:

```
OUTPUT 711;":Arm"
OUTPUT 711;"ARM:IMM"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

Two conditions must be met before this command arms the trigger.

- Manual arming must be selected (ARM:SOUR is MAN).
- Bit 1 (Settling) or bit 6 (RDY for ARM) of the Operational Status condition register must be set to 1.

ARM:IMM is ignored at all other times.

ARM:RPM:INCRement command/query

Specifies the number of RPM in a step for RPM step arming.

Command Syntax: `ARM:RPM:INCRement {<number>[<unit>]}|<step>|<bound>`

```
<number> ::= a real number (NRf data)
             limits:  1:500000
  <unit> ::= [RPM]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"arm:rpm:incr 10"
OUTPUT 711;"Arm:Rpm:Increment 500"
```

Query Syntax: `ARM:RPM:INCRement?`

Return Format: `NR3`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +6.0E+001
SCPI Compliance:  instrument-specific
```

Description:

This command determines the size of each step used when RPM step arming is enabled with the ARM:SOUR RPM command.

The first arm occurs when the RPM value reaches the threshold value (specified with the ARM:RPM:THR command). Subsequent arms occur at the RPM interval specified with this command.

ARM:RPM:MODE

command/query

Enables the Start RPM Arming qualifier.

Command Syntax: `ARM:RPM:MODE {OFF|0|UP|DOWN}`

Example Statements:

```
OUTPUT 711;":ARM:RPM:MODE DOWN"
OUTPUT 711;"arm:rpm:mode OFF"
```

Query Syntax: `ARM:RPM:MODE?`

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  UP
SCPI Compliance:  instrument-specific
```

Description:

For a runup measurement send ARM:RPM:MODE UP. The first arm occurs when the RPM value reaches the specified threshold value (specified with ARM:RPM:THR).

For a rundown measurement send ARM:RPM:MODE DOWN. The first arm occurs when the RPM value reaches the specified threshold value (specified with ARM:RPM:THR).

To specify subsequent arms to occur at RPM intervals measured from the starting RPM value see the ARM:TIMer command.

To disable the Start RPM Arming qualifier, send ARM:RPM:MODE OFF.

ARM:RPM:THReshold command/query

Specifies the starting RPM value.

Command Syntax: `ARM:RPM:THReshold (<number>[<unit>])|<step>|<bound>`

```
 <number> ::= a real number (NRf data)
              limits:  5:491520

   <unit> ::= [RPM]

   <step> ::= UP|DOWN

  <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Arm:Rpm:Thr 8000"
OUTPUT 711;"ARM:RPM:THR 1000"
```

Query Syntax: `ARM:RPM:THReshold?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +6.0 E+002
SCPI Compliance:  instrument-specific
```

Description:

This command is only valid when the Start RPM Arming qualifier is enabled with the ARM:RPM:MODE command.

The starting value is either the lowest RPM used for devices with increasing RPM (ARM:RPM:MODE UP) or the highest RPM for devices with decreasing RPM (ARM:RMP:MODE DOWN).

ARM:SOURce command/query

Specifies the type of arming for the analyzer's trigger.

Command Syntax:

```
ARM:SOURce {IMMediate|MANual|RPM|TIMer}
```

Example Statements:

```
OUTPUT 711;":arm:sour RPM"
OUTPUT 711;"Arm:Sour RPM"
```

Query Syntax:

```
ARM:SOURce?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  IMM
SCPI Compliance:  confirmed
```

Description:

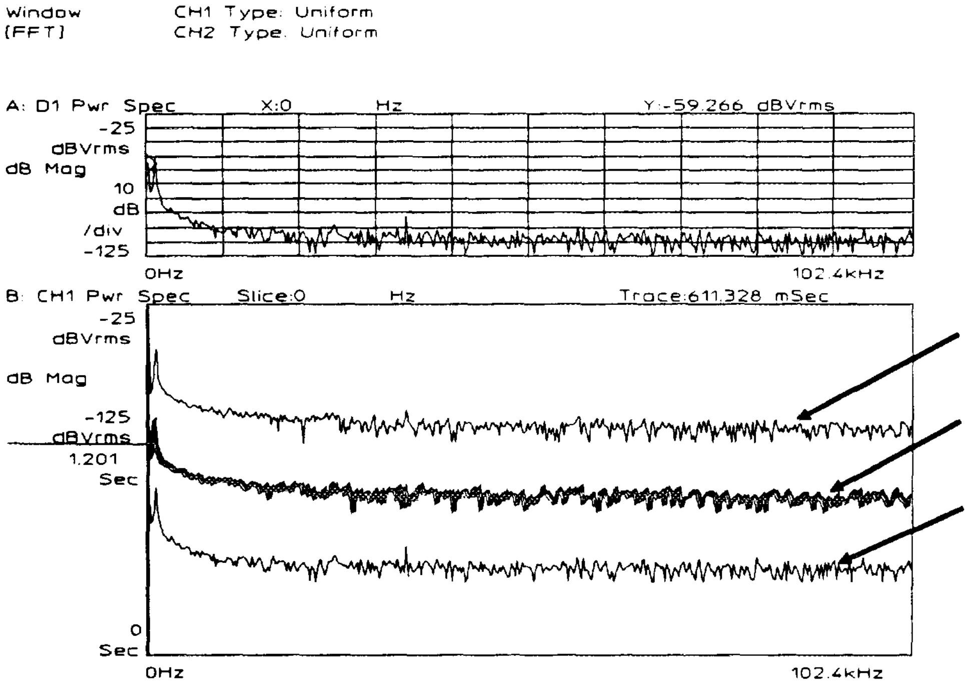
To select automatic arming send IMM. The analyzer waits for the hardware to settle and then waits for a trigger signal (specified by TRIG:SOUR) before starting the measurement. When the measurement is completed, the trigger is automatically re- armed.

To select manual arming send MAN. The analyzer waits for the hardware to settle then waits for the ARM[:IMM] command, and then waits for a trigger signal before starting the measurement. The ARM[:IMM] command must be sent to re-arm the trigger after the measurement is completed.

To select RPM step arming send RPM. The analyzer waits for the hardware to settle, waits for the next RPM level to be reached, and then waits for the trigger signal before starting the measurement. See the TRIGger:TACHometer commands for more information about using the analyzer's tachometer.

To select time step arming send TIM. The instrument waits for the hardware to settle, waits for a trigger and starts the measurement by collecting the first time record. The analyzer then waits until the end of the time step interval (specified by ARM:TIM) and waits for the trigger signal before collecting the next time record. In waterfall displays, *all* waterfall traces are referenced from the start time. See figure 10-2.

The [SENSe:]AVERage:TCONtrol REPeat command allows you to get a complete set of averages at each arming event during manual, RPM step and time step arming.



Window (FFT) CH1 Type: Uniform
CH2 Type: Uniform

Taken 1.2 seconds from start

Taken .6 seconds from start

Taken at start

Figure 10-2. Recorded Time of Triggered Event Referenced to Start Time

ARM:TIMer command/query

Specifies the size of the step used in time step arming.

Command Syntax:

```
ARM:TIMer {<number>[<unit>]}|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  0.0:500000
       <unit> ::= [S]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"ARM:TIMER .5"
OUTPUT 711;"arm:timer 3600"
```

Query Syntax: ARM:TIMer?

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +5.0 E-001
SCPI Compliance:  instrument-specific
```

Description:

This command is only valid with time step arming (ARM:SOUR TIM).

The default unit is seconds.

CALCulate

This subsystem contains commands that control the processing of measurement data. The block diagram in figure 11-1 shows you how measurement data is processed.

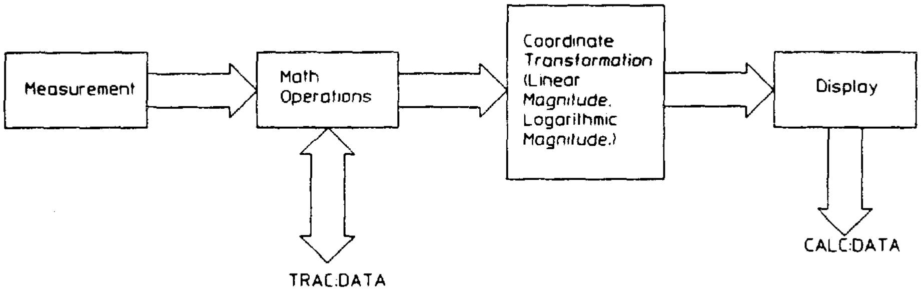

Measurement
Math Operations
Coordinate Transformation (Linear Magnitude, Logarithmic Magnitude.)
Display
TRAC:DATA
CALC:DATA

Figure 11-1. Flow of Measurement Data

After measurement data is collected, any specified math operations are performed. Data is then transformed into the specified coordinate system and sent to the display. TRAC:DATA gives you access to the raw measurement data after the analyzer performs math operations. CALC:DATA gives you access to the display data—after the coordinate transformation.

The CALCulate subsystem lets you:

- Specify the type of measurement.
- Select a coordinate system for display of the measurement data.
- Define trace math functions and constants.
- Curve fit and synthesize trace data.
- Perform limit testing.
- Control the analyzer's marker functions.
- Read marker values.
- Dump coordinate transformed data to your controller.

Note

You can transfer measurement data *from* the analyzer with either the TRAC:DATA command or the CALC:DATA command. However, you can only transfer data *to* the analyzer with the TRAC:DATA command.

The CALCulate mnemonic contains an optional trace specifier: [1|2]. To direct a command to trace A, omit the specifier or use 1. To direct a command to trace B, use 2. Commands that are not trace-specific—the CALC:MATH commands, the CALC:CFIT commands and the CALC:SYNT commands—ignore the specifier.

CALCulate[1|2]:ACTive command/query

Selects the active trace.

Command Syntax: `CALCulate[1|2]:ACTive {OFF|0|ON|1}`

Example Statements:

```
OUTPUT 711;":Calc2:Act OFF"
OUTPUT 711;"CALCULATE2:ACT ON"
```

Query Syntax: `CALCulate[1|2]:ACTive?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1 (trace A
               +0 (trace B)
SCPI Compliance:  instrument-specific
```

Description:

There is only one active trace at any time.

For example if CALC[2]:ACT ON is sent, trace B becomes the active trace and trace A is turned off. Similarly, if CALC[1]:ACT OFF is sent, trace A is turned off and trace B becomes the active trace.

Trace A is the default if the trace specifier is not used.

CALCulate[1|2]:CFIT:ABORt command

Aborts the curve fit operation.

Command Syntax:

```
CALCulate[1|2]:CFIT:ABORt
```

Example Statements:

```
OUTPUT 711;"calculate:cfit:abor"
OUTPUT 711;"Calculate:Cfit:Abort"
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command aborts the current curve fit operation. As a result of the CALC:CFIT:ABOR command, the curve fit table is not updated.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:COPY command

Copies the synthesis table into the curve fit table.

Command Syntax: `CALCulate[1|2]:CFIT:COPY {SYNThesis}`

Example Statements:

```
OUTPUT 711;":CALC:CFIT:COPY SYNThesis"
OUTPUT 711;"calc:cfit:copy SYNThesis"
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The synthesis table must be in pole-zero format. To convert a synthesis table to pole-zero format, use CALC:SYNT:TTYP PZER command.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:DATA command/query

Loads values into the curve fit table.

Command Syntax:

```
CALCulate[1|2]:CFIT:DATA <SYNFIT>

<SYNFIT> ::= see Description
```

Example Statements:

```
OUTPUT 711;"Calc:Cfit:Data SYNFIT"
OUTPUT 711;"CALCULATE:CFIT:DATA SYNFIT"
```

Query Syntax:

```
CALCulate[1|2]:CFIT:DATA?
```

Return Format:

```
<SYNFIT>
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command transfers a complete curve fit table from your controller to the analyzer.

When you transfer a curve fit table to the analyzer, you must use the definite length block syntax. Data must be 64-bit binary floating-point numbers (see the FORMat[:DATA] REAL command). The elements of the definite length block for a curve fit table are defined below. See "Block Data" in chapter 4 for additional information about transferring block data.

Note This command is not trace specific. It ignores the trace specifier.

```
         <SYNFIT> ::= <Point1><Point2> . . . <Point175>
         <Point1> ::= Table type
                      <0> = pole zero
                      <1> = pole residue
                      <2> = polynomial
         <Point2> ::= number_of_lines_in_left_column
         <Point3> ::= number_of_lines_in_right_column
         <Point4> ::= number_of_lines_in_Laurent_column
         <Point5> ::= real_part_first_term_in_left_column
         <Point6> ::= imaginary_part_first_term_in_left_column
         <Point7> ::= real_part_second_term_in_left_column
         <Point8> ::= imaginary_part_second_term_in_left_column
             •
             •
             •
        <Point47> ::= real_part_first_term_in_right_column
        <Point48> ::= imaginary_part_first_term_in_right_column
             •
             •
             •
        <Point89> ::= real_part_first_term_in_Laurent_column
        <Point90> ::= imaginary_part_first_term_in_Laurent_column
             •
             •
             •
       <Point131> ::= first_curve_fit_term_left_column
       <Point132> ::= second_curve_fit_term_left_column
 <curve_fit_term> ::= 0 for moveable
                      1 for fixed
             •
             •
             •
       <Point152> ::= first_curve_fit_term_right_column
       <Point153> ::= second_curve_fit_term_right_column
 <curve_fit_term> ::= 0 for moveable
                      1 for fixed
             •
             •
             •
       <Point173> ::= gain
       <Point174> ::= frequency_scale
      <Point 175> ::= time_delay
```

CALCulate[1|2]:CFIT:DESTination command/query

Selects the data register for the results of the curve fit operation.

Command Syntax:

```
CALCulate[1|2]:CFIT:DESTination (D1|D2|D3|D4|D5|D6|D7|D8)
```

Example Statements:

```
OUTPUT 711;":calculate2:cfit:destination D4"
OUTPUT 711;"Calc:Cfit:Destination D7"
```

Query Syntax:

```
CALCulate[1|2]:CFIT:DESTination?
```

Return Format: CHAR

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  D6
SCPI Compliance:  instrument-specific
```

Description:

This command specifies which data register holds the synthesis of the intermediate and final curve fit models. The default register is D6.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:FREQuency[:AUTO] command/query

Specifies the region included in the curve fit operation.

Command Syntax:

```
CALCulate[1|2]:CFIT:FREQuency[:AUTO] {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"CALC2:CFIT:FREQ:AUTO ON"
OUTPUT 711;"calculate:cfit:frequency ON"
```

Query Syntax:

```
CALCulate[1|2]:CFIT:FREQuency[:AUTO]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

To specify the full span send CALC:CFIT:FREQ ON.

To limit the region of the curve fit send CALC:CFIT:FREQ OFF. The commands, CALC:CFIT:FREQ:STAR and CALC:CFIT:FREQ:STOP define the region of a curve fit over a limited span.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:FREQuency:STARt command/query

Specifies the start frequency for a curve fit operation over a limited frequency span.

Command Syntax:

```
CALCulate[1|2]:CFIT:FREQuency:STARt
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:114999.9023
  <unit> ::= [HZ]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Calc2:Cfit:Frequency:Star 97344.6"
OUTPUT 711;"CALCULATE:CFIT:FREQ:START 316.834"
```

Query Syntax: `CALCulate[1|2]:CFIT:FREQuency:STARt?`

Return Format: NR3

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +0.00
SCPI Compliance:  instrument-specific
```

Description:

This command is not valid unless the CALC:CFIT:FREQ OFF command has been sent.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:FREQuency:STOP command/query

Specifies the stop frequency for a curve fit operation over a limited frequency span.

Command Syntax:

```
CALCulate[1|2]:CFIT:FREQuency:STOP
(<number>[<unit>])|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0.390625:115000.0
  <unit> ::= [HZ]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calc:cfit:freq:stop 53173.8"
OUTPUT 711;"Calculate:Cfit:Frequency:Stop 61248.7"
```

Query Syntax: `CALCulate[1|2]:CFIT:FREQuency:STOP?`

Return Format: NR3

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +5.12E+004
SCPI Compliance:  instrument-specific
```

Description:

This command is not valid unless the CALC:CFIT:FREQ OFF command has been sent.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:FSCale — command/query

Specifies the frequency scaling used in the curve fit operation.

Command Syntax:

```
CALCulate[1|2]:CFIT:FSCale <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  le-6:le6
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":CALC2:CFIT:FSC .19155"
OUTPUT 711;"calculate2:cfit:fsc 2"
```

Query Syntax:

```
CALCulate[1|2]:CFIT:FSCale?
```

Return Format: NR3

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +1.0
SCPI Compliance:  instrument-specific
```

Description:

The analyzer scales the frequency axis (the X-axis) by f/frequency scale, where f is frequency in Hertz.

Note

This command must be sent before CALC:CFIT. This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT[:IMMediate] command

Starts the curve fit process.

Command Syntax: `CALCulate[1|2]:CFIT[:IMMediate]`

Example Statements:

```
OUTPUT 711;"Calculate:Cfit:Immediate"
OUTPUT 711;"CALC:CFIT"
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The results of the curve fit are stored in the curve fit data register specified by the CALC:CFIT:DEST command.

To abort the curve fit operation, send CALC:CFIT:ABOR.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:ORDer:AUTO command/query

Determines the operation of the curve fitter.

Command Syntax:

```
CALCulate[1|2]:CFIT:ORDer:AUTO {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":calculate:cfit:order:auto ON"
OUTPUT 711;"Calc:Cfit:Ord:Auto OFF"
```

Query Syntax:

```
CALCulate[1|2]:CFIT:ORDer:AUTO?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

This command determines the curve fit operation mode. AUTO ON (the default value) places the curve fitter in automatic order selection. AUTO OFF gives a model with fixed numerator and denominator order.

In automatic order selection (CALC:CFIT:ORDER:AUTO is ON), the curve fit operation starts with 1 pole and 1 zero. If the fit is poor, the orders increment and another curve fit operation is performed. This iterative process continues until a model is found which more closely matches the measured frequency response. The curve fit operation uses the number of poles and zeros specified with the CALC:CFIT:ORD:POLES and CALC:CFIT:ORD:ZER commands as the upper bounds for this iterative search.

If CALC:CFIT:ORDER:AUTO OFF is sent, the curve fit operation provides a model with the number of poles and zeros specified with the CALC:CFIT:ORD:POLES and CALC:CFIT:ORD:ZER commands. There is no iterative search for a better model.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:ORDer:POLes command/query

Specifies the number of poles used in the curve fit operation.

Command Syntax:

```
CALCulate[1|2]:CFIT:ORDer:POLes <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:20
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"CALC:CFIT:ORDER:POL 12"
OUTPUT 711;"calculate:cfit:ord:poles 12"
```

Query Syntax:

```
CALCulate[1|2]:CFIT:ORDer:POLes?
```

Return Format: NR1

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +20
SCPI Compliance:  instrument-specific
```

Description:

The actual number of poles used in the curve fit operation is determined by the mode selected by the CALC:CFIT:ORD:AUTO command.

If CALC:CFIT:ORD:AUTO is ON (the default value), an optimum number of poles is used for the model. The number specified with this command represents the upper bounds for the iterative search. The number of poles will not exceed the number specified by CALC:CFIT:ORD:POL.

If CALC:CFIT:ORD:AUTO OFF is sent, the number of poles specified by CALC:CFIT:ORD:POL is used for the model.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:ORDer:ZERos **command/query**

Specifies the number of zeros used in the curve fit operation.

Command Syntax:

```
CALCulate[1|2]:CFIT:ORDer:ZERos <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:20
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Calc:Cfit:Ord:Zer 4"
OUTPUT 711;"CALCULATE2:CFIT:ORDER:ZER 17"
```

Query Syntax:

```
CALCulate[1|2]:CFIT:ORDer:ZERos?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +20
SCPI Compliance:  instrument-specific
```

Description:

The actual number of zeros used in the curve fit operation is determined by the mode selected by the CALC:CFIT:ORD:AUTO command.

If CALC:CFIT:ORD:AUTO is ON (the default value), an optimum number of zeros is used for the model. The number specified with this command represents the upper bounds for the iterative search. The number of zeros will not exceed the number specified by CALC:CFIT:ORD:ZER.

If CALC:CFIT:ORD:AUTO OFF is sent, the number of zeros specified by CALC:CFIT:ORD:POL is used for the model.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:TDELay **command/query**

Specifies a time delay value for the curve fit operation.

Command Syntax:

```
CALCulate[1|2]:CFIT:TDELay
{<number>[<unit>]}|<step>|<bound>

      <number> ::= a real number (NRf data)
                   limits:  -100:100
        <unit> ::= [S]
        <step> ::= UP|DOWN
       <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calc2:cfit:tdel .01 mS"
OUTPUT 711;"Calculate:Cfit:Tdel -1 uS"
```

Query Syntax: `CALCulate[1|2]:CFIT:TDELay?`

Return Format: `NR3`

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

Use this command to include a time delay value which removes any time delay from the frequency response to be fitted.

Positive delay is entered as a positive value.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:WEIGht:AUTO command/query

Determines the weighting function used in the curve fit operation.

Command Syntax:

```
CALCulate[1|2]:CFIT:WEIGht:AUTO (OFF|0|ON|1)
```

Example Statements:

```
OUTPUT 711;":CALCULATE2:CFIT:WEIGHT:AUTO OFF"
OUTPUT 711;"calc:cfit:weig:auto OFF"
```

Query Syntax:

```
CALCulate[1|2]:CFIT:WEIGht:AUTO?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

If CALC:CFIT:WEIGHT:AUTO is ON (the default value), the curve fit operation automatically generates a weighting function. It stores the result in the curve fit weight register. See CALC:CFIT:WEIG:REG command for more information about the curve fit weight register.

If CALC:CFIT:WEIGHT:AUTO OFF is sent, the curve fit operation uses the weighting function stored in the specified register. The curve fit operation will abort if the curve fit weight register is empty or contains invalid data.

Note

This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:CFIT:WEIGht:REGister command/query

Selects the data register which contains the weighting function for the curve fit operation.

Command Syntax:

```
CALCulate[1|2]:CFIT:WEIGht:REGister
(D1|D2|D3|D4|D5|D6|D7|D8)
```

Example Statements:

```
OUTPUT 711;"Calc:Cfit:Weight:Reg D6"
OUTPUT 711;"CALCULATE:CFIT:WEIG:REGISTER D7"
```

Query Syntax:

```
CALCulate[1|2]:CFIT:WEIGht:REGister?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  D7
SCPI Compliance:  instrument-specific
```

Description:

The default register is D7.

If you select the auto weight feature (the default is CALC:CFIT:WEIG:AUTO ON), the analyzer automatically generates a weighting function and stores it in the register specified with this command.

If you disable the auto weight feature by sending the CALC:CFIT:WEIG:AUTO OFF command, the curve fit operation uses the weighting function stored in the register specified with this command. The analyzer does not generate and store a new weighting function.

The curve fit operation will abort if the curve fit weight register is empty or contains invalid data.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:DATA? query

Returns trace data that has been transformed to the currently selected coordinate transform (specified with the CALC:FORMat command).

Query Syntax: `CALCulate[1|2]:DATA?`

Example Statements:

```
OUTPUT 711;":calc:data?"
OUTPUT 711;"Calc2:Data?"
```

Return Format: `DEF_BLOCK`

If FORMat[:DATA] REAL:

```
<DEF_BLOCK> ::= #<byte><length_bytes><1st_Y-axis_value> . . .
                <last_Y-axis_value>
     <byte> ::= number of length bytes to follow (ASCII encoded)
<length_bytes> ::= number of data bytes to follow (ASCII encoded)
```

If FORMat[:DATA] ASCii:

```
<DEF_BLOCK> ::= <1st_Y-axis_value> . . .<last_Y-axis_value>
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This query returns a definite length block of coordinate-transformed trace data.

The block is returned as a series of amplitude (Y-axis) values. The unit for these values is the same as the reference level unit. To determine the unit, send DISP:TRAC:Y:BOTT? UNIT.

The CALC:DATA:HEAD:POIN? query returns the number of discrete points along the specified display's X-axis. Refer to the TRACe:X:DATA command for information on retrieving X-axis values.

This query has no command form. Therefore, you cannot return trace data to the display with CALC:DATA. To send data that has *not* been transformed, use the TRAC:DATA command. See the introduction to this chapter for more information about the differences between these commands.

CALCulate[1|2]:DATA:HEADer:POINts? query

Returns the number of points in the data block returned with the CALC:DATA? query.

Query Syntax:

```
CALCulate[1|2]:DATA:HEADer:POINts?
```

Example Statements:

```
OUTPUT 711;"CALCULATE:DATA:HEADER:POIN?"
OUTPUT 711;"calc:data:head:points?"
```

Return Format:

NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The display's X-axis is divided into discrete points. Use this query to determine how many discrete points there are along the specified display's X-axis. This is the number of values sent to the analyzer's output queue when you send the CALC:DATA? query.

CALCulate[1|2]:FEED command/query

Selects the measurement data to be displayed in the specified trace.

Command Syntax:

```
CALCulate[1|2]:FEED <CMDSTR>

<CMDSTR> ::= command string
             see description
```

Example Statements:

```
OUTPUT 711;":Calc:Feed CMDSTR"
OUTPUT 711;"CALCULATE2:FEED CMDSTR"
```

Query Syntax:

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

Return Format:

```
STRING
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command selects the measurement results. The available measurement data varies for different instrument modes. See table 11-1 for a complete listing of measurement results and their related <CMDSTR> for each instrument mode.

In **FFT analysis instrument mode** (INST:SEL FFT), the following commands are available:

To select the contents of the specified data register, send CALC:FEED 'D1|D2|D3|D4|D5|D6|D7|D8'.

To select the contents of the time capture buffer for the specified channel, send CALC:FEED 'TCAP [1|2]'.

To select the contents of the specified waterfall register, send CALC:FEED 'W1|W2|W3|W4|W5|W6|W7|W8'.

To select the linear spectrum function for the specified channel, send CALC:FEED 'XFR:POW:LIN [1|2]'.

To select the power spectrum function for the specified channel, send CALC:FEED 'XFR:POW [1|2]'.

To select the coherence function, send CALC:FEED 'XFR:POW:COH 1,2'. INPut2 must be ON.

To select the cross spectrum, send CALC:FEED 'XFR:POW:CROS 1,2'. INPut2 must be ON.

To select the most recent frequency response function, send CALC:FEED 'XFR:POW:RAT 2,1'. INPut2 must be ON.

To select the most recent time record for the specified channel, send CALC:FEED 'XTIM:VOLT [1|2]'.

To select the most recent windowed time record for the specified channel, send CALC:FEED 'XTIM:VOLT:WIND [1|2]'.

To select the orbit diagram, send CALC:FEED 'XVOL:VOLT 1,2'. The orbit diagram presents channel 1 time along the X-axis and channel 2 time along the Y- axis. INPut2 must be ON.

In **correlation instrument mode** (SEL:INST CORR) the following commands are available:

CALC:FEED 'D1|D2|D3|D4|D5|D6|D7|D8'.

CALC:FEED 'TCAP [1|2]'.

CALC:FEED 'W1|W2|W3|W4|W5|W6|W7|W8'.

CALC:FEED 'XTIM:VOLT [1|2]'.

CALC:FEED 'XTIM:VOLT:WIND [1|2]'.

To select the most recent auto correlation for the specified channel, send CALC:FEED 'XTIM:CORR [1|2]'.

To select the most recent cross correlation for the specified channel, send CALC:FEED 'XTIM:CORR:CROS 1,2'.

In histogram instrument mode (SEL:INST HIST):

CALC:FEED 'D1|D2|D3|D4|D5|D6|D7|D8'.

CALC:FEED 'TCAP [1|2]'.

CALC:FEED 'W1|W2|W3|W4|W5|W6|W7|W8'.

To select the most recent unfiltered time record for the specified channel, send CALC:FEED 'XTIM:VOLT [1|2]'.

To select the most recent histogram for the specified channel, send CALC:FEED 'XTIM:VOLT:HIST [1|2]'.

To select the probability density function for the specified channel, send CALC:FEED 'XTIM:VOLT:PDF [1|2]'. The histogram is normalized to unit area.

To select the cumulative density function for the specified channel, send CALC:FEED 'XTIM:VOLT:CDF [1|2]'. This shows the probability that a level ≤ to a specific level occurred.

In order analysis instrument mode (SEL:INST ORD; Option 1D0), available commands are:

CALC:FEED 'D1|D2|D3|D4|D5|D6|D7|D8'.

CALC:FEED 'TCAP [1|2]'.

CALC:FEED 'W1|W2|W3|W4|W5|W6|W7|W8'.

CALC:FEED 'XFR:POW [1|2]'. Order track must be off (sent with the ORD:TRAC:STAT OFF command).

CALC:FEED 'XTIM:VOLT [1|2]'. Order track must be off (sent with the ORD:TRAC:STAT OFF command).

CALC:FEED 'XVOL:VOLT 1,2'. Order track must be off (sent with the ORD:TRAC:STAT OFF command). INPut2 must be on.

To display one of five order tracks, send CALC:FEED 'XORD:TRAC [1|2],[1|2|3|4|5]'. The first parameter specifies the channel. The second parameter specifies the order track. Order track must be on (sent with the ORD:TRAC:STAT ON command).

To display time versus RPM (which tells you how long the RPM runup or rundown took), send CALC:FEED 'XRPM:PROF'. Order track must be on (sent with the ORD: TRAC:STAT ON command).

To select composite power for the specified channel, send CALC:FEED 'XFR:POW: COMP [1|2]'. Order track must be on (sent with the ORD:TRAC:STAT ON command). The [SENSe:]WINDow:ORDer:DC command specifies whether the analyzer should use the dc bins in calculating composite power.

In octave analysis instrument mode (SEL:INST OCT; Option 1D1), available commands are:

CALC:FEED 'D1|D2|D3|D4|D5|D6|D7|D8'.

CALC:FEED 'TCAP [1|2]'.

CALC:FEED 'W1|W2|W3|W4|W5|W6|W7|W8'.

CALC:FEED 'XFR:POW [1|2]'.

In **swept sine instrument mode** (SEL:INST SINE; Option 1D2), available commands are:

CALC:FEED 'D1|D2|D3|D4|D5|D6|D7|D8'.

CALC:FEED 'W1|W2|W3|W4|W5|W6|W7|W8'.

CALC:FEED 'XFR:POW:LIN [1|2]'.

CALC:FEED 'XTIM:VOLT [1|2]'.

CALC:FEED 'XFR:POW:CROS 1,2'. INPut2 must be on.

CALC:FEED 'XFR:POW:RAT 2,1'. INPut2 must be on.

To display the normalized variance for the specified channel, send CALC:FEED 'XFR:POW:VAR [1|2]'.

Table 11-1. Measurement Results for each Instrument Mode

Measumrent Data CALC:FEED command	INSTrument:SELect					
	FFT	CORR	HIST	ORD	OCT	SINE
Auto correlation XTIM:CORR [1 | 2]		X				
Capture buffer TCAP [1 | 2]	X	X	X	X	X	
Coherence XFR:POW:COH 1,2	X					
Composite Power XFR:POW:COMP [1 | 2]				X		
Cross Correlation XTIM:CORR:CROS 1,2		X				
Cross Spectrum XFR:POW:CROS 1,2	X					X
Cumulative Density Function XTIM:VOLT:CDF [1 | 2]			X			
Data Register D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8	X	X	X	X	X	X
Frequence Response XFR:POW:RAT 2,1	X					X
Histogram XTIM:VOLT:HIST [1 | 2]			X			
Linear Spectrum XFR:POW:LIN [1 | 2]	X					X
Normalized Variance XFR:POW:VAR [1 | 2]						X
Orbit Diagram XVOL:VOLT 1,2	X			X		
Order Track XORD:TRACK [1 | 2],[1 | 2 | 3 | 4 | 5]				X		
Power Spectrum XFR:POW [1 | 2]	X			X	X	
Probability Density Function XTIM:VOLT:PDF [1 | 2]			X			
RPM Profile XRPM:PROF				X		
Time XTIM:VOLT [1 | 2]	X	X		X		X
Unfiltered Time XTIM:VOLT [1 | 2]			X			
Waterfall Register W1 | W2 | W3 | W4 | W5 | W6 | W7 | W8	X	X	X	X	X	X
Windowed Time XTIM:VOLT:WIND [1 | 2]	X	X				

CALCulate[1|2]:FORMat command/query

Selects a coordinate system for displaying measurement data and for transferring coordinate transformed data to a controller.

Command Syntax:

```
CALCulate[1|2]:FORMat
(MLINear|MLOGarithmic|PHASe|REAL|IMAGinary|NYQuist|UPHase)
```

Example Statements:

```
OUTPUT 711;"calculate2:form IMAGinary"
OUTPUT 711;"Calc:Format REAL"
```

Query Syntax:

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

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  MLOG trace A
               REAL trace B
SCPI Compliance:  confirmed
```

Description:

To select a coordinate system that displays linear magnitude data along the Y-axis, send CALC:FORM MLIN.

To display linear magnitude data on a logarithmic Y-axis scale, send CALC:FORM MLIN;:DISP:TRAC:Y:SPAC LOG.

To select a coordinate system that displays logarithmic magnitude data on a linear Y- axis scale, send CALC:FORM MLOG.

To select a coordinate system that displays wrapped phase along the Y-axis, send CALC:FORM PHAS. All phase is shifted to fall between -180° to +180°.

To select a coordinate system that displays imaginary numbers along the Y-axis, send CALC:FORM IMAG. This coordinate system shows the imaginary component of complex data at each point along the X-axis. If the data point is real rather than complex, a value of 0 is displayed for all X-axis points.

To select a coordinate system that displays real numbers along the Y-axis, send CALC:FORM REAL. This coordinate system shows real data or the real component of complex data at each point along the X-axis. This is the only valid selection for orbit displays (CALC:FEED 'XVOL:VOLT 1,2'). CALC:FORM REAL is the default selection for orbit displays.

To select a coordinate system that displays unwrapped phase along the Y-axis, send CALC:FORM UPH. The displayed phase is referenced to the lowest measured frequency. It is *not* shifted to fall between -180° to +180°. This is an approved SCPI command.

To select a coordinate system that displays a Nyquist diagram (imaginary numbers along the Y-axis and real numbers along the X-axis), send CALC:FORM NYQ. It is not valid for RPM Profiles (CALC:FEED 'XRPM:PROF'). This is an approved SCPI command.

To query display data—that is, the analyzer has applied a coordinate system to the measurement data—use the CALC:DATA command. To access "raw" measurement data—only math operations have been performed on the data—use the TRAC:DATA command.

CALCulate[1|2]:LIMit:BEEP[:STATe] command/query

Turns the limit-fail beeper on and off.

Command Syntax:

```
CALCulate[1|2]:LIMit:BEEP[:STATe] {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":CALC2:LIMIT:BEEP OFF"
OUTPUT 711;"calc:limit:beep:state ON"
```

Query Syntax:

```
CALCulate[1|2]:LIMit:BEEP[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

The limit-fail beeper emits an audible tone when all of the following conditions are met:

- CALC:LIM:BEEP is ON.
- CALC:LIM:STAT is ON.
- The trace falls outside its current limits.

You can use CALC:LIM:LOW:SEGM and CALC:LIM:UPP:SEGM or CALC:LIM:LOW:TRAC and CALC:LIM:UPP:TRAC to define a trace's current limits via the HP-IB.

If a trace specifier is not used, the command defaults to trace A.

CALCulate[1|2]:LIMit:FAIL? query

Returns the result of the last limit test; 0 for pass or 1 for fail.

Query Syntax:	`CALCulate[1|2]:LIMit:FAIL?`

Example Statements:	`OUTPUT 711;"Calc2:Lim:Fail?"`
	`OUTPUT 711;"CALC:LIMIT:FAIL?"`

Return Format:	`NR1`

Attribute Summary:	`Option: not applicable`
	`Overlapped: no`
	`Preset State: +1`
	`SCPI Compliance: confirmed`

Description:

This query returns "+0" if the trace passes the limit test. It returns a "+1" if the trace fails the limit test. Limit testing must be on (DISP:LIM:STAT ON) and a limit must be defined for the specified trace.

If limit testing is not on or limits are not defined, this query returns a +1 (fail).

Use the SYST:ERR? query to verify a failed limit test. If limit testing is not on, the SYST:ERR? query returns the message, "Limit testing is turned off." If limits are not defined the SYST:ERR? query returns the message, "Limits are undefined." If a valid limit test failed, the SYST:ERR? query does not return a message.

See "Limit Fail Register Set" in chapter 5 for additional information about the Limit Fail status registers.

Note CALC:LIM:FAIL? returns +1 (fail) if limit testing is not turned on or limits are not defined.

CALCulate[1|2]:LIMit:LOWer:CLEar[:IMMediate] command

Deletes the lower limit line from the specified display.

Command Syntax:

```
CALCulate[1|2]:LIMit:LOWer:CLEar[:IMMediate]
```

Example Statements:

```
OUTPUT 711;":calc2:limit:low:clear"
OUTPUT 711;"Calc:Lim:Lower:Cle:Immediate"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

To delete a lower limit line, send CALC:LIM:LOW:CLE. To delete an upper limit, send CALC:LIM:UPP:CLE.

You can delete part of a limit line if it consists of segments. See the CALC:LIM:LOW:SEGM:CLE command for information about deleting a segment of the lower limit line.

CALCulate[1|2]:LIMit:LOWer:MOVE:Y **command**

Moves all segments of the lower limit line up or down in the specified trace.

Command Syntax:

```
CALCulate[1|2]:LIMit:LOWer:MOVE:Y <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"CALC:LIM:LOWER:MOVE:Y -3.42181e+37"
OUTPUT 711;"calc:lim:lower:move:y 8.83814e+37"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command increments or decrements all segments in a lower limit by the specified value along the Y-axis. The value is unitless and assumes the current vertical/division unit (returned with DISP:TRAC:Y:PDIV? UNIT).

CALCulate[1|2]:LIMit:LOWer:REPort[:DATA]? query

Returns the X-axis value of the failed points for the lower limit test.

Query Syntax: `CALCulate[1|2]:LIMit:LOWer:REPort[:DATA]?`

Example Statements:

```
OUTPUT 711;":Calc2:Lim:Lower:Rep?"
OUTPUT 711;"CALCULATE:LIM:LOW:REPORT:DATA?"
```

Return Format: `<DEF_BLOCK>`

When data is ASCII-encoded, (FORMat ASCii) <DEF_BLOCK> takes the following form:

```
        <DEF_BLOCK> ::= [<X-axis value>,<X-axis value> . . .]
```

When data is binary-encoded, (FORMat REAL) <DEF_BLOCK> takes the following form:

```
        <DEF_BLOCK> ::= #<byte><length_bytes>[<X-axis value>,<X-axis value> . . .]
             <byte> ::= one ASCII-encoded byte specifying the number of
                        length bytes to follow
     <length_bytes> ::= ASCII-encoded bytes specifying the number of
                        data bytes to follow
```

The following definitions apply to both ASCII- and binary-encoded data.

```
    <X-axis value> ::= a real number (X-axis value of the failed point)
                       limits:  -9.9e+37 : 9.9e+37
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the X-axis value for data points which fail the lower limit test.

Data is not returned if limit testing is turned off (CALC:LIM:STAT OFF) or if all trace points are above the specified lower limit.

CALCulate[1|2]:LIMit:LOWer:REPort:YDATa? query

Returns the Y-axis value of the failed points for the lower limit test.

Query Syntax: `CALCulate[1|2]:LIMit:LOWer:REPort:YDATa?`

Example Statements:

```
OUTPUT 711;"calculate:lim:low:report:ydat?"
OUTPUT 711;"Calculate:Lim:Low:Report:Ydat?"
```

Return Format: `<DEF_BLOCK>`

When data is ASCII-encoded, (FORMat ASCii) <DEF_BLOCK> takes the following form:

```
    <DEF_BLOCK> ::= [<Y-axis value>,<Y-axis value> . . .]
```

When data is binary-encoded, (FORMat REAL) <DEF_BLOCK> takes the following form:

```
    <DEF_BLOCK> ::= #<byte><length_bytes>[<Y-axis value>,
                    <Y-axis value> . . .]
         <byte> ::= one ASCII-encoded byte specifying the number of
                    length bytes to follow
 <length_bytes> ::= ASCII-encoded bytes specifying the number of
                    data bytes to follow
```

The following definitions apply to both ASCII- and binary-encoded data.

```
 <Y-axis value> ::= a real number
                    (Y-axis value of the failed point)
                    limits:  - 9.9e+37 : 9.9e+37
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the Y-axis value for data points which fail the lower limit test.

Data is not returned if limit testing is turned off (CALC:LIM:STAT OFF) or if all trace points are above the specified lower limit.

CALCulate[1|2]:LIMit:LOWer:SEGMent command/query

Defines the lower limit as a series of line segments in the specified display.

Command Syntax: `CALCulate[1|2]:LIMit:LOWer:SEGMent <LIMIT>`

When data is ASCII-encoded, (FORMat ASCii) <LIMIT> takes the following form:

```
      <LIMIT> ::= <segment>[,<segment> . . . .]
    <segment> ::= <start_X-axis_value>, <start_Y- axis_value>
                  <stop_X-axis_value>, <stop_Y-axis_value>
```

When data is binary-encoded, (FORMat REAL) <LIMIT> takes the following form:

```
        <LIMIT> ::= #<byte>[<length_bytes>]<segment> [,<segment> . . .]
         <byte> ::= one ASCII-encoded byte specifying the number of
                    length bytes to follow
 <length_bytes> ::= ASCII-encoded bytes specifying the number of
                    data bytes to follow
      <segment> ::= <start_X-axis_value>, <start_Y- axis_value>
                    <stop_X-axis_value>, <stop_Y-axis_value>
```

The following definitions apply to both ASCII- and binary-encoded data.

```
<start_X-axis_value> ::= a real number
                         limits:  -9.9e+37 : 9.9e+37
<start_Y-axis_value> ::= a real number
                         limits:  -9.9e+37 : 9.9e+37
 <stop_X-axis_value> ::= a real number
                         limits:  -9.9e+37 : 9.9e+37
 <stop_Y-axis_value> ::= a real number
                         limits:  -9.9e+37 : 9.9e+37
```

Example Statements:

```
OUTPUT 711;":CALCULATE2:LIM:LOW:SEGMENT 10, 2, 100, 3"
OUTPUT 711;"calc2:limit:low:segm 200000, -5, 3000, -5,
80000, -2, 90000, -2"
```

Query Syntax: `CALCulate[1|2]:LIMit:LOWer:SEGMent?`

Return Format: definite length <LIMIT>

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command loads all segments of a limit. Each segment must consist of a start value (start_X-axis_value, start_Y-axis_value) and a stop value (stop_X-axis_value, stop_Y-axis_value).

The analyzer does not clear the previous lower limit definition when you send new segments. It only overwrites those portions of the limit redefined by the new segments. Send CALC:LIM:LOW:CLE to clear the previous limit.

CALCulate[1|2]:LIMit:LOWer:SEGMent:CLEar command

Deletes a segment from the lower limit line.

Command Syntax:

```
CALCulate[1|2]:LIMit:LOWer:SEGMent:CLEar <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Calculate2:Lim:Lower:Segm:Cle 10000"
OUTPUT 711;"CALCULATE:LIM:LOWER:SEGM:CLE 2000"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command deletes any segment which contains the X-axis value you specify. Adjacent segments are not affected, although the limit line may be discontinuous.

The value entered for a limit line segment is unitless.

To delete all segments of a lower limit line, send CALC:LIM:LOW:CLE.

CALCulate[1|2]:LIMit:LOWer:TRACe[:IMMediate] command

Converts the active trace into a lower limit line.

Command Syntax:

```
CALCulate[1|2]:LIMit:LOWer:TRACe[:IMMediate]
```

Example Statements:

```
OUTPUT 711;":calculate:lim:lower:trac"
OUTPUT 711;"Calc:Limit:Low:Trace:Imm"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Use the CALC[1|2]:ACTive command to specify the active trace.

CALCulate[1|2]:LIMit:STATe command/query

Turns limit testing on and off for the specified trace.

Command Syntax:

```
CALCulate[1|2]:LIMit:STATe {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"CALC:LIMIT:STAT OFF"
OUTPUT 711;"calculate:lim:stat ON"
```

Query Syntax:

```
CALCulate[1|2]:LIMit:STATe?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

When limit testing is on, the specified trace is evaluated against the limits defined in its upper and lower limit registers. If a trace specifier is not used, the command defaults to trace A. You can load these registers via the HP-IB using the CALC:LIM:LOW:SEGM and the CALC:LIM:UPP:SEGM commands or the CALC:LIM:UPP:TRAC and CALC:LIM:LOW:TRAC commands.

To determine whether or not a trace is within the specified limits, you can send the CALC:LIM:FAIL query or monitor the bits in the Limit Fail condition register. (For more information, see "Limit Fail Register Set" in chapter 5.)

To return failed points on the X-axis, send the CALC:LIM:LOW:REP and the CALC:LIM:UPP:REP queries. To return failed points on the Y-axis, send the CALC:LIM:LOW:REP:YDAT and the CALC:LIM:UPP:REP:YDAT queries.

Note

Limit lines are not automatically displayed when limit testing is enabled. To display limits you must send DISP[:WIND[1|2]]:LIM ON.

CALCulate[1|2]:LIMit:UPPer:CLEar[:IMMediate] command

Deletes the upper limit line from the specified display.

Command Syntax:

```
CALCulate[1|2]:LIMit:UPPer:CLEar[:IMMediate]
```

Example Statements:

```
OUTPUT 711;":Calculate2:Lim:Upper:Cle"
OUTPUT 711;"CALC:LIMIT:UPP:CLEAR:IMM"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

To delete an upper limit line, send CALC:LIM:UPP:CLE. To delete a lower limit line, send CALC:LIM:LOW:CLE.

You can delete part of a limit line if it consists of segments. See the CALC:LIM:UPP:SEGM:CLE command for information about deleting a segment of the upper limit line.

CALCulate[1|2]:LIMit:UPPer:MOVE:Y command

Moves all segments of the upper limit line up or down in the specified trace box.

Command Syntax:

```
CALCulate[1|2]:LIMit:UPPer:MOVE:Y <number>|<bound>

        <number> ::= a real number (NRf data)
                     limits:  -9.9e37:9.9e37
         <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calc:limit:upp:move:y -.02"
OUTPUT 711;"Calc:Limit:Upp:Move:Y -5"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command increments or decrements all segments in a upper limit by the specified value along the Y-axis. The value is unitless and assumes the current vertical/division unit (returned with DISP:TRAC:Y:PDIV? UNIT).

To specify trace box A, send CALC1:LIM:UPP:MOVE:Y. To specify trace box B, send CALC2:LIM:UPP:MOVE:Y. If a trace specifier is not included in the command, the trace specifier defaults to trace box A.

CALCulate[1|2]:LIMit:UPPer:REPort[:DATA]? query

Returns the X-axis value of the failed points for the upper limit test.limit test;reporting failed points

Query Syntax: `CALCulate[1|2]:LIMit:UPPer:REPort[:DATA]?`

Example Statements:

```
OUTPUT 711;":CALC2:LIMIT:UPP:REPORT?"
OUTPUT 711;"calc2:lim:upper:rep:data?"
```

Return Format: `<DEF_BLOCK>`

When data is ASCII-encoded, (FORMat ASCii) <DEF_BLOCK> takes the following form:

```
<DEF_BLOCK> ::= [<X-axis value>,<X-axis value> . . .]
```

When data is binary-encoded, (FORMat REAL) <DEF_BLOCK> takes the following form:

```
    <DEF_BLOCK> ::= #<byte><length_bytes>[<X-axis value>,<X-axis value> . . .]
         <byte> ::= one ASCII-encoded byte specifying the number of
                    length bytes to follow
<length_bytes> ::= ASCII-encoded bytes specifying the number of
                    data bytes to follow
```

The following definitions apply to both ASCII- and binary-encoded data.

```
<X-axis value> ::= a real number (X-axis value of the failed point)
                   limits:  -9.9e+37 : 9.9e+37
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the X-axis value for data points which fail the upper limit test.

Data is not returned if limit testing is turned off (CALC:LIM:STAT OFF) or if all trace points are below the specified upper limit.

CALCulate[1|2]:LIMit:UPPer:REPort:YDATa? query

Returns the Y-axis value of the failed points for the upper limit test.

Query Syntax: `CALCulate[1|2]:LIMit:UPPer:REPort:YDATa?`

Example Statements:

```
OUTPUT 711;"Calc:Lim:Upper:Rep:Ydata?"
OUTPUT 711;"CALC:LIM:UPPER:REP:YDATA?"
```

Return Format: `<DEF_BLOCK>`

When data is ASCII-encoded, (FORMat ASCii) <DEF_BLOCK> takes the following form:

```
<DEF_BLOCK> ::= [<Y-axis value>,<Y-axis value> . . .]
```

When data is binary-encoded, (FORMat REAL) <DEF_BLOCK> takes the following form:

```
      <DEF_BLOCK> ::= #<byte><length_bytes>[<Y-axis value>,<Y-axis value> . . .]
           <byte> ::= one ASCII-encoded byte specifying the number of
                      length bytes to follow
   <length_bytes> ::= ASCII-encoded bytes specifying the number of
                      data bytes to follow
```

The following definitions apply to both ASCII- and binary-encoded data.

```
   <Y-axis value> ::= a real number (Y-axis value of the failed point)
                      limits:  -9.9e+37 : 9.9e+37
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the Y-axis value for data points which fail the upper limit test.

Data is not returned if limit testing is turned off (CALC:LIM:STAT OFF) or if all trace points are below the specified upper limit.

CALCulate[1|2]:LIMit:UPPer:SEGMent command/query

Defines the upper limit as a series of line segments in the specified display.

Command Syntax: `CALCulate[1|2]:LIMit:UPPer:SEGMent <LIMIT>`

When data is ASCII-encoded, (FORMat ASCii) <LIMIT> takes the following form:

```
          <LIMIT> ::= <segment>[,<segment> . . .]
        <segment> ::= <start_X-axis_value>, <start_Y- axis_value>
                      <stop_X-axis_value>, <stop_Y-axis_value>
```

When data is binary-encoded, (FORMat REAL) <LIMIT> takes the following form:

```
          <LIMIT> ::= #<byte>[<length_bytes>]<segment> [<segment> . . .]
           <byte> ::= one ASCII-encoded byte specifying the number
                      of length bytes to follow
   <length_bytes> ::= ASCII-encoded bytes specifying the number
                      of data bytes to follow
        <segment> ::= <start_X-axis_value>, <start_Y- axis_value>
                      <stop_X-axis_value>, <stop_Y-axis_value>
```

The following definitions apply to both ASCII- and binary-encoded data.

```
<start_X-axis_value> ::= a real number
                         limits:  -9.9e+37 : 9.9e+37
<start_Y-axis_value> ::= a real number
                         limits:  -9.9e+37 : 9.9e+37
 <stop_X-axis_value> ::= a real number
                         limits:  -9.9e+37 : 9.9e+37
 <stop_Y-axis_value> ::= a real number
                         limits:  -9.9e+37 : 9.9e+37
```

Example Statements:

```
OUTPUT 711;":calc2:lim:upper:segm 10, 2, 100, 3"
OUTPUT 711;"Calculate2:Lim:Upp:Segment 20000, -5, 3000,
-5, 80000, -2, 90000, -2"
```

Query Syntax: `CALCulate[1|2]:LIMit:UPPer:SEGMent?`

Return Format: `definite length <LIMIT>`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command loads all segments of a limit. Each segment must consist of a start value (start_X-axis_value, start_Y-axis_value) and a stop value (stop_X-axis_value, stop_Y-axis_value).

The analyzer does not clear the previous upper limit definition when you send new segments. It only overwrites those portions of the limit redefined by the new segments. Send CALC:LIM:UPP:CLE to clear the previous limit.

CALCulate[1|2]:LIMit:UPPer:SEGMent:CLEar command

Deletes a segment from the upper limit line.

Command Syntax:

```
CALCulate[1|2]:LIMit:UPPer:SEGMent:CLEar <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"CALC:LIMIT:UPP:SEGM:CLEAR 50000"
OUTPUT 711;"calc:limit:upp:segm:clear .002"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command deletes any segment which contains the X-axis value you specify. Adjacent segments are not affected, although the limit line may be discontinuous.

The value entered for a limit line segment is unitless.

To delete all segments of an upper limit line, send CALC:LIM:UPP:CLE.

CALCulate[1|2]:LIMit:UPPer:TRACe[:IMMediate] command

Converts the active trace into an upper limit line.

Command Syntax:

```
CALCulate[1|2]:LIMit:UPPer:TRACe[:IMMediate]
```

Example Statements:

```
OUTPUT 711;":Calc2:Limit:Upp:Trac"
OUTPUT 711;"CALCULATE2:LIM:UPPER:TRAC:IMM"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Use the CALC[1|2]:ACTive command to specify the active trace.

CALCulate[1|2]:MARKer:BAND:STARt command/query

Specifies the lowest frequency of the band in which power is calculated.

Command Syntax:

```
CALCulate[1|2]:MARKer:BAND:STARt
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
  <unit> ::= [HZ|ORD|S]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calculate2:mark:band:star 0.0025 S"
OUTPUT 711;"Calc:Marker:Band:Start 10000 HZ"
```

Query Syntax: `CALCulate[1|2]:MARKer:BAND:STARt?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command defines the start value for the band used in calculating marker functions selected with the CALC:MARK:FUNC command. The specified value affects only the *currently selected* marker function.

The value specified with the CALC:MARK:BAND:STAR command *must be less* than the value specified with the CALC:MARK:BAND:STOP command.

Note If you want to move the frequency band below the current frequency band, you must reset the start value first. If you want to move the frequency band above the current frequency band, you must reset the stop value first.

To increment the value to the next largest point on the X-axis, send CALC:MARK:BAND:STAR UP.

To decrement the value to the next smallest point on the X-axis, send CALC:MARK:BAND:STAR DOWN.

You can also set the value with an expression. Send CALC:MARK:BAND:STAR (CALC:MARK:X?) to set the value to the current X-axis marker value.

If the X-axis is in time, this command specifies the start time for computation of time domain parameters. The default position is at the left edge of the trace.

The query returns the value of the current start frequency of the band (or the start time) in X-axis units. The value is returned even if the band markers are not on.

To determine the X-axis units, send CALC:MARK:BAND:STAR? UNIT.

CALCulate[1|2]:MARKer:BAND:STOP command/query

Specifies the highest frequency of the band in which power is calculated.

Command Syntax:

```
CALCulate[1|2]:MARKer:BAND:STOP
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
  <unit> ::= {HZ|ORD|S]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":CALC:MARK:BAND:STOP 75000 HZ"
OUTPUT 711;"calculate:mark:band:stop .0028 S"
```

Query Syntax: `CALCulate[1|2]:MARKer:BAND:STOP?`

Return Format: `NR3`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command defines the stop value for the band used in calculating marker functions specified with the CALC:MARK:FUNC command. The specified value affects only the *currently selected* marker function.

The value specified with the CALC:MARK:BAND:STOP command *must be greater* than the value specified with the CALC:MARK:BAND:STARt command.

Note If you want to move the frequency band below the current frequency band, you must reset the start value first. If you want to move the frequency band above the current frequency band, you must reset the stop value first.

To increment the value to the next largest point on the X-axis, send CALC:MARK:BAND:STOP UP.

To decrement the value to the next smallest point on the X-axis, send CALC:MARK:BAND:STOP DOWN.

You can also set the value with an expression. Send CALC:MARK:BAND:STOP (CALC:MARK:X?) to set the value to the current X-axis marker value.

If the X-axis is in time, this command specifies the stop time for computation of time domain parameters. The default position is at the right edge of the trace.

The query returns the value of the current stop frequency of the band (or the stop time) in X-axis units. The value is returned even if the band markers are not on.

To determine the X-axis units, send CALC:MARK:BAND:STOP? UNIT.

CALCulate[1|2]:MARKer:COUPled[:STATe] command/query

Couples the marker on the inactive trace with the marker of the active trace.

Command Syntax:

```
CALCulate[1|2]:MARKer:COUPled[:STATe] {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"Calc:Marker:Coup:Stat ON"
OUTPUT 711;"CALCULATE:MARK:COUPLED OFF"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:COUPled[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

This command moves the main maker of the inactive trace to the same X-axis point as the marker of the active trace. This ties the movement of the main markers together when used with the upper/lower display format (DISP:FORM ULOW) or with the front/back display format (DISP:FORM FBACK).

The position of each marker is updated, even when the active trace alone is displayed. You can not move a marker beyond the maximum number of points in the active trace.

When coupled markers are used in a zoomed measurement (starting frequency > 0), the first point of both traces is assumed to be the same (equal).

Note

This command couples the markers for each trace by X-axis position; *not* X-axis values.

CALCulate[1|2]:MARKer:FUNCtion command/query

Selects one of the analyzer's marker functions.

Command Syntax:

```
CALCulate[1|2]:MARKer:FUNCtion
{OFF|0|HPOWer|THD|BPOWer|BRMS|SPOWer|OVERshoot|RTIMe
|STIMe|DTIMe|SSLevel|GMARgin|PMARgin|GCRossover|PCRossover
|FREQuency|DAMPing|SINFo|WEIGht|TPOWer}
```

Example Statements:

```
OUTPUT 711;":calc:mark:function PCRossover"
OUTPUT 711;"Calc:Marker:Func SPOWer"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:FUNCtion?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command selects the marker function; send CALCulate:MARKer:RESult? to read marker values.

To turn off the marker function, send CALC:MARK:FUNC OFF.

Marker functions depend on the type of measurement data. To define the range of the calculation, use the CALC:MARK:BAND commands.

The following marker functions are available for **frequency data**:

To select the harmonic power function, send CALC:MARK:FUNC HPOW.

To select the total harmonic distortion function, send CALC:MARK:FUNC THD.

To select the band power function, send CALC:MARK:FUNC BPOW.

To select the square root of the band power function, send CALC:MARK:FUNC BRMS.

To select the sideband power function, send CALC:MARK:FUNC SPOW.

The following marker functions are available for **frequency response data** (CALC:FEED 'XFR:POW:RAT 2,1'):

To select the gain margin power function, send CALC:MARK:FUNC GMAR.

To select the phase margin power function, send CALC:MARK:FUNC PMAR.

To select the gain crossover function, send CALC:MARK:FUNC GCR.

To select the phase crossover power function, send CALC:MARK:FUNC PCR.

To select the resonant frequency, send CALC:MARK:FUNC FREQ.

To select the damping function, send CALC:MARK:FUNC DAMP.

The following marker functions are available for **time data** (CALC:FEED 'XTIM:VOLT[1|2]'.

To select the delay time function, send CALC:MARK:FUNC DTIM.

To select the maximum overshoot function, send CALC:MARK:FUNC OVER.

To select the rise time function, send CALC:MARK:FUNC RTIM.

To select the settling time function, send CALC:MARK:FUNC STIM.

To select the steady state level function, send CALC:MARK:FUNC SSL.$Imarker functions;steady state level

In **correlation analysis instrument mode** (INST:SEL CORR), the following marker functions are available:

To select the delay time function, send CALC:MARK:FUNC DTIM.

To select the maximum overshoot function, send CALC:MARK:FUNC OVER.

To select the rise time function, send CALC:MARK:FUNC RTIM.

To select the settling time function, send CALC:MARK:FUNC STIM.

To select the steady state level function, send CALC:MARK:FUNC SSL.

In **octave analysis instrument mode** (INST:SEL OCT; Option 1D1) the following marker functions are available:

To select the band power function, send CALC:MARK:FUNC BPOW.

To select the square root of the band power function, send CALC:MARK:FUNC BRMS.

To select the total power bin, send CALC:MARK:FUNC TPOW.

In **order analysis instrument mode** (INST:SEL ORD; Option 1D0) the following marker functions are available:

To select the harmonic power function, send CALC:MARK:FUNC HPOW.

To select the total harmonic distortion function, send CALC:MARK:FUNC THD.

To select the band power function, send CALC:MARK:FUNC BPOW.

To select the square root of the band power function, send CALC:MARK:FUNC BRMS.

To select the sideband power function, send CALC:MARK:FUNC SPOW.

In **histogram analysis instrument mode** (INST:SEL HIST) the following marker functions are available for unfiltered time data (CALC:FEED 'XTIM:VOLT [1|2]'):

To select the delay time function, send CALC:MARK:FUNC DTIM.

To select the maximum overshoot function, send CALC:MARK:FUNC OVER.

To select the rise time function, send CALC:MARK:FUNC RTIM.

To select the settling time function, send CALC:MARK:FUNC STIM.

To select the steady state level function, send CALC:MARK:FUNC SSL.

Note

Marker functions are *not* available for following types of trace data:
Nyquist diagram (CALC:FORM NYQ)
Orbit diagram (CALC:FEED 'XVOL:VOLT 1,2')
Waterfall displays (DISP:FORM WAT)

In addition to function calculations, use this command to determine if one of the filters (A-weight, B-weight, or C-weight) was applied to the measurement data or to determine the Z-axis value for the measurement data.

To determine if a filter was applied to the measurement data, send CALC:MARK:FUNC WEIGht. Send the CALC:MARK:FUNC:RES? query to determine the results. The query returns 1 if the A-weight filter was applied to the measurement data; 2 if the B-weight filter was applied and 3 if the C-weight filter was applied. If a filter was *not* applied, the query returns 0 (false). The query returns 4 if the value is undefined. For example, if you are using two data registers in a math expression, one which used the A- weight filter and the other which used the B-weight filter, the results are undefined.

To determine the Z-axis value for the measurement, send CALC:MARK:FUNC SINFo. Send the CALC:MARK:FUNC:RES? query to determine the Z-axis value for the measurement data. The Z-axis value indicates where the measurement data was extracted from a waterfall. It tells you when the measurement data was armed.

CALCulate[1|2]:MARKer:FUNCtion:RESult? query

Returns the result of the calculation for the currently selected marker function.

Query Syntax:

```
CALCulate[1|2]:MARKer:FUNCtion:RESult?
```

Example Statements:

```
OUTPUT 711;"CALC:MARKER:FUNC:RESULT?"
OUTPUT 711;"calc:mark:function:res?"
```

Return Format:

```
NR3
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command returns the value of the marker function result calculation.

For example, send CALC:MARK:FUNC GMAR then send CALC:MARK:FUNC:RES? to query the value of the gain margin. To determine the units, send CALC:MARK:FUNC:RES? UNIT.

Refer to the CALC:MARK:FUNCtion command for a complete listing of the available marker functions.

Caution The analyzer returns the result 9.91E37, if it can *not* calculate a marker function.

In addition to function calculations, use this command to determine if one of the filters (A-weight, B-weight, or C-weight) was applied to the measurement data or to determine the Z-axis value for the measurement data.

To determine if a filter was applied to the measurement data, send CALC:MARK:FUNC WEIGht. Send the CALC:MARK:FUNC:RES? query to determine the results. The query returns 1 if the A-weight filter was applied to the measurement data; 2 if the B-weight filter was applied and 3 if the C-weight filter was applied. If a filter was *not* applied, the query returns 0 (false). The query returns 4 if the value is undefined. For example, if you are using two data registers in a math expression, one which used the A- weight filter and the other which used the B-weight filter, the value is undefined.

To determine the Z-axis value for the measurement, send CALC:MARK:FUNC SINFo. Send the CALC:MARK:FUNC:RES? query to determine the Z-axis value for the measurement data. The Z-axis value indicates where the measurement data was extracted from a waterfall. It tells you when the measurement data was armed.

CALCulate[1|2]:MARKer:HARMonic:COUNt command/query

Specifies the maximum number of harmonic markers for the display.

Command Syntax:

```
CALCulate[1|2]:MARKer:HARMonic:COUNt
<number>|<step>|<bound>

    <number> ::= a real number (NRf data)
                 limits:  0:400
      <step> ::= UP|DOWN
     <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Calculate2:Mark:Harm:Count 345"
OUTPUT 711;"CALC2:MARKER:HARM:COUN 270"
```

Query Syntax: `CALCulate[1|2]:MARKer:HARMonic:COUNt?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The value can be specified numerically or with nonnumeric parameters. The actual number of harmonics is determined by the analyzer.

To increase the number of harmonic markers by one, send CALC:MARK:HARM:COUN UP.

To decrease the number of harmonic markers by one, send CALC:MARK:HARM:COUN DOWN.

The query returns the number of harmonic currently specified for the display. The value is returned even if the harmonic markers are not on.

CALCulate[1|2]:MARKer:HARMonic:FUNDamental command/query

Specifies the fundamental frequency for harmonic markers and calculations.

Command Syntax:

```
CALCulate[1|2]:MARKer:HARMonic:FUNDamental
{<number>[<unit>]}|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  0.0:102400.0
       <unit> ::= [HZ|ORD]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calculate:mark:harmonic:fund 46949.4"
OUTPUT 711;"Calc:Marker:Harm:Fundamental 98266.7"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:HARMonic:FUNDamental?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The value can be specified numerically or with nonnumeric parameters.

To increment the value to the next largest point on the X-axis, send CALC:MARK:HARM:FUND UP.

To decrement the value to the next smallest point on the X-axis, send CALC:MARK:HARM:FUND DOWN.

You can also set the value with an expression. Send CALC:MARK:FUND (CALC:MARK:X?) to set the value to the current X-axis marker value.

The query returns the value of the fundamental frequency (in X-axis units) currently used for harmonic markers and calculations. The value is returned even if the harmonic markers are not on.

To determine the X-axis units, send CALC:MARK:HARM:FUND? UNIT.

CALCulate[1|2]:MARKer:MAXimum[:GLOBal] command

Moves the main marker to the highest point on the specified trace.

Command Syntax:

```
CALCulate[1|2]:MARKer:MAXimum[:GLOBal]
```

Example Statements:

```
OUTPUT 711;":CALC2:MARK:MAXIMUM"
OUTPUT 711;"calc:marker:max:glob"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command moves the marker to the highest peak one time. Another command —CALC:MARK:MAX:TRAC—controls a marker function that automatically moves the marker to the highest peak each time the trace is updated.

CALCulate[1|2]:MARKer:MAXimum[:GLOBal]:TRACk command/query

Turns the peak tracking function on or off.

Command Syntax:

```
CALCulate[1|2]:MARKer:MAXimum[:GLOBal]:TRACk {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"Calculate:Mark:Maximum:Glob:Trac ON"
OUTPUT 711;"CALCULATE:MARK:MAXIMUM:TRAC OFF"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:MAXimum[:GLOBal]:TRACk?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

When peak tracking is enabled, the analyzer automatically positions the main marker on the largest peak of the active trace (CALC:ACT ON) each time the trace is updated.

To move the marker to the highest peak one time, use the CALC:MARK:MAX:GLOB command.

CALCulate[1|2]:MARKer:MAXimum:LEFT command

Moves the main marker one peak to the left of its current location on the specified trace.

Command Syntax:

```
CALCulate[1|2]:MARKer:MAXimum:LEFT
```

Example Statements:

```
OUTPUT 711;":calc2:marker:max:left"
OUTPUT 711;"Calc2:Mark:Maximum:Left"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

A peak is a local maximum on the *displayed* trace. The slope of a trace is positive to the left of a peak and negative to the right. In addition, the slope on one side of a peak must not change for at least one vertical division (one-third division on both sides if in octave analysis instrument mode or one-tenth division (one decade) if using logarithmic scaling of the Y-axis).

This command only finds peaks that are at least one point to the left of the current marker position. If the analyzer does not find a peak, the marker does not move. You can increase the number of peaks found by the analyzer by decreasing the value of vertical scale division (DISP:TRAC:Y:PDIV).

CALCulate[1|2]:MARKer:MAXimum:RIGHt command

Moves the main marker one peak to the right of its current location on the specified trace.

Command Syntax:

```
CALCulate[1|2]:MARKer:MAXimum:RIGHt
```

Example Statements:

```
OUTPUT 711;"CALCULATE2:MARK:MAX:RIGHT"
OUTPUT 711;"calc:marker:max:righ"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

A peak is a local maximum on the *displayed* trace. The slope of a trace is positive to the left of a peak and negative to the right. In addition, the slope on one side of a peak must not change for at least one vertical division (one-third division on both sides if in octave analysis instrument mode or one-tenth division (one decade) if using logarithmic scaling of the Y-axis).

This command only finds peaks that are at least one point to the right of the current marker position. If the analyzer does not find a peak, the marker does not move. You can increase the number of peaks found by the analyzer by decreasing the value of vertical scale division (DISP:TRAC:Y:PDIV).

CALCulate[1|2]:MARKer:MODE command/query

Selects absolute or relative marker values.

Command Syntax:

```
CALCulate[1|2]:MARKer:MODE {ABSolute|RELative}
```

Example Statements:

```
OUTPUT 711;":Calculate2:Mark:Mode ABSolute"
OUTPUT 711;"CALC2:MARK:MODE ABSolute"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:MODE?
```

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  ABS
SCPI Compliance:  instrument-specific
```

Description:

To select relative marker values, send CALC:MARK:MODE REL. A marker reference is displayed and marker values are reported as relative distances between the reference point and the marker position.

To select absolute marker values, send CALC:MARK:MODE ABS. The marker values are reported as the position of the marker on the trace.

CALCulate[1|2]:MARKer:POSition command/query

Specifies the main marker's independent axis position.

Command Syntax:

```
CALCulate[1|2]:MARKer:POSition (<number>[<unit>])|<step>|<bound>

 <number> ::= a real number (NRf data)
                limits:  -9.9e37:9.9e37
   <unit> ::= [HZ|S|ORD|COUNT|AVG|V|VPK|RPM|EU|REV]
   <step> ::= UP|DOWN
  <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calc2:marker:pos .013 S"
OUTPUT 711;"Calc:Marker:Pos 102400 HZ"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:POSition?
```

Return Format:

```
NR3
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command is identical to the CALC:MARK:X command *with the exception* of orbits measurement data (CALC:FEED 'XVOL:VOLT 1,2') or when measurement data is displayed in a Nyquist diagram (CALC:FORM NYQ).

In **orbits** (CALC:FEED 'XVOL:VOLT 1,2'), the independent axis is labeled time (T). The main marker displays the values for the three axis; X for time channel 1 amplitude value, Y for the time channel 2 amplitude value and T for the independent axis (in seconds). This command moves the main marker to a position along the T-axis.

In a **Nyquist diagram** (CALC:FORM NYQ) the X-axis is the real component of the measurement data and the Y-axis is the imaginary component. The independent axis is determined by the instrument mode. (Depending upon the instrument mode, the independent axis may be in terms of frequency, time, volts, RPM or orders.) This command moves the main marker to a position along the independent axis.

Note

For measurement data other than orbits or Nyquist diagrams this command behaves like the CALC:MARK:X[:ABSolute] command.

CALCulate[1|2]:MARKer:POSition:POINt command/query

Moves the main marker to a specific display point.

Command Syntax:

```
CALCulate[1|2]:MARKer:POSition:POINt
<number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:2047

  <step> ::= UP|DOWN

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":CALCULATE2:MARK:POS:POINT 223"
OUTPUT 711;"calc2:marker:pos:poin 1158"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:POSition:POINt?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the main marker's X-axis position by point number.

A trace composed of frequency data displays 401 points along the X-axis. A trace composed of time data displays 1024 points along the X-axis.

In correlation analysis there are 512 points.

In histogram analysis, the number of points is determined by the number of bins (set with the HIST:BINS command). The maximum number is 1024 points.

In octave analysis, the number of points is determined by the bandwidth of the filters. There are 11 points for full octave, 33 points for 1/3 octave and 132 points for 1/12 octave.

In some cases the number of points is arbitrary. These include swept sine measurement data, order tracking measurement data and time capture data.

To specify the main marker's position by a value along the X-axis, use the CALC:MARK:X command. To specify the main marker's position by a value along the independent axis, use the CALC:MARK:POS.

Note You *cannot* move the main marker beyond the maximum displayed point nor below the minimum displayed point.

CALCulate[1|2]:MARKer:REFerence:X command/query

Specifies the marker reference X-axis position.

Command Syntax:

```
CALCulate[1|2]:MARKer:REFerence:X
(<number>[<unit>])|<step>|<bound>

    <number> ::= a real number (NRf data)
                 limits:  -9.9e37:9.9e37
      <unit> ::= [HZ|S|ORD|COUNT|AVG|V|VPK|RPM|EU|REV]
      <step> ::= UP|DOWN
     <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Calculate:Mark:Reference:X 35000 HZ"
OUTPUT 711;"CALC:MARKER:REF:X .003 S"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:REFerence:X?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the absolute X-axis position for the reference marker.

To specify the marker reference X-axis position relative to the main marker's position, use the CALC:MARK:X:REL command.

CALCulate[1|2]:MARKer:REFerence:Y command/query

Specifies the marker reference Y-axis position.

Command Syntax:

```
CALCulate[1|2]:MARKer:REFerence:Y
{<number>[<unit>]}|<step>|<bound>

     <number> ::= a real number (NRf data)
                    limits:  -9.9e37:9.9e37
       <unit> ::= [DBVRMS|VPK|DBVPK|V|DBV|EU|DBEU|VRMS]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":calc:mark:reference:y 1 VPK"
OUTPUT 711;"Calculate2:Mark:Ref:Y -3 DBVRMS"
```

Query Syntax: `CALCulate[1|2]:MARKer:REFerence:Y?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the absolute Y-axis position for the reference marker.

To specify the marker reference Y-axis position relative to the main marker's position, use the CALC:MARK:Y:REL command.

CALCulate[1|2]:MARKer:SIDeband:CARRier command/query

Specifies the carrier frequency used for sideband markers and calculations.

Command Syntax:

```
CALCulate[1|2]:MARKer:SIDeband:CARRier
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
               limits:  0.0:115000.0
  <unit> ::= [HZ|ORD]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"CALC:MARKER:SID:CARR 20753.8"
OUTPUT 711;"calculate:mark:sideband:carr 108491"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:SIDeband:CARRier?
```

Return Format:

```
NR3
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

To specify the number of sidebands, send the CALC:MARK:SID:COUN command.

To increment the carrier frequency value to the next largest point on the X-axis, send CALC:MARK:SID:CARR UP.

To decrement the carrier frequency value to the next smallest point on the X-axis, send CALC:MARK:SID:CARR DOWN.

You can also set the value with an expression. Send CALC:MARK:SID:CARR (CALC:MARK:X?) to set the carrier frequency value to the current X-axis marker value.

Note When you shift the carrier frequency up or down, all the sideband markers shift up or down by the same amount.

The query returns the value of the carrier frequency (in X-axis units) currently used for sideband markers and calculations. The value is returned even if the sideband markers are not on.

To determine the X-axis units, send CALC:MARK:SID:CARR? UNIT.

CALCulate[1|2]:MARKer:SIDeband:COUNt command/query

Specifies the number of sideband markers for the display.

Command Syntax:

```
CALCulate[1|2]:MARKer:SIDeband:COUNt
<number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:200
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Calc2:Marker:Sid:Count 84"
OUTPUT 711;"CALC:MARK:SIDEBAND:COUN 104"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:SIDeband:COUNt?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The value can be specified numerically or with nonnumeric parameters.

To increase the number of sideband markers by one, send CALC:MARK:HARM:COUN UP.

To decrease the number of sideband markers by one, send CALC:MARK:HARM:COUN DOWN.

The query (CALC:MARK:HARM:COUNT?) returns the number of sideband markers currently specified for the display. The value is returned even if the sideband markers are not on.

CALCulate[1|2]:MARKer:SIDeband:INCRement command/query

Specifies the frequency increment (or delta) between sideband markers.

Command Syntax:

```
CALCulate[1|2]:MARKer:SIDeband:INCRement
(<number>[<unit>])|<step>|<bound>

     <number> ::= a real number (NRf data)
                     limits:  0.0:102400.0
       <unit> ::= [HZ|ORD]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calculate2:mark:sid:increment 6721.92"
OUTPUT 711;"Calc:Marker:Sid:Incr 93543.2"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:SIDeband:INCRement?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The value can be specified numerically or with nonnumeric parameters.

To increment the value to the next largest acceptable value, send CALC:MARK:SID:INCR UP.

To decrement the value to the next smallest acceptable value, send CALC:MARK:SID:INCR DOWN.

The query returns the current sideband increment value. To determine the X-axis units, send CALC:MARK:HARM:INCR? UNIT.

CALCulate[1|2]:MARKer[:STATe] command/query

Turns on the main markers or turns off all markers and marker functions for a selected trace.

Command Syntax:

```
CALCulate[1|2]:MARKer[:STATe] {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":CALCULATE2:MARK ON"
OUTPUT 711;"calculate:mark:stat ON"
```

Query Syntax:

```
CALCulate[1|2]:MARKer[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

To display the main marker and its annotation, send CALC:MARK ON. The analyzer displays the X-axis and Y-axis values at the top of the grid.

To disable the display of the main markers, the marker reference for the active trace, send CALC:MARK OFF.

CALCulate[1|2]:MARKer:X[:ABSolute] command/query

Specifies the main marker's X-axis position.

Command Syntax:

```
CALCulate[1|2]:MARKer:X[:ABSolute]
(<number>[<unit>])|<step>|<bound>

    <number> ::= a real number (NRf data)
                 limits:  -9.9e37:9.9e37
      <unit> ::= [HZ|S|ORD|COUNT|AVG|V|VPK|RPM|EU|REV]
      <step> ::= UP|DOWN
     <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Calculate:Mark:X:Abs -2.00133e+37"
OUTPUT 711;"CALC:MARKER:X 3.72749e+37"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:X[:ABSolute]?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the main marker's X-axis position. Send CALC:MARK:X? UNIT to determine the units for the X-axis.

To specify the main marker's X-axis position by display point number, use the CALC:MARK:POSition:POINt command.

You cannot use this command to set the independent axis position. To specify the main marker's independent axis position for Nyquist diagrams and orbits, send the CALC:MARK:POSition command. You can, however, use the query CALC:MARK:X? to determine the channel 1 amplitude value in orbits and the real portion of the amplitude in Nyquist diagrams.

Note You *cannot* move the main marker beyond the maximum displayed X-axis value nor below the minimum displayed X-axis value.

In **octave analysis instrument mode** (INST:SEL OCT), CALC:MARK:X MAX moves the main marker to the far right band.

CALCulate[1|2]:MARKer:X:RELative command/query

Specifies the marker reference X-axis position relative to the main marker.

Command Syntax:

```
CALCulate[1|2]:MARKer:X:RELative
(<number>[<unit>])|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -102400.0:102400.0
  <unit> ::= [HZ|S|ORD|COUNT|AVG|V|VPK|RPM|EU|REV]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":calculate:mark:x:relative 53565.1"
OUTPUT 711;"Calc2:Marker:X:Rel -19454.2"
```

Query Syntax: `CALCulate[1|2]:MARKer:X:RELative?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the marker reference X-axis position relative to the main marker's position. To query the units for the X-axis, send CALC:MARK:X:REL? UNIT.

To specify an absolute X-axis position for the marker reference, use the CALC:MARK:REF:X command.

CALCulate[1|2]:MARKer:Y[:ABSolute]? query

Returns the main marker's Y-axis position.

Query Syntax: `CALCulate[1|2]:MARKer:Y[:ABSolute]?`

Example Statements:

```
OUTPUT 711;"CALCULATE:MARK:Y:ABS?"
OUTPUT 711;"calc:marker:y?"
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query always returns the Y-axis position of the main marker, even if the marker is not currently displayed on the analyzer's screen. The returned value tells you the amplitude of the specified trace at the marker's X-axis position (specified with CALC:MARK:X or CALC:MARK:POS:POIN). In orbit displays, the returned value specifies the channel 2 amplitude. In Nyquist diagrams, the query returns the imaginary amplitude value.

Send CALC:MARK:Y? UNIT to determine the units for the Y-axis.

CALCulate[1|2]:MARKer:Y:RELative command/query

Specifies the marker reference Y-axis position relative to the main marker.

Command Syntax:

```
CALCulate[1|2]:MARKer:Y:RELative <number>|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  -150:150
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Calculate2:Mark:Y:Relative -112.425"
OUTPUT 711;"CALC:MARKER:Y:REL -4.60983"
```

Query Syntax:

```
CALCulate[1|2]:MARKer:Y:RELative?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the marker reference Y-axis position relative to the main marker's position.

To specify an absolute Y-axis position for the marker reference, use the CALC:MARK:REF:Y command.

CALCulate[1|2]:MATH:CONStant[1|2|3|4|5] command/query

Defines the value of one of the constant registers.

Command Syntax:

```
CALCulate[1|2]:MATH:CONStant[1|2|3|4|5] <real_part>
[,<imaginary_part>]

        <real_part> ::= <number>|<bound>
           <number> ::= a real number (NRf data)
                        limits:  -9.9e37:9.9e37
            <bound> ::= MAX|MIN
   <imaginary_part> ::= <number>|<bound>
           <number> ::= a real number (NRf data)
                        limits:  -9.9e37:9.9e37
            <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calculate:math:constant -5.49524e+37,
7.3878e+37"
OUTPUT 711;"Calc2:Math:Constant2 5.70674e+36, -9.872e+37"
```

Query Syntax: `CALCulate[1|2]:MATH:CONStant[1|2|3|4|5]?`

Return Format: `NR3, NR3`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

The analyzer assumes the first parameter is the real part of the constant. If the second parameter is used, an imaginary part is specified.

To use a constant in a math function, you must first load it into one of the analyzer's five constant registers, 1 through 5. You can include the constant register's name (K1|K2|K3|K4|K5) at the appropriate place in your function. Functions are defined with the CALC:MATH:EXPR command.

To display a math constant as a trace, create a math function with the CALC:MATH:EXPR command. For example, CALC:MATH:EXPR1 K1 loads the value of the math constant register, K1, into the math function register, F1. Then use the command CALC1:MATH:SEL F1;STATe ON to display the math constant (K1) in trace box A.

Note This command ignores the trace specifier.

CALCulate[1|2]:MATH:DATA command/query

Loads a complete set of math definitions.

Command Syntax: `CALCulate[1|2]:MATH:DATA <MATH>`

```
            <MATH> ::- <file_type><function_1><function_2><function_3>
                       <funct ion_4><function_5><constant_1><constant_2>
                       <constant_3><constant_4><constant_5>
       <file_type> ::- 1503  specifies math table
      <function_1> ::- 270 bytes specifying function expression,
                       terminating with null character
      <function_2> ::- 270 bytes specifying function expression,
                       terminating with null character
      <function_3> ::- 270 bytes specifying function expression,
                       terminating with null character
      <function_4> ::- 270 bytes specifying function expression,
                       terminating with null character
      <function_5> ::- 270 bytes specifying function expression,
                       terminating with null character
```

270 bytes must be sent when specifying a function expression. All characters following the first null character are discarded.

```
      <constant_1> ::- <real_part_constant><imaginary_part_constant>
      <constant_2> ::- <real_part_constant><imaginary_part_constant>
      <constant_3> ::- <real_part_constant><imaginary_part_constant>
      <constant_4> ::- <real_part_constant><imaginary_part_constant>
      <constant_5> ::- <real_part_constant><imaginary_part_constant>
<real_part_constant> ::- 8 byte floating point number
                       (REAL variable in HP Instrument BASIC)
        <imaginary
    _part_constant> ::- 8 byte floating point number
                       (REAL variable in HP Instrument BASIC)
```

Example Statements:

```
OUTPUT 711;":CALC:MATH:DATA MATH"
OUTPUT 711;"calc:math:data MATH"
```

Query Syntax: `CALCulate[1|2]:MATH:DATA?`

Return Format: `definite length <MATH>`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command allows you to transfer a complete set of math definitions— the same information contained in a math file—between the analyzer and your controller.

When you transfer a set of math definitions to the analyzer, you can use either the definite or indefinite length block syntax. When the analyzer returns the set of math definitions, it always uses the definite length block syntax. See "Block Data" in chapter 4 for more information.

The MMEM:STOR:MATH command and the MMEM:LOAD:MATH command also transfer a complete set of math definitions using one of the analyzer's mass storage devices. See these commands for more information about loading and storing data in the function registers and in the constant registers.

The example program, "LOADMATH," in chapter 6 uses this command to download a complete set of math definitions. The example program, "GETMATH," in chapter 6 uses the query form of this command to upload a complete set of math definitions.

CALCulate[1|2]:MATH[:EXPRession[1|2|3|4|5]] command/query

Defines a math function.

Command Syntax:

```
CALCulate[1|2]:MATH[:EXPRession[1|2|3|4|5]] <EXPR>

          <EXPR> ::= ([<expr_element>]...)
  <expr_element> ::= see operations and operands listed below.
```

Example Statements:

```
OUTPUT 711;"Calculate2:Math:Expr (K1*FRES)"
OUTPUT 711;"CALCULATE2:MATH (TIME1-TIME2)"
```

Query Syntax:

```
CALCulate[1|2]:MATH[:EXPRession[1|2|3|4|5]]?
```

Return Format:

```
STRING
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This command loads an expression into one of five function registers.

Before you can display the results of a trace math function, you must load the function definition into one of the analyzer's five function registers: F1 through F5. Once you have loaded the function register with CALC:MATH:EXPR, you execute the expression and display the results with the CALC:MATH:SEL command. CALC:MATH:STAT must be ON.

Note This command is *not* trace specific. It ignores the trace specifier.

To define trace math functions, combine the elements (listed below) according to the rules of standard algebraic notation. Use parentheses to control the order of operations.

- Operations
  - AWEIGHT Apply A-weight filter
  - BWEIGHT Apply B-weight filter
  - CWEIGHT Apply C-weight filter
  - CONJ Complex Conjugate
  - DIFF Differentiate
  - DJOM Divide by j θ
  - EXP Exponential
  - FFT Fast Fourier Transform
  - IFFT Inverse Fast Fourier Transform

- INTEG Integrate
- IMAG Imaginary Part
- LN Natural Logarithm
- MAG Magnitude
- PSD Power Spectral Density
- REAL Real Part
- SQRT Square Root
- XJOM multiply by j θ
- + Add
- - Subtract
- * Multiply
- / Divide

- Operands
  - D1|D2|D3|D4|D5|D6|D7|D8 Contents of data registers
  - F1|F2|F3|F4|F5 Contents of function registers
  - K1|K2|K3|K4|K5 Contents of constant registers
  - Measurement Data (depends on instrument mode)
    [1|2] specifies which trace contains the measurement data.
    - ACORR[1|2] Autocorrelation (INST:SEL CORR only)
    - CDF[1|2] Cumulative Density Function (INST:SEL HIST only)
    - COH Coherence
    - CPOW[1|2] Composite Power (INST:SEL ORD only; Option 1D0)
    - CSPEC Cross Spectrum
    - FRES Frequency Response
    - HIST[1|2] Histogram (INST:SEL HIST only)
    - LSPEC[1|2] Linear Spectrum
    - NVAR[1|2] Normalized Variance (INST:SEL SIN only; Option 1D2)
    - PDF[1|2] Probability Density Function (INST:SEL HIST only)
    - PSPEC[1|2] Power Spectrum
    - RPM RPM Profile (INST:SEL ORD only; Option 1D0)
    - TIME[1|2] Time Data
    - TIME[1|2] Resampled Time (INST:SEL ORD only; Option 1D0)
    - TIME[1|2] Unfiltered Time (INST:SEL HIST only)
    - TRACK[1|2|3|4|5][1|2] Order Track (INST:SEL ORD only; Option 1D0)
      [1|2|3|4|5] specifies which order track
    - WTIME[1|2] Windowed Time Data
    - XCORR Cross Correlation (INST:SEL CORR only)

Refer to online help or the *HP 35665A Concepts Guide* for more information on math operations.

CALCulate[1|2]:MATH:SELect command/query

Selects and displays the designated math function, if CALC:MATH:STAT ON.

Command Syntax:

```
CALCulate[1|2]:MATH:SELect {F1|F2|F3|F4|F5}
```

Example Statements:

```
OUTPUT 711;":calculate:math:sel F5"
OUTPUT 711;"Calculate:Math:Select F1"
```

Query Syntax:

```
CALCulate[1|2]:MATH:SELect?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  F1
SCPI Compliance:  instrument-specific
```

Description:

The results are displayed in the specified trace box.

Load the math function into the specified function register with the CALC:MATH:EXPR command. An error is generated if an expression contains operands not available in the selected instrument mode.

To display a math constant as a trace, create a math function with the CALC:MATH:EXPR command. For example, CALC:MATH:EXPR1 K1 loads the value of the math constant register, K1, into the math function register, F1. Then use this command (CALC1:MATH:SEL F1;STATe ON) to display the math constant (K1) in trace box A.

Note CALC[1|2]:MATH:STATe must be ON for the specified trace box.

CALCulate[1|2]:MATH:STATe command/query

Evaluates the currently selected math operation for the specified trace and displays the results.

Command Syntax: `CALCulate[1|2]:MATH:STATe (OFF|0|ON|1)`

Example Statements:

```
OUTPUT 711;"CALC2:MATH:STATE ON"
OUTPUT 711;"calc:math:stat OFF"
```

Query Syntax: `CALCulate[1|2]:MATH:STATe?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

CALC:MATH:STATE must be ON to perform math operations.

To define a math function, use CALC:MATH:EXPR or CALC:MATH:DATA. (To define the value of the constant registers, use the CALC:MATH:CONS command.) Select the function with the CALC:MATH:SEL command. Execute the function and display the results with the CALC:MATH:STAT ON command.

CALC:MATH:STATE OFF turns off math operations. You can *not* execute or display a math operation unless CALC:MATH:STATE is ON.

Note

Each trace box *always* has a selected math function. When the analyzer receives the CALC[1|2]:MATH:STAT ON command, it evaluates the selected function and displays the results in the specified trace box. If a trace box is not specified, the function defaults to trace box A. When the analyzer receives the CALC[1|2]:MATH:STAT off, it stops evaluating the function. The analyzer returns the data it was displaying in the trace box before receiving the CALC:MATH:STAT ON command.

CALCulate[1|2]:SYNThesis:COPY command

Copies the contents of the curve fit table into the synthesis table.

Command Syntax:

```
CALCulate[1|2]:SYNThesis:COPY (CFIT)
```

Example Statements:

```
OUTPUT 711;":Calc:Synthesis:Copy CFIT"
OUTPUT 711;"CALCULATE2:SYNT:COPY CFIT"
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command overwrites the synthesis table with the contents of the curve fit table. You cannot recover the contents of the previous synthesis table after sending this command.

The analyzer does not copy “engineering units.” Use the CALC:SYNT:GAIN command to simulate engineering units with synthesis.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:SYNThesis:DATA command/query

Loads values into the synthesis table.

Command Syntax:

```
CALCulate[1|2]:SYNThesis:DATA <SYNFIT>

        <SYNFIT> ::= see Description
```

Example Statements:

```
OUTPUT 711;"calculate:synt:data SYNFIT"
OUTPUT 711;"Calc2:Synt:Data SYNFIT"
```

Query Syntax:

```
CALCulate[1|2]:SYNThesis:DATA?
```

Return Format:

```
definite length <SYNFIT>
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command transfers a complete synthesis table from your controller to the analyzer.

When you transfer a synthesis table to the analyzer, you must use the definite length block syntax. Data must be 64-bit binary floating-point numbers (see the FORMat[:DATA] REAL command). The elements of the definite length block for a synthesis table are defined below. See "Block Data" in chapter 4 for more information about transferring block data.

Note This command is not trace specific. It ignores the trace specifier.

```
        <SYNFIT> ::= <Point1><Point2> . . . <Point175>
        <Point1> ::= Table type
                     <0> = pole zero
                     <1> = pole residue
                     <2> = polynomial
        <Point2> ::= number_of_lines_in_left_column
        <Point3> ::= number_of_lines_in_right_column
        <Point4> ::= number_of_lines_in_Laurent_column
        <Point5> ::= real_part_first_term_in_left_column
        <Point6> ::= imaginary_part_first_term_in_left_column
        <Point7> ::= real_part_second_term_in_left_column
        <Point8> ::= imaginary_part_second_term_in_left_column
            •
            •
            •
       <Point47> ::= real_part_first_term_in_right_column
       <Point48> ::= imaginary_part_first_term_in_right_column
            •
            •
            •
       <Point89> ::= real_part_first_term_in_Laurent_column
       <Point90> ::= imaginary_part_first_term_in_Laurent_column
            •
            •
            •
      <Point131> ::= first_curve_fit_term_left_column
      <Point132> ::= second_curve_fit_term_left_column
<curve_fit_term> ::=  0
            •
            •
            •
      <Point152> ::= first_curve_fit_term_right_column
      <Point153> ::= second_curve_fit_term_right_column
<curve_fit_term> ::=  0
            •
            •
            •
      <Point173> ::= gain
      <Point174> ::= frequency_scale
     <Point 175> ::= time_delay
```

Note

If a curve fit term = 1, "fxd" (fixed) appears by the curve fit term in the table. It has no effect on synthesis.

CALCulate[1|2]:SYNThesis:DESTination command/query

Selects the data register for the results of the synthesis operation.

Command Syntax:

```
CALCulate[1|2]:SYNThesis:DESTination
{D1|D2|D3|D4|D5|D6|D7|D8}
```

Example Statements:

```
OUTPUT 711;":CALC:SYNTHESIS:DEST D8"
OUTPUT 711;"calc:synthesis:dest D7"
```

Query Syntax:

```
CALCulate[1|2]:SYNThesis:DESTination?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  D8
SCPI Compliance:  instrument-specific
```

Description:

This command specifies which data register holds the synthesis of the intermediate and final synthesis models. The default register is D8.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:SYNThesis:FSCale command/query

Specifies a frequency scale for the synthesis operation.

Command Syntax:

```
CALCulate[1|2]:SYNThesis:FSCale <number>|<step>|<bound>

     <number> ::= a real number (NRf data)
                    limits:  1e-6:1e6
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Calculate2:Synt:Fsc -1.5"
OUTPUT 711;"CALCULATE:SYNT:FSCALE .14321"
```

Query Syntax:

```
CALCulate[1|2]:SYNThesis:FSCale?
```

Return Format: NR3

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +1.0
SCPI Compliance:  instrument-specific
```

Description:

This command scales the synthesis model along the X-axis by f/frequency scale, where f is frequency in Hz. The frequency scale must be a positive value.

The value can be used to scale poles and zeros from Hz to radians-per-second by setting the scaling value to 1 / (2 π).

Poles and zero terms are not multiplied by 1/frequency scale.

Note

This command must be sent before CALC:SYNT. This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:SYNThesis:GAIN command/query

Specifies the gain constant, K, for a synthesis operation.

Command Syntax:

```
CALCulate[1|2]:SYNThesis:GAIN <number>|<step>|<bound>

        <number> ::= a real number (NRf data)
                     limits:  -9.9e37:9.9e37  (excluding 0.0)
          <step> ::= UP|DOWN
         <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":calc:synt:gain .1"
OUTPUT 711;"Calc2:Synthesis:Gain -0.5"
```

Query Syntax: `CALCulate[1|2]:SYNThesis:GAIN?`

Return Format: NR3

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +1.0
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the desired gain of a synthesized frequency response function. The limits exclude 0.0.

The gain constant, K, is unitless.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:SYNThesis[:IMMediate] command

Creates a frequency response curve from the synthesis table.

Command Syntax:

```
CALCulate[1|2]:SYNThesis[:IMMediate]
```

Example Statements:

```
OUTPUT 711;"CALC2:SYNTHESIS:IMM"
OUTPUT 711;"calculate:synt"
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command creates a frequency response curve based on the current synthesis table.

Values for the table are entered with the CALC:SYNT:DATA command. The results of the synthesis operation are stored in the synthesis data register specified by the CALC:SYNT:DEST command.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:SYNThesis:SPACing command/query

Specifies a linear or logarithmic scale for the X-axis data spacing.

Command Syntax: `CALCulate[1|2]:SYNThesis:SPACing (LINear|LOGarithmic)`

Example Statements:

```
OUTPUT 711;":Calc:Synthesis:Spac LINear"
OUTPUT 711;"CALCULATE2:SYNT:SPAC LINear"
```

Query Syntax: `CALCulate[1|2]:SYNThesis:SPACing?`

Return Format: CHAR

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  LIN
SCPI Compliance:  instrument-specific
```

Description:

To specify a linear X-axis scale, send CALC:SYNT:SPAC LIN.

To specify a logarithmic scale, send CALC:SYNT:SPAC LOG.

This command should not be confused with the DISPlay:TRACe:X:SPACing command which changes the X-axis display grid between linear and logarithmic spacing.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:SYNThesis:TDELay **command/query**

Specifies a time delay value for the synthesis operation.

Command Syntax:

```
CALCulate[1|2]:SYNThesis:TDELay
(<number>[<unit>])|<step>|<bound>
```

```
<number> ::= a real number (NRf data)
             limits:  -100:100
  <unit> ::= [S]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calculate:synt:tdelay -65.2024"
OUTPUT 711;"Calc2:Synt:Tdelay -99.6365"
```

Query Syntax: `CALCulate[1|2]:SYNThesis:TDELay?`

Return Format: NR3

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

This command allows a time delay, that is a phase ramp, to be included in the synthesized response.

A positive delay produces a negative phase ramp.

Note This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:SYNThesis:TTYPe command/query

Converts the synthesis table to another table format.

Command Syntax:

```
CALCulate[1|2]:SYNThesis:TTYPe
{PZERo|PFRaction|POLYnomial}
```

Example Statements:

```
OUTPUT 711;":CALC:SYNTHESIS:TTYP PZERo"
OUTPUT 711;"calc:synthesis:ttyp PZERo"
```

Query Syntax:

```
CALCulate[1|2]:SYNThesis:TTYPe?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  1D3 Curve Fit/Synthesis
Overlapped:  no
Preset State:  PZER
SCPI Compliance:  instrument-specific
```

Description:

To convert the synthesis table to pole-zero format, send CALC:SYNT:TTYP PZER.

To convert the synthesis table to partial-fraction format, send CALC:SYNT:TTYP PFR. This format is identified as pole-residue in the table.

To convert the synthesis table to polynomial format, send CALC:SYNT:TTYP POLY.

The analyzer ignores this command if the table already exists in the specified format.

Note

Table conversions between formats are not allowed if the table data represents a non-Hermitian symmetric system. Hermitian symmetry is most easily defined in the polynomial table format: all numerator and denominator coefficients must be real.

This command is not trace specific. It ignores the trace specifier.

CALCulate[1|2]:UNIT:AMPLitude command/query

Selects the unit of amplitude for the Y-axis scale.

Command Syntax:

```
CALCulate[1|2]:UNIT:AMPLitude {PEAK|RMS}
```

Example Statements:

```
OUTPUT 711;"Calculate2:Unit:Ampl PEAK"
OUTPUT 711;"CALCULATE2:UNIT:AMPLITUDE RMS"
```

Query Syntax:

```
CALCulate[1|2]:UNIT:AMPLitude?
```

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  RMS (Channel 1)
PEAK (Channel 2)
SCPI Compliance:  instrument-specific
```

Description:

To display peak amplitude, send CALC:UNIT:AMPL PEAK.

To display RMS amplitude, send CALC:UNIT:AMPL RMS.

The default value is dependent upon the selected measurement data (CALC:FEED). Table 11-2 indicates valid unit selections for the CALC:UNIT:AMPL command. If measurement data does not appear in the table, you are *not* permitted to select the amplitude. In this case, a query returns a null string. See "Determining Units" in appendix E for information about available Y-axis units.

Table 11-2. Valid Unit Selections for CALC:UNIT:AMPL

Measurement Data CALC:FEED command	CALC:UNIT:AMPL	
	RMS	PEAK
Auto correlation CALC:FEED 'XTIM:CORR'		X
Capture buffer CALC:FEED 'TCAP'		X
Coherence CALC:FEED 'XFR:POW:COH'		
Composite Power CALC:FEED 'XFR:POW:COMP'	X	X
Cross Correlation CALC:FEED 'XTIM:CORR:CROS'		X
Cross Spectrum CALC:FEED 'XFR:POW:CROS'	X	X
Cumulative Density Function CALC:FEED 'XTIM:VOLT:CDF'		
Frequence Response CALC:FEED 'XFR:POW:RAT'		
Histogram CALC:FEED 'XTIM:VOLT:HIST'		
Linear Spectrum CALC:FEED 'XFR:POW:LIN'	X	X
Order Track CALC:FEED 'XORD:TRACK'	X	X
Power Spectrum CALC:FEED 'XFR:POW'	X	X
Probability Density Function CALC:FEED 'XTIM:VOLT:PDF'		
RPM Profile CALC:FEED 'XRPM:PROF'		
Time CALC:FEED 'XTIM:VOLT'		X
Windowed Time CALC:FEED 'XTIM:VOLT:WIND'		X

Units for data registers (D[1|2... |8]) and waterfall registers (W[1|2| ... |8]) are dependent upon the type of measurement data stored in the register. See the appropriate measurement data row for valid unit selections.

CALCulate[1|2]:UNIT:ANGLe command/query

Specifies the unit for phase coordinates.

Command Syntax:

```
CALCulate[1|2]:UNIT:ANGLe {DEGRee|RADian}
```

Example Statements:

```
OUTPUT 711;":calc:unit:angle DEGRee"
OUTPUT 711;"Calc2:Unit:Angl RADian"
```

Query Syntax:

```
CALCulate[1|2]:UNIT:ANGLe?
```

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  DEGR
SCPI Compliance:  instrument-specific
```

Description:

This command is only valid when phase trace coordinates are specified (CALC:FORM PHAS or CALC:FORM UPH).

To select phase units in degrees for the specified trace, send CALC:UNIT:ANGL DEGR. To select phase units in radians for the specified trace, send CALC:UNIT:ANGL RAD.

CALCulate[1|2]:UNIT:VOLTage command/query

Selects the vertical unit for the specified display's Y-axis.

Command Syntax:

```
CALCulate[1|2]:UNIT:VOLTage <STRING>
```

Example Statements:

```
OUTPUT 711;"CALC2:UNIT:VOLT 'V'"
OUTPUT 711;"calculate:unit:volt 'V2/HZ'"
```

Query Syntax:

```
CALCulate[1|2]:UNIT:VOLTage?
```

Return Format:

```
STRING
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  V2
SCPI Compliance:  instrument-specific
```

Description:

With some measurements, you can select the unit for the Y-axis scale.

- To select volts, send CALC:UNIT:VOLT 'V'.
- To select volts^2, send CALC:UNIT:VOLT 'V2'.
- To select square root power spectral density, send CALC:UNIT:VOLT 'V/RTHZ'.
- To select power spectral density, send CALC:UNIT:VOLT 'V2/HZ'.
- To select energy spectral density, send CALC:UNIT:VOLT 'V2S/HZ'.

Depending upon the measurement data selection and specified trace coordinates, the selection of the base unit may be restricted. In addition, the analyzer does not permit specification of the vertical unit for some types of measurement data.

Table 11-3 indicates valid unit selections for the CALC:UNIT:VOLT command. If measurement data does not appear in the table, you are *not* permitted to select the base unit. In this case, a query returns a null string.

Table 11-3. Valid Unit Selections for CALC:UNIT:VOLT

Measumrent Data **CALC:FEED command (INST:SEL command)**	**CALC:UNIT:VOLT**				
	V	V2	V/RTHZ	V2/HZ	V2S/HZ
Composite Power CALC:FEED 'XFR:POW:COMP' (INST:SEL ORD)	X	X			
Linear Spectrum CALC:FEED 'XFR:POW:LIN' (INST:SEL FFT)	X	X	X	X	X
Linear Spectrum CALC:FEED 'XFR:POW:LIN' (INST:SEL SINE)	X	X			
Order Track CALC:FEED 'XORD:TRACK' (INST:SEL ORD)	X	X			
Power Spectrum CALC:FEED 'XFR:POW' (INST:SEL FFT)	X	X	X	X	X
Power Spectrum CALC:FEED 'XFR:POW' (INST:SEL OCT)	X	X	X	X	X
Power Spectrum CALC:FEED 'XFR:POW' (INST:SEL ORD)	X	X			

Units for data registers (D[1|2... |8]) and waterfall registers (W[1|2| ... |8]) are dependent upon the type of measurement data stored in the register. See the appropriate measurement data row for valid unit selections.

See “Determining Units” in appendix E for information about available Y-axis units.

CALCulate[1|2]:WATerfall:COUNt command/query

Specifies the number of traces stored for a waterfall display.

Command Syntax:

```
CALCulate[1|2]:WATerfall:COUNt <number>[|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  1:50000
       <unit> ::=
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Calculate:Wat:Count 50"
OUTPUT 711;"CALC2:WAT:COUNT 32"
```

Query Syntax: `CALCulate[1|2]:WATerfall:COUNt?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +15
SCPI Compliance:  instrument-specific
```

Description:

This command determines the total capacity of the waterfall.

When you change instrument modes or start a new measurement with the ABOR;:INIT command, all current waterfall traces are lost.

Note The maximum number of traces stored for a waterfall display is dependent upon the amount of available memory and the size of each trace in the waterfall.

CALCulate[1|2]:WATerfall[:DATA]? query

Returns waterfall data that has been transformed to the currently selected coordinate transform (specified with CALC:FORMat).

Query Syntax:

```
CALCulate[1|2]:WATerfall[:DATA]?
```

Example Statements:

```
OUTPUT 711;"calc2:waterfall:data?"
OUTPUT 711;"Calc2:Waterfall?"
```

Return Format: <BLOCK>

If FORMat[:DATA] REAL:

```
          <BLOCK> ::= #<byte>[<length_bytes>] <1st_waterfall_value>
                      . . . <last_waterfall_value>
           <byte> ::= one byte specifying the number of length bytes
                      to follow (ASCII encoded)
   <length_bytes> ::= number of data bytes to follow (ASCII encoded)
```

If FORMat[:DATA] ASCii:

```
          <BLOCK> ::= <1st_waterfall_value> . . .<last_waterfall_value>
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns a definite length block of coordinate-transformed waterfall data.

The block is returned as a series of Y-axis values. Use the TRACe:X[:DATA]? query to determine the X-axis values for the waterfall data. Use the TRACe:Z[:DATA]? query to determine the Z-axis values for the waterfall data.

This query has no command form. You cannot return waterfall data to the display with CALC:WAT:DATA. To send data that has *not* been transformed, use the TRAC:WAT[:DATA] command. See the introduction to this chapter for more information about the differences between these commands.

CALCulate[1|2]:WATerfall:SLICe:COPY   command

Copies the selected waterfall slice to the designated data register.

Command Syntax:

```
CALCulate[1|2]:WATerfall:SLICe:COPY
{D1|D2|D3|D4|D5|D6|D7|D8}
```

Example Statements:

```
OUTPUT 711;":CALC2:WATERFALL:SLIC:COPY D4"
OUTPUT 711;"calculate:wat:slice:copy D2"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

A slice is a vertical line through the collection of waterfall traces at the same X- axis value. This command copies the slice to the specified data register. Use the CALC:WAT:SLICe:SELect command to select the slice.

CALCulate[1|2]:WATerfall:SLICe:SELect **command/query**

Selects the waterfall slice at the specified X-axis position.

Command Syntax:

```
CALCulate[1|2]:WATerfall:SLICe:SELect
(<number>[<unit>])|<step>|<bound>

    <number> ::= a real number (NRf data)
                 limits:  -16382:1e6
      <unit> ::= [HZ|S|ORD|COUNT|AVG|V|VPK|RPM|EU|REV]
      <step> ::= UP|DOWN
     <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Calc:Waterfall:Slic:Select 206725"
OUTPUT 711;"CALC2:WAT:SLICE:SEL 914830"
```

Query Syntax: `CALCulate[1|2]:WATerfall:SLICe:SELect?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the X-axis position where the waterfall slice is to be made. The slice can be copied to a data register with the CALC:WAT:SLIC:COPY command.

CALCulate[1|2]:WATerfall:SLICe:SELect:POINt *command/query*

Selects a waterfall slice by its display point value.

Command Syntax:

```
CALCulate[1|2]:WATerfall:SLICe:SELect:POINt
[<number>|<step>|<bound>]

<number> ::= a real number (NRf data)
             limits:  0:2048
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":calculate:wat:slic:select:poin 717"
OUTPUT 711;"Calculate:Wat:Slic:Select:Poin 394"
```

Query Syntax:

```
CALCulate[1|2]:WATerfall:SLICe:SELect:POINt?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the X-axis position for the waterfall slice by point number.

A waterfall composed of frequency data displays 401 points along the X-axis. A waterfall composed of time data displays 1024 points along the X-axis.

In correlation analysis there are 512 points.

In histogram analysis, the number of points is determined by the number of bins (set with the HIST:BINS command). The maximum number is 1024 points.

In octave analysis, the number of points is determined by the bandwidth of the filters. There are 11 points for full octave, 33 points for 1/3 octave and 132 points for 1/12 octave.

In some cases the number of points is arbitrary. These include waterfall displays from order tracking or from the arbitrary source.

See the CALC:WAT:SLICe:COPY command for information about saving a waterfall slice to a data register.

CALCulate[1|2]:WATerfall:TRACe:COPY command

Saves the selected trace to the specified data register.

Command Syntax:

```
CALCulate[1|2]:WATerfall:TRACe:COPY
{D1|D2|D3|D4|D5|D6|D7|D8}
```

Example Statements:

```
OUTPUT 711;"CALCULATE:WAT:TRAC:COPY D2"
OUTPUT 711;"calc2:waterfall:trac:copy D6"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command copies a trace, selected with the CALC:WAT:TRACe:SELect command, from the waterfall to the specified data register.

See the CALC:WAT:SLICe commands for information about selecting and saving waterfall slices.

CALCulate[1|2]:WATerfall:TRACe:SELect command/query

Selects a waterfall trace by its Z-axis value.

Command Syntax:

```
CALCulate[1|2]:WATerfall:TRACe:SELect
{<number>[<unit>]}|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  -9.9e37:9.9e37
       <unit> ::= [S|RPM|COUNT|AVG]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Calculate:Wat:Trace:Sel -8.83941e+37"
OUTPUT 711;"CALC:WATERFALL:TRAC:SELECT 5.61155e+37"
```

Query Syntax:

```
CALCulate[1|2]:WATerfall:TRACe:SELect?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

This command selects a waterfall trace by its Z-axis value. The Z-axis value tells you when the measurement data was armed.

See the CALC:WAT:TRAC:COPY command for information about copying a selected trace to a data register.

CALCulate[1|2]:WATerfall:TRACe:SELect:POINt command/query

Selects a waterfall trace by its step value.

Command Syntax:

```
CALCulate[1|2]:WATerfall:TRACe:SELect:POINt <number>|<step>|<bound>

     <number> ::= a real number (NRf data)
                    limits:  0:50000
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"calc:wat:trace:sel:point 365"
OUTPUT 711;"Calc2:Wat:Trace:Sel:Point 4500"
```

Query Syntax:

```
CALCulate[1|2]:WATerfall:TRACe:SELect:POINt?
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command selects a waterfall trace by its step value. The total number of waterfall steps is specified with the CALC:WAT:COUNt command.

A value of 1 specifies the first trace collected for the waterfall display. A value of 2 specifies the second trace collected for the waterfall display.

See the CALC:WAT:TRAC:COPY command for information about copying a selected trace to a data register.

12

CALibration

This subsystem contains commands related to calibration of the analyzer.

CALibration[:ALL]? query

Calibrates the analyzer and returns the result.

Query Syntax: `CALibration[:ALL]?`

Example Statements:

```
OUTPUT 711;":CAL?"
OUTPUT 711;"cal:all?"
```

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The analyzer performs a full calibration when you send this query. If the calibration completes without error, the analyzer returns 0. If the calibration fails, the analyzer returns 1.

This query is the same as the *CAL? query.

CALibration:AUTO **command/query**

Calibrates the analyzer or sets the state of the autocalibration function.

Command Syntax: `CALibration:AUTO (OFF|0|ON|1|ONCE)`

Example Statements:

```
OUTPUT 711;"Cal:Auto OFF"
OUTPUT 711;"CAL:AUTO ONCE"
```

Query Syntax: `CALibration:AUTO?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

Send CAL:AUTO ON to enable the analyzer's autocalibration function, OFF to disable it. This function calibrates the analyzer several times during the first hour of operation and once per 140 minutes thereafter.

Send CAL:AUTO ONCE to initiate a single calibration.

Note CAL:AUTO is set to 0 (OFF) after *RST.

13

DISPlay

This subsystem contains commands that control the analyzer's presentation of data on its front-panel display.

The DISPlay subsystem contains commands grouped under the WINDow mnemonic. The WINDow mnemonic contains an optional trace specifier: [1|2]. To direct a command to trace A, omit the specifier or use 1. To direct a command to trace B, use 2.

WINDow is an implied mnemonic. Therefore, you can omit it from all DISPlay commands. However, if you wish to direct a DISPlay command to a specific trace, you must use the WINDow trace specifier. See "Implied Mnemonics" in chapter 3 for more information.

Note Directing a command with a trace specifier makes that display the active trace box.

DISPlay:ANNotation command/query

Turns the display of screen annotation on or off.

Command Syntax:

```
DISPlay:ANNotation {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":display:ann ON"
OUTPUT 711;"Display:Ann OFF"
```

Query Syntax:

```
DISPlay:ANNotation?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

When DISP:ANN is OFF, following information is *not* displayed on the analyzer's screen:

- X-axis annotation
- Y-axis annotation
- Z-axis annotation
- Marker annotation
- Mini-state

When DISP:ANN is OFF, this information is not printed or plotted. It is available, however, HP-IB queries.

When DISP:ANN is ON, all information, including the annotation, is displayed on the analyzer's screen.

DISPlay:BODE command

Selects a Bode diagram format.

Command Syntax:

```
DISPlay:BODE
```

Example Statements:

```
OUTPUT 711;"DISP:BODE"
OUTPUT 711;"disp:bode"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The Bode diagram formats the display as follows:

- Frequency response of the measurement data (traces A and B).
- Trace coordinate for trace A is dB magnitude.
- Trace coordinate for trace B is phase.
- The X-axis scale is logarithmic.
- Markers are coupled.

Note

To change the X-axis to linear scaling, send the DISPlay:WINDow[1|2]:TRAC:X:SPAC LIN command.

DISPlay:CONTents command/query

Specifies what is displayed on the analyzer's screen.

Command Syntax:

```
DISPlay:CONTents
(TRACe|MSTate|MMEMory|STABle|CTABle|FTABle|TTABle|MEMory
|OPTion|CAPTure|IState)
```

Example Statements:

```
OUTPUT 711;":Disp:Cont CAPTure"
OUTPUT 711;"DISPLAY:CONT MEMory"
```

Query Syntax: `DISPlay:CONTents?`

Return Format: `CHAR`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  TRAC
SCPI Compliance:  confirmed
```

Description:

This command specifies the contents of the analyzer's display area.

- To display measurement data, send TRAC. Use the CALC:FEED commands to specify the measurement data to be displayed in the active trace.
- To display the analyzer's current configuration, send MST.
- To display the contents of the default disk, send MMEM. (You select the default disk with the MMEM:MSIS command.)
- To display the synthesis table, send STAB.
- To display the curve fit table, send CTAB.
- To display the fault log table, send FTAB.
- To display the test log table, send TTAB.
- To display the memory usage table, send MEM.
- To display the option configuration of the analyzer, send OPT.
- To display header information for the time capture buffer, send CAP.
- To display the analyzer's current input configuration, send IST.

DISPlay:ENABle command/query

Enables the analyzer's display.

Command Syntax: `DISPlay:ENABle (OFF|0|ON|1)`

Example Statements:

```
OUTPUT 711;"display:enab ON"
OUTPUT 711;"Disp:Enable OFF"
```

Query Syntax: `DISPlay:ENABle?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

To disable the analyzer's display, send DISP:ENAB OFF. All information appearing on the analyzer's screen (except the softkey menus), send is blanked out. The message, "Display Blanking On" appears on the analyzer's screen.

To enable the display or to turn display blanking off, send DISP:ENAB ON.

Note

To turn on a blanked screen from the front panel, you must preset the analyzer. To turn on a blanked screen *without* presetting the analyzer send the DISP:ENAB ON command over the HP-IB.

DISPlay:FORMat **command/query**

Selects a format for displaying trace data.

Command Syntax:

```
DISPlay:FORMat {SINGle|ULOWer|FBACk|WATerfall}
```

Example Statements:

```
OUTPUT 711;":DISP:FORMAT ULOWer"
OUTPUT 711;"disp:form WATerfall"
```

Query Syntax:

```
DISPlay:FORMat?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  SING
SCPI Compliance: confirmed
```

Description:

When you select SING, the analyzer uses the entire screen for the active trace box (CALC:ACT ON).

When you send ULOW, the analyzer uses the upper half of the screen for trace box A and the lower half for trace box B.

When you select FBAC, the analyzer uses the entire screen, but overlays the two trace boxes in the area.

When you select WAT, the analyzer uses the upper quarter of the screen for trace box A and the lower trace box for a waterfall display of trace B. Trace A and trace B are independent in a waterfall display. You can assign any available measurement data to either trace with the CALC[1|2]:FEED command. See the DISPlay:WATerfall commands for more information about the waterfall displays.

Note The WAT format is not allowed in swept sine instrument mode.

DISPlay:GPIB:ECHO command/query

Enables and disables the echoing of HP-IB command mnemonics to the analyzer's screen.

Command Syntax: `DISPlay:GPIB:ECHO (OFF|0|ON|1)`

Example Statements:

```
OUTPUT 711;"Display:Gpib:Echo ON"
OUTPUT 711;"DISP:GPIB:ECHO ON"
```

Query Syntax: `DISPlay:GPIB:ECHO?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

When echoing is enabled, the analyzer displays the HP-IB command mnemonic which corresponds to the operation executed from the front panel. The command mnemonic appears on the third line in the upper-left corner of the screen.

Not every keystroke generates an HP-IB command.

DISPlay:PROGram[:MODE] command/query

Selects the portion of the analyzer's screen to be used for HP Instrument BASIC program output.

Command Syntax:

```
DISPlay:PROGram[:MODE] {OFF|0|FULL|UPPer|LOWer}
```

Example Statements:

```
OUTPUT 711;":disp:program UPPer"
OUTPUT 711;"Disp:Prog:Mode UPPer"
```

Query Syntax:

```
DISPlay:PROGram[:MODE]?
```

Return Format: CHAR

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

FULL allocates the entire trace box for program output. UPP allocates the upper trace box. LOW allocates the lower trace box.

If DISP:PROG is OFF, the analyzer does *not* allocate any portion of the trace box for program output.

DISPlay:WATerfall:BASeline command/query

Specifies the percentage of each trace that is concealed in the waterfall display.

Command Syntax:

```
DISPlay:WATerfall:BASeline
(<number><unit>)|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:100
  <unit> ::= PCT
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"DISP:WATERFALL:BAS 67 PCT"
OUTPUT 711;"disp:waterfall:bas 57 PCT"
```

Query Syntax: `DISPlay:WATerfall:BASeline?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

This command allows you to mask a portion of each trace from the waterfall display. The percentage you specify is applied from the baseline of the trace, up towards the peak value and simplifies the waterfall by removing noise floor from the display.

For example, if DISP:WAT:BASE is 33 PCT, the lower third of the trace—from the baseline to 33 percent of the amplitude of the trace height—is suppressed. See the DISPlay:WATerfall:HEIGht command for more information.

DISPlay:WATerfall:COUNt command/query

Determines the number of traces displayed in the waterfall display.

Command Syntax:

```
DISPlay:WATerfall:COUNt (<number>[<unit>])|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  1e-6:9.9e37
       <unit> ::= [S|RPM|COUNT|AVG]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Display:Wat:Coun 5"
OUTPUT 711;"DISPLAY:WAT:COUNT 10"
```

Query Syntax:

```
DISPlay:WATerfall:COUNt?
```

Return Format:

```
NR3
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +15
SCPI Compliance:  instrument-specific
```

Description:

The number of traces displayed in a waterfall is determined by the range of Z-axis values specified with this command. The analyzer *may* adjust the specified range to include the trace selected with the CALC:WAT:TRAC:SEL command.

Note

All traces in the waterfall display are deleted when the ABORT;:INIT:CONT ON command is sent.

DISPlay:WATerfall:HEIGht command/query

Specifies the height of the waterfall trace box.

Command Syntax:

```
DISPlay:WATerfall:HEIGht {<number><unit>}|<step>|<bound>

         <number> ::= a real number (NRf data)
                      limits:  1:100
           <unit> ::= PCT
           <step> ::= UP|DOWN
          <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"disp:wat:height 66 PCT"
OUTPUT 711;"Disp:Waterfall:Heig 5 PCT"
```

Query Syntax:

```
DISPlay:WATerfall:HEIGht?
```

Return Format:

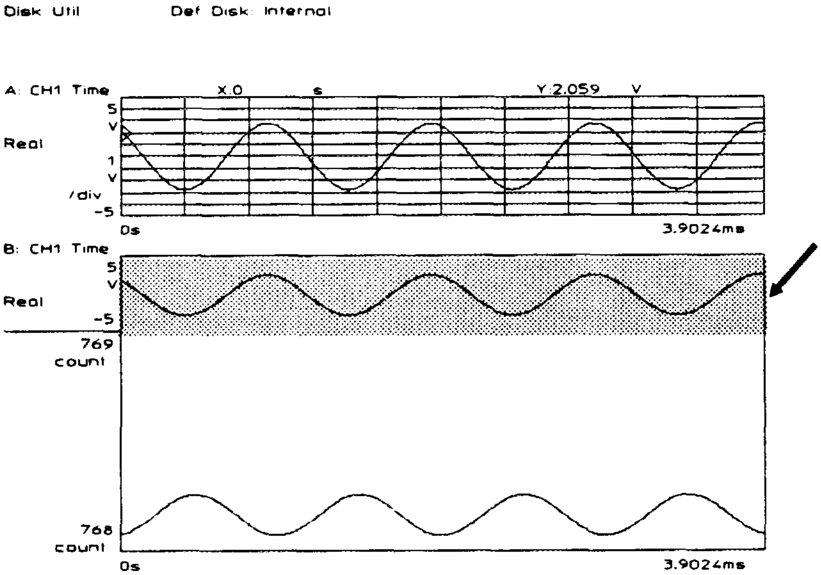
```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +39 PCT
SCPI Compliance:  instrument-specific
```

Description:

The value you specify determines the height of the waterfall trace box as a percentage of the total height of the waterfall display area. See figure 13-1.



Height is 25%

Figure 13-1. Height of the Waterfall Trace Box as a Percentage of the Total Height of the Waterfall Display Area

DISPlay:WATerfall:HIDDen command/query

Turns on or off the removal of hidden waterfall traces.

Command Syntax:

```
DISPlay:WATerfall:HIDDen {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":DISP:WATERFALL:HIDD OFF"
OUTPUT 711;"display:wat:hidd ON"
```

Query Syntax:

```
DISPlay:WATerfall:HIDDen?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

As traces are added to the waterfall display, segments of each trace may overlap and clutter the display making it difficult to read. When DISP:WAT:HID is ON (the default), the analyzer removes the hidden lines, those segments of the trace which fall behind or below the previous trace.

When DISP:WAT:HID is OFF, the analyzer displays all segments of all traces; even those obscured by the previous trace.

DISPlay[:WINDow[1|2]]:LIMit:STATe command/query

Turns limit lines on or off in the specified display.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:LIMit:STATe {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"Display:Wind:Limit:Stat OFF"
OUTPUT 711;"DISP:LIMIT:STAT OFF"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:LIMit:STATe?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

Sending DISP:LIM ON enables only the display of the limit lines in the specified trace. For example, send DISP:WIND2:LIM:STAT ON to turn on limit line for trace B. To test the trace against those lines, you must send CALC2:LIM ON. If a trace specifier is not used, the command defaults to trace A.

Note A trace can be evaluated against limits even when limit lines are not displayed.

DISPlay[:WINDow[1|2]]:TRACe:GRATicule:GRID[:STATe] command/query

Turns the display's overlay grid on or off.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:GRATicule:GRID[:STATe]
{OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":display:trac:grat:grid ON"
OUTPUT 711;"Disp:Window2:Trac:Grat:Grid:Stat OFF"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:GRATicule:GRID[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance: confirmed
```

Description:

The overlay grid (graticule) is not displayed on the analyzer's screen when it is turned off. In addition, the overlay grid is not plotted or printed.

DISPlay[:WINDow[1|2]]:TRACe:LABel command/query

Loads a label for the specified trace.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:LABel <STRING>
```

Example Statements:

```
OUTPUT 711;"DISPLAY:WIND2:TRAC:LABEL 'CEPSTRUM'"
OUTPUT 711;"disp:trace:lab 'SPL'"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]*]:TRACe:LABel?
```

Return Format:

```
STRING
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Trace titles replace the default trace titles supplied by the analyzer. They appear above the upper-left corner of traces. DISPlay:WINDow2:TRACe:LABel specifies trace B. If you do not send the optional trace specifier the command defaults to trace A.

Trace titles can be a maximum of 13 characters long.

You can remove the trace title by sending DISP:TRAC:LAB:DEF ON; the analyzer restores the default trace title.

Note

If you send *RST or SYST:PRES, trace titles are automatically erased. The analyzer restores the default title for both traces.

DISPlay[:WINDow[1|2]]:TRACe:LABel:DEFault[:STATe] command/query

Turns on or off the analyzer's default title for the specified trace.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:LABel:DEFault[:STATe]
{OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":Disp:Trace:Lab:Default ON"
OUTPUT 711;"DISP:WIND2:TRACE:LAB:DEFAULT:STAT OFF"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:LABel:DEFault[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State: +1
SCPI Compliance:  instrument-specific
```

Description:

See the DISP[:WINDow[1|2]]:TRACE:LABEL command for information on providing your own trace titles.

DISPlay[:WINDow[1|2]]:TRACe:X:MATCh[1|2] command

Modifies the X-axis scaling of a trace to match the X-axis scaling of the reference trace.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:X:MATCh[1|2]
```

Example Statements:

```
OUTPUT 711;"disp:window:trac:x:matc"
OUTPUT 711;"Disp:Trace:X:Match2"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command modifies the scaling of the X-axis of the trace selected with the [:WINDow[1|2]] trace specifier to match the X-axis scaling of the reference trace. The MATCH[1|2] trace specifier selects the reference trace.

The X-axis of the traces is not tested for compatibility. The analyzer uses the current start and stop X-axis values of the reference trace for the trace selected with the WINDow trace specifier. In addition, the analyzer modifies the spacing of the X-axis (specified with the DISP:TRAC:X:SPAC command) to that of the reference trace.

DISPlay[:WINDow[1|2]]:TRACe:X[:SCALe]:AUTO command/query

Scales the measurement data to fit the trace box.

Command Syntax: `DISPlay[:WINDow[1|2]]:TRACe:X[:SCALe]:AUTO {OFF|0|ONCE}`

Example Statements:

```
OUTPUT 711;":DISP:TRAC:X:AUTO ONCE"
OUTPUT 711;"display:wind2:trac:x:scal:auto OFF"
```

Query Syntax: `DISPlay[:WINDow[1|2]]:TRACe:X[:SCALe]:AUTO?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

To scale the measurement data, send DISP:TRAC:X:AUTO ONCE.

OFF has no effect on the analyzer. ON is not a valid option, because the analyzer does not support continuous scaling of the data along the X-axis.

DISPlay[:WINDow[1|2]]:TRACe:X[:SCALe]:LEFT command/query

Specifies the first X-axis value on the display.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:X[:SCALe]:LEFT
{<number>[<unit>]}|<bound>

   <number> ::= a real number (NRf data)
                limits:  -9.9e37:9.9e37
     <unit> ::= [HZ|S|ORD|COUNT|AVG|V|VPK|RPM|EU|REV]
    <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Disp:Wind:Trace:X:Scale:Left 4.9e+04 HZ"
OUTPUT 711;"DISP:TRACE:X:LEFT 1.1 MS"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:X[:SCALe]:LEFT?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

This command specifies the value of the first (most left) X-axis point on the display.

To determine the X-axis unit, send DISP:TRAC:X:LEFT? UNIT.

DISPlay[:WINDow[1|2]]:TRACe:X[:SCALe]:RIGHt command/query

Specifies the last X-axis value on the display.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:X[:SCALe]:RIGHt (<number>[<unit>])|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
  <unit> ::= [HZ|S|ORD|COUNT|AVG|V|VPK|RPM|EU|REV]
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":disp:trac:x:righ 35 KHZ"
OUTPUT 711;"Display:Wind:Trac:X:Scal:Right 2.4 ms"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:X[:SCALe]:RIGHt?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1.02400000E+005 (Channel 1)
               +3.90243530E-003 (Channel 2)
SCPI Compliance:  confirmed
```

Description:

This command specifies the value of the last (most right) X-axis point on the display.

To determine the X-axis unit, send DISP:TRAC:X:RIGH? UNIT.

DISPlay[:WINDow[1|2]]:TRACe:X:SPACing command/query

Specifies X-axis scaling.

Command Syntax: `DISPlay[:WINDow[1|2]]:TRACe:X:SPACing (LINear|LOGarithmic)`

Example Statements:

```
OUTPUT 711;"DISP:WIND2:TRACE:X:SPACING LOGarithmic"
OUTPUT 711;"disp:trac:x:spac LINear"
```

Query Syntax: `DISPlay[:WINDow[1|2]]:TRACe:X:SPACing?`

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  LIN
SCPI Compliance:  confirmed
```

Description:

To select linear scaling of the X-axis, send DISP:TRAC:X:SPAC LIN.

To select logarithmic scaling the X-axis, send DISP:TRAC:X:SPAC LOG.

DISPlay[:WINDow[1|2]]:TRACe:Y:MATCh[1|2] command

Modifies the Y-axis scaling of a trace to match the Y-axis scaling of the reference trace.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y:MATCh[1|2]
```

Example Statements:

```
OUTPUT 711;":Display:Trac:Y:Match"
OUTPUT 711;"DISP:WINDOW:TRAC:Y:MATCH"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command modifies the scaling of the Y-axis of the trace selected with the [:WINDow[1|2]] trace specifier to match the Y-axis scaling of the reference trace. The MATCH[1|2] trace specifier selects the reference trace.

The Y-axis of both traces must be compatible. Scaling of the Y-axis is specified with the CALCulate:FORMat command. The following coordinate systems are compatible:

- Linear magnitude for both traces (CALC:FORM MLIN).
- Linear magnitude data on a logarithmic Y-axis for both traces (CALC:FORM MLIN;DISP:TRAC:Y:SPAC LOG).
- Logarithmic magnitude for both traces (CALC:FORM MLOG).
- Phase (wrapped or unwrapped) for both traces (CALC:FORM PHAS or CALC:FORM UPH).
- Numbers (real or imaginary) for both traces (CALC:FORM REAL or CALC:FORM IMAG).
- Nyquist diagram for both traces (CALC:FORM NYQ).

If the trace formats are not compatible the Y-axis is *not* modified.

DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:AUTO command/query

Scales and repositions the trace vertically to provide the best display of trace data.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:AUTO
{OFF|0|ON|1|ONCE}
```

Example Statements:

```
OUTPUT 711;"disp:window:trac:y:scale:auto OFF"
OUTPUT 711;"Display:Trac:Y:Auto OFF"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:AUTO?
```

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  approved
```

Description:

To perform a single autoscale of the specified trace, send DISP:TRAC:Y:AUTO ONCE. The analyzer's autoscaling algorithm changes the values of DISP:TRAC:Y:REF and DISP:TRAC:Y:PDIV to optimize Y-axis scaling which provides the best display of your data.

ON autoscales the specified trace after every display update. OFF disables autoscaling.

DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:BOTTom command/query

Specifies the value of the bottom reference point of the display's Y-axis scale.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:BOTTom
(<number>[<unit>])|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
  <unit> ::= [DBVRMS|VRMS|VPK|DBVPK|V|DBV|EU|DBEU]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":DISP:TRACE:Y:BOTT 8.77578e+37"
OUTPUT 711;"display:wind:trace:y:scal:bottom -7.19304e+37"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:BOTTom?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  -1.51000000E+002 (Trace A)
               -5.00000000E-003 (Trace B)
SCPI Compliance:  confirmed
```

Description:

This command defines the bottom of a display's Y-axis scale. Specifying a Y-axis per-division value (DISP:TRAC:Y:PDIV) after using this command changes the top and center points of the display. The bottom point remains fixed.

See "Determining Units" in appendix E for information about available Y-axis units.

DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:CENTer command/query

Specifies the value of the center reference point of the display's Y-axis scale.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:CENTer {<num-
ber>[<unit>]}|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  -9.9e37:9.9e37
       <unit> ::= [DBVRMS|VRMS|VPK|DBVPK|V|DBV|EU|DBEU]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Disp:Window:Trac:Y:Scale:Cent -4.5528e+37"
OUTPUT 711;"DISPLAY:TRAC:Y:CENTER 9.64402e+36"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:CENTer?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  -5.10000000+E001 (Trace A)
               +0.00000000 (Trace B)
SCPI Compliance:  instrument-specific
```

Description:

This command defines the center of a display's Y-axis scale. Specifying a Y-axis per-division value (DISP:TRAC:Y:PDIV) after using this command changes the top and bottom points of the display. The center point remains fixed.

See "Determining Units" in appendix E for information about available Y-axis units.

DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:PDIVision command/query

Defines the height of each vertical division on the specified trace.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:PDIVision {<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
                 limits:  0:9.9e37
  <unit> ::= [DB|VRMS|VPK|V]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":disp:trace:y:pdiv 5"
OUTPUT 711;"Display:Wind2:Trace:Y:Scal:Pdivision 20"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:PDIVision?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1.00000000 E-001 (Trace A)
SCPI Compliance: approved
```

Description:

This command compresses or expands displayed data along the Y-axis.

The preset value for trace B is determined by the input range.

When trace coordinates are dB magnitude (CALC:FORM MLOG) the only valid unit selection is dB. See “Determining Units” in appendix E for information about available Y-axis units.

DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:REFerence command/query

Determines the Y-axis reference position for the specified display.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:REFerence
{TOP|CENTer|BOTTom|RANGe}
```

Example Statements:

```
OUTPUT 711;"DISP:WINDOW:TRAC:Y:SCALE:REF CENTer"
OUTPUT 711;"display:trac:y:reference BOTTom"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:REFerence?
```

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State: RANG
SCPI Compliance:  instrument-specific
```

Description:

When you change the height of the vertical division of the trace (DISP:TRAC:Y:PDIV) the specified Y-axis reference position remains constant or fixed. The specified trace moves up or down in the display area when you change the reference position.

- To fix the top of the specified display as the Y-axis reference point, send DISP:TRAC:Y:REF TOP.
- To fix the center of the specified display as the Y-axis reference point, send DISP:TRAC:Y:REF CENT.
- To fix the bottom of the specified display as the Y-axis reference point, send DISP:TRAC:Y:REF BOTT.

To specify a value for the reference point, send the DISP:WIND:TRACE:Y:TOP, DISP:WIND:TRACE:Y:CENT, or DISP:WIND:TRACE:Y:BOTT command.

To select Y-axis scaling which is based on the input range of the channel supplying measurement data, send DISP:TRAC:Y:REF RANG. This is called automatic reference tracking.

Your selection of measurement data (CALC:FEED) and trace coordinates (CALC:FORM) affects reference level tracking.

- When linear magnitude trace coordinates are selected, the bottom reference is kept at 0 (zero). The height of the vertical division of the trace (DISP:TRAC:Y:PDIV) is changed so the top reference is the input range.

- When logarithmic magnitude trace coordinates are selected, the top reference is kept at the input range.

- When the real or imaginary trace coordinates are selected, the center reference is set to 0 (zero). The height of the vertical division of the trace (DISP:TRAC:Y:PDIV) is changed so the top reference is ≥ the input range.

Reference level tracking is not allowed for phase trace coordinates (CALC:FORM:PHAS and CALC:FORM:UPH). It is also not valid with frequency response measurement data (CALC:FEED 'XFR:POW:RAT 2,1'), coherence measurement data (CALC:FEED 'XFR:POW:COH 1,2') or user math data (CALC:MATH:STAT ON).

Reference level tracking is disabled when autoscaling is turned on (DISP:TRAC:X:AUTO or DISP:TRAC:Y:AUTO). It is also disabled when the height of the vertical division (DISP:TRAC:Y:PDIV) changes for real (CALC:FORM REAL), imaginary (CALC:FORM IMAG) or linear magnitude (CALC:FORM MLIN) trace coordinates.

DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:TOP command/query

Specifies the value of the top reference point of the display's Y-axis scale.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:TOP
(<number>[<unit>])|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
  <unit> ::= [DBVRMS|VRMS|VPK|DBVPK|V|DBV|EU|DBEU]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Disp:Trace:Y:Top 6.59632e+37"
OUTPUT 711;"DISPLAY:WIND2:TRACE:Y:SCAL:TOP 7.45089e+37"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y[:SCALe]:TOP?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  -5.10000000E+01 (Trace A)
               +5.10000000E-003 (Trace B)
SCPI Compliance: confirmed
```

Description:

This command defines the top of a display's Y-axis scale. Specifying the height of the vertical division of the trace (DISP:TRAC:Y:PDIV) after using this command changes the center and bottom points of the display. The top point remains fixed.

See "Determining Units" in appendix E for information about available Y-axis units.

DISPlay[:WINDow[1|2]]:TRACe:Y:SPACing command/query

Specifies scaling of the Y-axis for linear magnitude coordinate data.

Command Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y:SPACing {LINear|LOGarithmic}
```

Example Statements:

```
OUTPUT 711;"disp:window:trac:y:spacing LOGarithmic"
OUTPUT 711;"Disp:Trace:Y:Spac LINear"
```

Query Syntax:

```
DISPlay[:WINDow[1|2]]:TRACe:Y:SPACing?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State: LIN
SCPI Compliance:  confirmed
```

Description:

To display linear magnitude coordinate data on a linear Y-axis scale, send DISP:TRAC:Y:SPAC LIN.

To display linear magnitude coordinate data on a logarithmic Y-axis scale, send DISP:TRAC:Y:SPAC LOG.

Use this command with the CALC:FORM MLIN command to display linear magnitude data on a logarithmic Y-axis scale, CALC:FORM;MLIN;:DISP:TRAC:Y:SPAC LOG.

Note Only magnitude data can be displayed on a logarithmic Y-axis scale.

14

FORMat

This subsystem contains one command—FORMat:DATA. The command determines which data type and data encoding is used when large blocks of numeric data are transferred between the HP 35665A and a controller.

FORMat[:DATA] command/query

Specifies the data type and date encoding to be used during transfers of a data block.

Command Syntax:

```
FORMat[:DATA] {ASCii|REAL}, [<number>|<bound>]

     <number> ::= a real number (NRf data)
                      limits:  3:64

      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":FORMAT ASCii, 8"
OUTPUT 711;"form:data REAL, 64"
```

Query Syntax:

```
FORMat[:DATA]?
```

Return Format:

```
CHAR, NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  ASC 12
SCPI Compliance:  confirmed
```

Description:

FORM:DATA only affects data transfers initiated by the following commands:

- CALC:DATA?
- CALC:CFIT:DATA?
- CALC:LIM:LOW:REP?
- CALC:LIM:LOW:SEGM
- CALC:LIM:UPP:REP?
- CALC:LIM:UPP:SEGM
- CALC:SYNT:DATA?
- CALC:WAT:DATA?
- PROG:SEL:NUMB
- TRAC[:DATA]
- TRAC:WAT[:DATA]
- TRAC:X[:DATA]?

FORM:DATA ASC selects NRf data for transfers *to* the analyzer and NR3 data for transfers *from* the analyzer. Data encoding is ASCII. You control the number of significant digits in the returned numbers with the second parameter, which has a range of 3 through 12 when the first parameter is ASC.

Note Data can be sent to the analyzer in ASCII, even if FORMat[:DATA] is REAL.

FORM:DATA REAL selects definite or indefinite length block data for transfers *to* the analyzer but only definite length block data for transfers *from* the analyzer. Data encoding is binary (the binary floating-point format defined in the IEEE 754-1985 standard). The only allowed values for the second parameter are 32 and 64; it determines how many bits are used for each number.

Note It is easiest for HP Instrument BASIC to read numbers if the format is REAL, 64.

See "Data Encoding for Block Data" in chapter 4 for more information.

15

HCOPy

The commands in this subsystem control the HP 35665A's print and plot operations. It contains commands that allow you to plot different portions of the analyzer's screen, use a time stamp, direct the print/plot operation to the internal disk drive or over the HP-IB.

The commands grouped under the PLOT mnemonic control plotting parameters. The command under the PRINt mnemonic tells the analyzer where to send the print data. The analyzer's print operation prints the entire screen. You cannot specify a portion of the screen to be printed.

HCOPy:DESTination command/query

Specifies where the print or plot operation is sent: either directly to the HP-IB device or to a file on the default disk.

Command Syntax:

```
HCOPy:DESTination (HPIB|FILE)
```

Example Statements:

```
OUTPUT 711;"Hcop:Dest FILE"
OUTPUT 711;"HCOPY:DEST HPIB"
```

Query Syntax:

```
HCOPy:DESTination?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  HPIB
SCPI Compliance:  instrument-specific
```

Description:

HPIB sends the print/plot directly to the HP-IB device.

FILE sends the print/plot to a file on the default disk. Specify the filename with the HCOP:FIL command. For information on how to send the print/plot operation to a file on a mass storage device *other* than the default disk, see the HCOPy:FILename command.

HCOPy:DEVice command/query

Specifies the output device.

Command Syntax:	`HCOPy:DEVice {PLOT|PRINt}`

Example Statements:	`OUTPUT 711;":hcopy:dev PRINt"`
	`OUTPUT 711;"Hcop:Device PLOT"`

Query Syntax:	`HCOPy:DEVice?`
Return Format:	CHAR

Attribute Summary:	`Option:  not applicable`
	`Overlapped:  no`
	`Preset State:  PLOT`
	`SCPI Compliance:  instrument-specific`

Description:

Use PLOT to specify a plotter. Use PRIN to specify a printer.

Note You can not use the HCOPy:SOURce commands if you specify a printer (HCOP:DEV PRIN).

HCOPy:EJECt

command/query

Turns the page-eject feature on or off.

Command Syntax:

```
HCOPy:EJECt {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"HCOP:EJECT ON"
OUTPUT 711;"hcop:ejec ON"
```

Query Syntax:

```
HCOPy:EJECt?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

Check the documentation for your device to verify that it supports the requested page-eject state.

HCOPy:FILename command/query

Specifies a filename for the output of a print or plot operation.

Command Syntax: `HCOPy:FILename <FILENAME>`

```
    <FILENAME> ::= [<MSIS>]<filename>
        <MSIS> ::= RAM:|NVRAM:|INT:|
                   EXT[,<select_code>[,<unit_number>]]:
    <filename> ::= ASCII characters (see description for
                   <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":Hcopy:Fil FILENAME"
OUTPUT 711;"HCOPY:FIL FILENAME"
```

Query Syntax: `HCOPy:FILename?`

Return Format: `STRING`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command is used with the HCOP:DEST FILE command.

The valid character set for <filename> depends on the disk format.

DOS file names are limited to 8 ASCII characters followed by a period and a three character extension. The period and extension are not required. File names are not case sensitive.

LIF file names are limited to 10 character which may include all characters except ":" "<" and "|". The first character must be a letter. File names are case sensitive.

HCOPy[:IMMediate] command

Plots or prints the currently specified item.

Command Syntax:

```
HCOPy[:IMMediate]
```

Example Statements:

```
OUTPUT 711;"hcop:immediate"
OUTPUT 711;"Hcop"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If this command is sent when the HP-IB is the specified destination (HCOP:DEST HPIB); the analyzer expects to find a plotter at the address specified with the HCOP:PLOT:ADDR command or a printer at the HCOP:PRIN:ADDR command. Of a device is not at the specified address, the operation is aborted.

See the HCOP:DEST FILE command for more information about printing or plotting to a file.

Note

Pass Control Required.

See chapter 2, "Passing Control" for information on how to pass control to the analyzer.

HCOPy:PLOT:ADDRess command/query

Tells the analyzer which HP-IB address is assigned to your plotter.

Command Syntax:

```
HCOPy:PLOT:ADDRess <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:30

  <step> ::= UP|DOWN

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":HCOPY:PLOT:ADDR 5"
OUTPUT 711;"hcopy:plot:address 9"
```

Query Syntax: `HCOPy:PLOT:ADDRess?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State: not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

When you initiate a plot with the HCOPy[:IMM] command, the analyzer expects to find a plotter at the HP-IB address specified with HCOP:PLOT:ADDR. If a plotter is not at the specified address, the plot is automatically aborted.

HCOPy:PLOT:LTYPe:TRACe[1|2] command/query

Selects the line type for the specified trace.

Command Syntax:

```
HCOPy:PLOT:LTYPe:TRACe[1|2] <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -6:6
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Hcop:Plot:Ltype:Trac 2"
OUTPUT 711;"HCOPY:PLOT:LTYP:TRACE -1"
```

Query Syntax: `HCOPy:PLOT:LTYPe:TRACe[1|2]?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0 (both traces)
SCPI Compliance:  instrument-specific
```

Description:

The <number> parameter is an encoded value. Encoded values for the most commonly used line types are:

- Solid: 0
- Dotted: 1
- Dashed: 2

Check your plotter's documentation to see if it supports additional line types.

The trace specifier determines whether you are selecting the line type for trace A or trace B. Omit the specifier or send 1 for trace A; send 2 for trace B.

HCOPy:PLOT:PEN:ALPHa command/query

Selects the pen used for plotting miscellaneous annotations.

Command Syntax:

```
HCOPy:PLOT:PEN:ALPHa <number>|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  0:16
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":hcop:plot:pen:alph 14"
OUTPUT 711;"Hcopy:Plot:Pen:Alph 15"
```

Query Syntax: `HCOPy:PLOT:PEN:ALPHa?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +4
SCPI Compliance:  instrument-specific
```

Description:

The alpha pen is used to plot the following:

- Instrument state.
- Disk catalog.
- Mini-state.
- Fault log.
- Test log.
- Time stamp.

Nothing is plotted with a pen whose value is specified as 0 or with a pen whose specified value is too large for your plotter.

HCOPy:PLOT:PEN:DEFault command

Specifies default values for the plotter pen assignments.

Command Syntax:

```
HCOPy:PLOT:PEN:DEFault
```

Example Statements:

```
OUTPUT 711;"HCOP:PLOT:PEN:DEFAULT"
OUTPUT 711;"hcop:plot:pen:def"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The default plotter pen assignments are as follows:

- PLOT:PEN:ALPH = 4
- PLOT:PEN:GRID = 1
- PLOT:PEN:MARK1 = 5
- PLOT:PEN:MARK2 = 6
- PLOT:PEN:TRAC1 = 2
- PLOT:PEN:TRAC2 = 3

HCOPy:PLOT:PEN:GRID command/query

Selects the pen used to plot the overlay grid.

Command Syntax:

```
HCOPy:PLOT:PEN:GRID <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:16
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Hcopy:Plot:Pen:Grid 1"
OUTPUT 711;"HCOP:PLOT:PEN:GRID 15"
```

Query Syntax: `HCOPy:PLOT:PEN:GRID?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

The grid pen is used to plot the overlay grid (trace graticules), the border around the instrument state and the border around the disk catalog.

HCOPy:PLOT:PEN:MARKer[1|2] command/query

Selects the pen used to plot markers for the specified trace.

Command Syntax:

```
HCOPy:PLOT:PEN:MARKer[1|2] <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:16
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"hcopy:plot:pen:mark 0"
OUTPUT 711;"Hcop:Plot:Pen:Marker 9"
```

Query Syntax: `HCOPy:PLOT:PEN:MARKer[1|2]?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +5(MARK1)
               +6 (MARK2)
SCPI Compliance:  instrument-specific
```

Description:

The marker pen is used to plot all markers; including the main markers, limit lines, the marker reference and marker functions.

The trace specifier you send with this command determines whether you are selecting the pen number for trace A markers or trace B markers. Omit the specifier or send 1 for trace A; send 2 for trace B.

HCOPy:PLOT:PEN:TRACe[1|2] command/query

Selects the pen used to plot the specified trace and annotation.

Command Syntax:

```
HCOPy:PLOT:PEN:TRACe[1|2] <number>|<step>|<bound>

      <number> ::= a real number (NRf data)
                   limits:  0:16

        <step> ::= UP|DOWN

       <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":HCOP:PLOT:PEN:TRAC 0"
OUTPUT 711;"hcopy:plot:pen:trace 6"
```

Query Syntax: `HCOPy:PLOT:PEN:TRACe[1|2]?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +2 (TRAC1)
               +3 (TRAC2)
SCPI Compliance:  instrument-specific
```

Description:

The trace pen is used to plot traces and all of the following trace-specific annotation:

- Trace title
- Marker readout
- X-axis annotation
- Y-axis annotation

The trace specifier you send with this command determines whether you are selecting the pen number for trace A or trace B. Omit the specifier or send 1 for trace A; send 2 for trace B.

HCOPy:PLOT:SPEed command/query

Specifies the plotting speed for all plotting operations initiated by the analyzer.

Command Syntax:

```
HCOPy:PLOT:SPEed <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  1:100
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Hcop:Plot:Spe 53"
OUTPUT 711;"HCOP:PLOT:SPE 28"
```

Query Syntax: `HCOPy:PLOT:SPEed?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +50
SCPI Compliance:  instrument-specific
```

Description:

This command allows you to specify the plotting speed in units of centimeters per second (cm/s). Check your plotter's documentation to be sure that it supports the requested plotting speed.

For example, send HCOP:PLOT:SPE 50 to select a plotting speed of 50 cm/second. Send HCOP:PLOT:SPE 10 to select a slower plotting speed of 10 cm/second.

HCOPy:PRINt:ADDRess command/query

Tells the analyzer which HP-IB address is assigned to your printer.

Command Syntax:

```
HCOPy:PRINt:ADDRess <number>|<step>|<bound>

   <number> ::= a real number (NRf data)
                limits:  0:30
     <step> ::= UP|DOWN
    <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":hcopy:prin:addr 3"
OUTPUT 711;"Hcopy:Prin:Address 1"
```

Query Syntax: `HCOPy:PRINt:ADDRess?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

When you initiate a print operation with the HCOPy[:IMM] command, the analyzer expects to find a printer at the HP-IB address specified with HCOP:PRIN:ADDR. If a printer is not at the specified address, the print operation is automatically aborted.

HCOPy:SOURce

command/query

Selects the portion of the analyzer's screen you want to plot.

Command Syntax:

```
HCOPy:SOURce {ALL|TRACe|MARKer|REFerence|GRID}
```

Example Statements:

```
OUTPUT 711;"HCOP:SOUR GRID"
OUTPUT 711;"hcopy:sour MARKer"
```

Query Syntax:

```
HCOPy:SOURce?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  ALL
SCPI Compliance:  instrument-specific
```

Description:

This command is not valid when the specified device is a printer (HCOP:DEV PRIN). This command can only be used with the HCOP:DEV PLOT command.

To plot everything currently displayed on the analyzer's screen, send HCOP:SOUR ALL;IMM . Everything on the screen is plotted *except* the status line and the softkey menu.

To plot the displayed trace(s), send HCOP:SOUR TRAC;IMM. Traces are plotted without grid lines, annotation or markers.

Send HCOP:SOUR MARK;IMM to plot the main marker for all displayed trace(s) The main marker must be displayed (CALC:MARK ON) before it can be plotted. The marker is annotated with its X-axis and Y-axis coordinates. The annotation appears above the marker.

Send HCOP:SOUR REF;IMM to plot the marker reference. The marker reference must be displayed (CALC:MARK:MODE REL) before it can be plotted. The marker reference is annotated with its X-axis and Y-axis coordinates. The annotation appears above the marker reference.

Send HCOP:SOUR GRID;IMM to plot the graticule only for all displayed traces. The grid is plotted without the trace, markers or annotation.

HCOPy:TSTamp:MODE **command/query**

Specifies the format of the time stamp used for plotting and printing.

Command Syntax:

```
HCOPy:TSTamp:MODE
{FORMat1|FORMat2|FORMat3|FORMat4|FORMat5}
```

Example Statements:

```
OUTPUT 711;":Hcopy:Tst:Mode FORMat1"
OUTPUT 711;"HCOPY:TST:MODE FORMat3"
```

Query Syntax:

```
HCOPy:TSTamp:MODE?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  FORM1
SCPI Compliance:  instrument-specific
```

Description:

This command selects the time and date format for plot and print operations.

- To select a 24 hour, Day/Month/Year Hour:Minute:Second format, send HCOPY:TSTAMP:MODE FORM1.
- To select a 24 hour, Day. Month.Year Hour:Minute:Second format, send HCOPY:TSTAMP:MODE FORM2.
- To select a 24 hour, Year Month Day Hour:Minute:Second format, send HCOPY:TSTAMP:MODE FORM3.
- To select a 12 hour, Day/Month/Year Hour:Minute:Second AM format, send HCOPY:TSTAMP:MODE FORM4.
- To select a 12 hour, Month-Day-Year Hour:Minute:Second AM format, send HCOPY:TSTAMP:MODE FORM5.

HCOPy:TSTamp[:STATe] command/query

Turns a time stamp on or off for print and plot operations.

Command Syntax: `HCOPy:TSTamp[:STATe] {OFF|0|ON|1}`

Example Statements:

```
OUTPUT 711;"hcop:tst:state OFF"
OUTPUT 711;"Hcop:Tstamp OFF"
```

Query Syntax: `HCOPy:TSTamp[:STATe]?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

When time stamp is ON, time and date information is printed with the screen data you specify with the HCOP:SOUR command. Use the HCOP:TST:MODE command to specify the time stamp format.

When time stamp is OFF, time and date information is *not* printed.

INITiate

The INITiate subsystem controls the initiation of the TRIGger system. The commands initiate all TRIGger sequences as a group.

Figure 16-1 shows the model for the HP 35665A's ARM-INITiate-TRIGger functions.

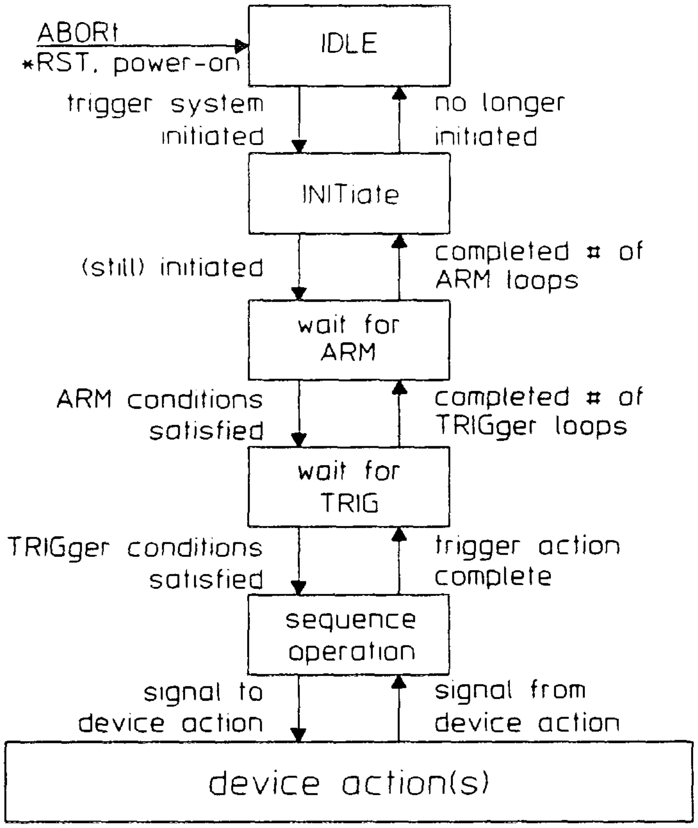


Figure 16-1. The HP 35665A's ARM-INITiate-TRIGger Functions

INITiate:CONTinuous **command/query**

Sets the trigger system to a continuously initiated state.

Command Syntax: `INITiate:CONTinuous (OFF|0|ON|1)`

Example Statements:

```
OUTPUT 711;":INIT:CONT OFF"
OUTPUT 711;"initiate:cont OFF"
```

Query Syntax: `INITiate:CONTinuous?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

INIT:CONT OFF suspends the measurement process. It "pauses" the current measurement. If the measurement is averaged (AVER:STAT ON), the current average is completed before the measurement pauses.

INIT:CONT ON restarts a "paused" measurement. It also allows you to add more data to the running average of a completed measurement. For example, if the analyzer has completed a 10-average measurement and you send INIT CONT ON, 10 more records are averaged with the old data, bringing the total number of averages to 20.

Note After *RST, INIT:CONT is set to +0 (OFF).

INIT:CONT ON is valid *only* if the measurement has been paused. You must use the INIT[:IMM] command to start a measurement for the first time.

INITiate[:IMMediate] command

Starts a measurement and forces the trigger system to exit the idle state.

Command Syntax:

```
INITiate[:IMMediate]
```

Example Statements:

```
OUTPUT 711;"Initiate:Imm"
OUTPUT 711;"INIT"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command starts a new measurement and ensures that any changes made to the analyzer's state are reflected in the measurement results. The new measurement is started immediately whether the current measurement is running, "paused," or completed. All data from the previous measurement is discarded when the new measurement is started.

INIT[:IMM] causes the trigger system to initiate and complete one full trigger cycle.

If the command INIT:CONT ON has been sent, the INIT[:IMM] command has no affect.

The program message ABOR;INIT:IMM serves a special synchronization function. When you send this message to restart a measurement, the analyzer's No Pending Operation (NPO) flag is set to 1 until the measurement is complete. The two commands that test the state of this flag—*WAI and *OPC—allow you to hold off subsequent actions until the measurement is complete. See "Synchronization" in chapter 2 for more information on the NPO flag.

17

INPut

The commands in this subsystem control the characteristics of the analyzer's input channels. They configure the inputs for Channel 1 and Channel 2.

Because there are two channels, you must specify the channel you want to configure when you send a command. If you do not explicitly specify one of the channels, the analyzer configures Channel 1.

Note The HP 35665A has two input channels (1 and 2) and two trace boxes (A and B). Neither of the two channels is linked to a particular trace box. You can display Channel 1 data in either trace box A or trace box B. The same is true for Channel 2 data.

INPut[1|2]:BIAS[:STATe] **command/query**

Enables/disables the ICP supply on the corresponding input channel.

Command Syntax:	`INPut[1|2]:BIAS[:STATe] {OFF|0|ON|1}`

Example Statements:	`OUTPUT 711;":input:bias ON"`
	`OUTPUT 711;"Input:Bias:Stat ON"`

Query Syntax:	`INPut[1|2]:BIAS[:STATe]?`
Return Format:	`NR1`

Attribute Summary:	`Option: not applicable`
	`Overlapped: no`
	`Preset State: +0`
	`SCPI Compliance: approved`

Description:

This command connects (or disconnects) the internal 4 mA current source to the input connector. The nominal voltage output is 24 V dc (open circuit).

When INP:BIAS is ON, the ICP ON indicator appears on the front panel.

If the channel specifier is not used, the command defaults to channel 1.

INPut[1|2]:COUPling command/query

Selects AC or DC coupling for the specified channel.

Command Syntax: `INPut[1|2]:COUPling {AC|DC}`

Example Statements:

```
OUTPUT 711;"INPUT:COUP AC"
OUTPUT 711;"input:coup DC"
```

Query Syntax: `INPut[1|2]:COUPling?`

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  DC
SCPI Compliance:  confirmed
```

Description:

If the channel specifier is not used, the command defaults to channel 1.

INPut[1|2]:FILTer:AWEighting[:STATe] command/query

Enables/disables the A-weight filter on the specified input channel.

Command Syntax: `INPut[1|2]:FILTer:AWEighting[:STATe] {OFF|0|ON|1}`

Example Statements:

```
OUTPUT 711;":Inp2:Filter:Awe OFF"
OUTPUT 711;"INPUT:FILT:AWE:STATE OFF"
```

Query Syntax: `INPut[1|2]:FILTer:AWEighting[:STATe]?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  approved
```

Description:

The A-weight filter is normally used with octave measurements.

If the channel specifier is not used, the command defaults to channel 1.

INPut[1|2]:FILTer[:LPASs][:STATe] command/query

Enables/disables the anti-alias filter for the specified input channel.

Command Syntax:

```
INPut[1|2]:FILTer[:LPASs][:STATe] (OFF|0|ON|1)
```

Example Statements:

```
OUTPUT 711;"inp:filter:lpas:stat ON"
OUTPUT 711;"Input:Filt ON"
```

Query Syntax:

```
INPut[1|2]:FILTer[:LPASs][:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

When INPut[1|2]:FILTer is OFF, the analyzer's corresponding input bypasses the anti- alias low pass filter. Measurement results are *not* corrected for front end flatness; only front end DC offset is calibrated.

If the channel specifier is not used, the command defaults to channel 1.

Note The anti-alias low pass filter is always bypassed in histogram instrument mode.

INPut[1|2]:LOW command/query

Sets the specified channel's input shield to float or to ground.

Command Syntax: `INPut[1|2]:LOW {GROund|FLOat}`

Example Statements:

```
OUTPUT 711;":INPUT:LOW FLOat"
OUTPUT 711;"inp:low FLOat"
```

Query Syntax: `INPut[1|2]:LOW?`

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State: FLO
SCPI Compliance:  confirmed
```

Description:

To connect the analyzer's input shield to ground through 55 Ω, send INP[1|2]:LOW GRO.

To float the analyzer's input shield through 1 MΩ, send INP[1|2]:LOW FLO. The input connector ground is not completely isolated from the chassis ground.

If the channel specifier is not used, the command defaults to channel 1.

INPut[1|2]:REFerence:DIRection command/query

Sets transducer direction for measurement point.

Command Syntax:

```
INPut[1|2]:REFerence:DIRection <number>|<bound>

        <number> ::= a real number (NRf data)
                     limits:  0:32767

         <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Inp:Reference:Dir 8223"
OUTPUT 711;"INP:REFERENCE:DIR 8404"
```

Query Syntax:

```
INPut[1|2]:REFerence:DIRection?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command allows you to document the directional placement of the transducer on the device under test. Directions available are:

- 0 = no direction
- 1 = X
- 2 = Y
- 3 = Z
- 4 = (radial)
- 5 = T (tangential θ $\angle$)
- 6 = P (tangential θ ϕ)
- 7 = TX
- 8 = TY
- 9 = TZ

The query response provides the current direction associated with the specified channel.

INPut[1|2]:REFerence:POINt command/query

Sets the number for the transducer point.

Command Syntax: `INPut[1|2]:REFerence:POINt <number>|<step>|<bound>`

```
<number> ::= a real number (NRf data)·
             limits:  0:32767
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":input:ref:poin 11081"
OUTPUT 711;"Input:Ref:Point 12720"
```

Query Syntax: `INPut[1|2]:REFerence:POINt?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command allows you to specify a number for the point at which the transducer is attached to the device under test.

The query returns the current point number associated with the input channel.

INPut[1|2][:STATe] command/query

Turns two-channel measurements on or off.

Command Syntax:	`INPut[1|2][:STATe] (OFF|0|ON|1)`

Example Statements:	`OUTPUT 711;"INP:STAT ON"`
	`OUTPUT 711;"input2 OFF"`

Query Syntax:	`INPut[1|2][:STATe]?`
Return Format:	`NR1`

Attribute Summary:	`Option: not applicable`
	`Overlapped: no`
	`Preset State: +1 (INPut1)`
	`+0 (INPut2)`
	`SCPI Compliance: confirmed`

Description:

To select a two-channel measurement, send INP2 ON.

To select a one-channel measurement, send INP2 OFF. The analyzer takes data from channel 1 only.

Note INP1 OFF is *not* a valid command. The channel 1 input cannot be disabled.

18

INSTrument

The commands in this subsystem select the instrument mode of the analyzer. Instrument mode specifies the type of measurement being made and whether signals applied to the front panel input connections are being measured.

Instrument mode is a major selection that changes the "personality" of the analyzer. This means that other parameters change when you change instrument mode.

Note The analyzer "remembers" a separate set of parameters for each instrument mode.

INSTrument:NSELect command/query

Selects one of the analyzer's six major instrument modes.

Command Syntax:

```
INSTrument:NSELect <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:5

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Inst:Nselect 2"
OUTPUT 711;"INST:NSEL 4"
```

Query Syntax:

```
INSTrument:NSELect?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D0 Computed Order Tracking
         1D1 Realtime Octave
         1D2 Swept Sine
Overlapped:  no
Preset State:  FFT (0)
SCPI Compliance:  confirmed
```

Description:

The following commands select a major instrument mode:

- INST:NSEL 0 – FFT Analysis
- INST:NSEL 4 – Histogram
- INST:NSEL 5 – Correlation Analysis

The following commands are valid if the appropriate option is installed:

- INST:NSEL 1 – Octave Analysis – Option 1D1
- INST:NSEL 2 – Order Analysis – Option 1D0
- INST:NSEL 3 – Swept Sine – Option 1D2

The HP-IB command set changes with each instrument mode. For example, the internal source is not used with order analysis and the SOUR commands are not valid when in that instrument mode. Parameters sent to setup a measurement for each instrument mode do not affect setups for other instrument modes. As a result, you should select the instrument mode near the beginning of any program sequence that defines the instrument state.

The default instrument mode is FFT at powerup and reset.

INSTrument:SELect command/query

Selects one of the analyzer's six major instrument modes.

Command Syntax:

```
INSTrument:SELect
{FFT|OCTave|ORDer|SINE|HISTogram|CORRelation}
```

Example Statements:

```
OUTPUT 711;"instrument:sel FFT"
OUTPUT 711;"Instrument:Sel OCTave"
```

Query Syntax:

```
INSTrument:SELect?
```

Return Format:

CHAR

Attribute Summary:

```
Option:   1D0 Computed Order Tracking
          1D1 Realtime Octave
          1D2 Swept Sine
Overlapped:  no
Preset State:  FFT
SCPI Compliance:  confirmed
```

Description:

The following commands select a major instrument mode:

- INST:SEL CORR – Correlation Analysis
- INST:SEL FFT – FFT Analysis
- INST:SEL HIST – Histogram

The following commands are valid if the appropriate option is installed:

- INST:SEL OCT – Octave Analysis – Option 1D1
- INST:SEL ORD – Order Analysis – Option 1D0
- INST:SEL SINE – Swept Sine – Option 1D2

The HP-IB command set changes with each instrument mode. For example, the internal source is not used with order analysis and the SOUR commands are not valid when in that instrument mode. Parameters sent to setup a measurement for each instrument mode do not affect setups for other instrument modes. As a result, you should select the instrument mode near the beginning of any program sequence that defines the instrument state.

19

MEMory

This subsystem contains commands which manage instrument memory. This excludes memory used for mass storage, which is defined in the MMEMory subsystem.

MEMory:CATalog[:ALL]? query

Returns information on the current contents and state of the analyzer's memory.

Query Syntax: `MEMory:CATalog[:ALL]?`

Example Statements:

```
OUTPUT 711;":MEM:CATALOG?"
OUTPUT 711;"mem:catalog:all?"
```

Return Format:

```
<bytes_in_use>,<bytes_available>,<ITEM>,<ITEM>,
<ITEM>,<ITEM>,<ITEM>
```

```
    <bytes_in_use> ::= total amount of memory currently allocated, in bytes
 <bytes_available> ::= largest memory block currently available, in bytes
          <ITEM> ::= <NAME>,<TYPE>,<SIZE>
          <NAME> ::= TCAPture
                     WATerfall
                     WREG (waterfall register)
                     PROG (HP Instrument BASIC program)
                     RDISK (volatile RAM disk)
          <TYPE> ::= BIN (binary file)
          <SIZE> ::= size of file in bytes
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

Use this query to determine the analyzer's memory usage.

The analyzer allows you to allocate memory for the following items:

- time capture buffer
- waterfall display
- waterfall registers
- HP Instrument BASIC programs
- the analyzer's volatile RAM disk

This query returns a directory list and memory sizes for these items. The query returns <NAME> in short form (for example, TCAP for time capture).

MEMory:CATalog:NAME? query

Returns information about memory usage allocated for a specific item.

Query Syntax:

```
MEMory:CATalog:NAME?
{TCAPture|WATerfall|WREGister|PROGram|RDISk}
```

Example Statements:

```
OUTPUT 711;"Mem:Catalog:Name? WATerfall"
OUTPUT 711;"MEMORY:CAT:NAME? WREGister"
```

Return Format:

```
STRING
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the amount of the memory allocated for a specific item.

To allocate memory for a time capture buffer, send the [SENSe:]TCAPture:MALLocate command.

To allocate memory for a waterfall display, send the CALCulate:WATerfall:COUNt command.

The analyzer allocates memory for a waterfall register when you send the TRACe:WATerfall[:DATA] command.

The analyzer allocates memory for an HP Instrument BASIC program when you create the program. The memory usage is allocated for all HP Instrument BASIC programs. If multiple programs reside within the analyzer, you can not determine the memory allocated for a single HP Instrument BASIC program with this command.

To allocate memory for the volatile RAM disk, send the MMEMory:INITialize command.

The query returns <NAME> in short form (for example, TCAP for time capture).

MEMory:DELete:ALL command

Purges all allocated memory in the analyzer.

Command Syntax:

```
MEMory:DELete:ALL
```

Example Statements:

```
OUTPUT 711;":memory:del:all"
OUTPUT 711;"Mem:Del:All"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command purges the analyzer's allocated memory for *all* the following items:

- time capture buffer
- waterfall display
- waterfall registers
- HP Instrument BASIC programs
- the analyzer's volatile RAM disk

The memory is available for reuse.

MEMory:DELete[:NAME] command

Purges the memory allocated for a specific item.

Command Syntax:

```
MEMory:DELete[:NAME]
{TCAPture|WATerfall|WREGister|PROGram|RDISk}
```

Example Statements:

```
OUTPUT 711;"MEM:DELETE:NAME WATerfall"
OUTPUT 711;"mem:delete RDISk"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command deletes the item. The analyzer's memory is available for reuse.

MEMory:FREE[:ALL]? query

Returns information on the state of the analyzer's memory.

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

Example Statements:

```
OUTPUT 711;":Mem:Free?"
OUTPUT 711;"MEM:FREE:ALL?"
```

Return Format: `NR1, NR1`

```
        <NR1>,<NR1> ::= <bytes_available>,<bytes_in_use>
  <bytes_available> ::= largest memory block currently available, in bytes
     <bytes_in_use> ::= total amount of memory currently allocated, in bytes
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This query returns two values.

The first value specifies the amount of memory currently available for allocation. The second value specifies the total amount of memory currently allocated for the following items.

- time capture buffer
- waterfall display
- waterfall registers
- HP Instrument BASIC programs
- the analyzer's volatile RAM disk

MMEMory

Commands in this subsystem control the analyzer's mass storage (disk) functions. Two of the mass storage devices are RAM-based disks—one using non-volatile RAM and the other using volatile RAM. Another mass storage device is an internal disk drive that uses 3.5 flexible disks. In addition, the analyzer can access an external disk drive. The disk drives (internal or external) can access either LIF or DOS disk formats. A maximum of five units can be active at any one time.

In most cases, if you do not send a mass storage specifier with a command that requires one, a default specifier is assumed. You select the default specifier with the MMEM:MSIS command.

Syntax Descriptions

Syntax descriptions in this chapter use the following conventions:

<FILE> ::=

This string is used to describe the name of a file. It does not include any disk drive information.

The valid character set for <FILE> depends on the disk format.

DOS file names are limited to 8 ASCII characters followed by a period and a three character extension. The period and extension are not required. File names are *not* case sensitive.

LIF file names are limited to 10 character which may include any character except ":" "," and "|". The first character must be a letter. File names *are* case sensitive.

<FILENAME> ::=

This string is used to describe the name of a file on the default disk or on the specified disk. The allowed form is:

"[MSIS:]filename"

where MSIS: must be replaced with:
RAM: which selects volatile RAM.
NVRAM: which selects non-volatile RAM.
INT: which selects the internal disk drive.
EXT[,<select_code>[,<unit_number>]]: which selects the external disk drive.

The brackets are not literal in the parameter. They indicate that the disk drive designation is optional. If the disk drive is not specified, the default disk drive is used. Select the default specifier with the MMEM:MSIS command.

Note

The <select_code> and <unit_number> specified with the MMEM:MSIS command becomes the *default* disk and unit address for the external disk drive (EXT:).

The valid character set for <FILENAME> depends on the disk format.

DOS file names are limited to 8 ASCII characters followed by a period and a three character extension. The period and extension are not required. File names are *not* case sensitive.

LIF file names are limited to 10 character which may include any character except ":" "," and "|". The first character must be a letter. File names *are* case sensitive.

<MSINAME> ::=

This parameter specifies the mass storage device. It is a <STRING> with one of the following values:

"RAM:"
"NVRAM:"
"INT:"
"EXT[,<select_code>[,<unit_number>]]:"

The brackets are not literal in the parameter. They indicate that the external disk drive's address is optional. If the <select_code> and <unit_number> are not specified, the analyzer uses the default values specified with the MMEM:MSIS command.

<MMEMNAME> ::=

This parameter specifies a single file or an entire mass storage device. It takes the same form as <FILENAME> except either the name of the file, the name of the mass storage device or both must be present. It can take one of the following forms:

"[MSIS:]filename"
"MSIS:[filename]"
"MSIS:filename"

See the description for <FILENAME> for the valid mass storage device specifiers and the valid file name character set.

MMEMory:COPY **command**

Copies the contents of one disk to another or one file to another.

Command Syntax:

```
MMEMory:COPY <MMEMNAME>, <MMEMNAME>

     <MMEMNAME> ::= <disk>[<filename>]
         <disk> ::=NVRAM:|RAM:|INT:|EXT[,<select_code>[,<unit_number>]]:
  <select_code> ::= a real number (NRf data)
                    limits:  700:730
  <unit_number> ::=a real number (NRf data)
                    limits:  0:3
     <filename> ::=ASCII characters (see beginning of chapter
                    for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"mmem:copy 'INT:FILE1', 'EXT:FILE1'"
OUTPUT 711;"Mmem:Copy 'EXT:', 'INT:'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The first <MMEMNAME> is the source; the second is the destination.

To copy a disk, use the disk specifier for each <MMEMNAME>. The select code and unit number specifiers are valid only with the EXT: disk specifier.

Caution All files on the destination disk are overwritten when you specify a a disk copy.

To copy a file, use disk specifiers and filenames. If you want to rename a file, use the MMEM:REN command.

Note When accessing the external mass storage device (EXT:); the active controller on the HP-IB must temporarily pass control to the analyzer. After the command has been executed, the analyzer must pass control back. See chapter 2, "Behavior in an HP-IB System," for more information on passing control.

MMEMory:DELete **command**

Deletes one file or the contents of an entire disk.

Command Syntax:

```
MMEMory:DELete <MMEMNAME>

     <MMEMNAME> ::= <disk>[<filename>]
         <disk> ::=NVRAM:|RAM:|INT:|EXT[,<select_code>[,<unit_number>]]:
  <select_code> ::= a real number (NRf data)
                    limits:  700:730
  <unit_number> ::= a real number (NRf data)
                    limits:  0:3
     <filename> ::= ASCII characters (see beginning of chapter
                    for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":MMEMORY:DEL 'INT:JUNK'"
OUTPUT 711;"mmemory:del 'EXT:STATE.STA'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes only for EXT:
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

To delete all files from a mass storage device only specify the <disk>. The select code and unit number specifiers are valid only with the EXT: disk specifier.

You can use the "*" as a wildcard (for example, '*.DAT').

Note When accessing the external mass storage device (EXT:); the active controller on the HP-IB must temporarily pass control to the analyzer. After the command has been executed, the analyzer must pass control back. See chapter 2, "Behavior in an HP-IB System," for more information on passings control.

MMEMory:DISK:ADDRess command/query

Tells the analyzer which HP-IB address is assigned to your external disk.

Command Syntax:

```
MMEMory:DISK:ADDRess <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:30
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Mmem:Disk:Addr 21"
OUTPUT 711;"MMEMORY:DISK:ADDR 6"
```

Query Syntax: `MMEMory:DISK:ADDRess?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

When you initiate an external disk operation with one of the MMEMory commands, the analyzer expects to find an external disk at the HP-IB address specified with MMEM:DISK:ADDR. If an external disk is not at the specified address, the operation is automatically aborted.

MMEMory:DISK:UNIT command/query

Specifies the unit of the external disk drive.

Command Syntax:

```
MMEMory:DISK:UNIT <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:10
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":mmemory:disk:unit 0"
OUTPUT 711;"Mmem:Disk:Unit 1"
```

Query Syntax: `MMEMory:DISK:UNIT?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

MMEMory:FSYStem? query

Returns the type of file system for the default disk.

Query Syntax: `MMEMory:FSYStem?`

Example Statements:

```
OUTPUT 711;"MMEM:FSYSTEM?"
OUTPUT 711;"mmem:fsys?"
```

Return Format: CHAR

```
CHAR ::= {LIF|DOS}
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the type of file system on the default disk. To specify the default disk, use the MMEMory:MSIS command.

MMEMory:INITialize **command**

Formats the specified disk.

Command Syntax:

```
MMEMory:INITialize [<MSINAME>], [{LIF|DOS}], [<format_op-
tion>], [<interleave_factor>]

          <MSINAME> ::= <disk>
             <disk> ::=NVRAM:|RAM:|INT:|EXT[,<select_code>[,<unit_number>]]:
      <select_code> ::=a real number (NRf data)
                        limits:  700:730
      <unit_number> ::=a real number (NRf data)
                        limits:  0:3
    <format_option> ::= <number>
           <number> ::= a real number (NRf data)
                        limits:  0:7000064
<interleave_factor> ::= <number>
           <number> ::= a real number (NRf data)
                        limits:  0:256
```

Example Statements:

```
OUTPUT 711;":Mmemory:Init 'INT:', LIF, 0, 3"
OUTPUT 711;"MMEMORY:INIT 'RAM:', DOS, 100000"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The select code and unit number specifiers are valid only with the EXT: disk specifier.

The first <number> specifies the format option. You can specify the format option for the internal disk (INT:) and an external disk (EXT:). In NVRAM, the format option is ignored. In RAM the memory size is already specified in bytes. Use the <format_option> field to specify memory size in bytes.

The <number> you enter after a floppy disk specifier is actually an encoded value that determines the disk's formatted capacity in kilobytes. See table 20-1.

The second <number> is the interleave factor. The default value is 0.

Note

When accessing the external mass storage device (EXT:); the active controller on the HP-IB must temporarily pass control to the analyzer. After the command has been executed, the analyzer must pass control back. See chapter 2, "Behavior in an HP-IB System," for more information on passing control.

Table 20-1. Flexible Disk Format Options

Media	Format Option	Bytes/Sector	Sectors/Track	Tracks/Surface	Maximum Capacity (bytes)
1-MByte	0	256	16	77	630,784
	1*	256	16	77	630,784
	2	512	9	77	709,632
	3	1,024	5	77	788,480
	4**	256	16	77	270,336
	16	512	9	80	737,280
	0	256	32	77	1,261,568
	1***	256	32	77	1,261,568
	2	512	18	77	1,419,264
2-MByte	3	1,024	10	77	1,576,960
	4***	256	32	77	1,261,568
	16	512	18	80	1,474,560

* Same as Option 0 (default) when using 1-MByte media.

** Not supported in internal disk drive (INT:).

*** Same as Option 0 (default) when using 2-MByte

MMEMory:LOAD:CFIT — command

Loads a curve fit table into the analyzer from a file on the specified disk.

Command Syntax:

```
MMEMory:LOAD:CFIT <FILENAME>

<FILENAME> ::= [<MSIS>]<filename>
    <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
<filename> ::= ASCII characters (see beginning of chapter
               for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"mmem:load:cfit 'CFFILE'"
OUTPUT 711;"Mmemory:Load:Cfit 'INT:CURVE1'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads a curve fit table into the curve fit buffer. The file *must have been saved* with the MMEM:STOR:CFIT or MMEM:STOR:SYNT commands. The file appears with the file extension ".FIT" or ".SYN".

If you are loading a synthesis table into the curve fit buffer (file extension is ".SYN"), the table must be in pole-zero format. See the CALC:SYNT:TTYP command for information about table formats.

The current curve fit table is overwritten.

Note

The file extensions ".FIT" and ".SYN" are naming conventions. The file type is created with the MMEM:STOR:CFIT or MMEM:STOR:SYNT commands. This file type data is embedded within the file.

MMEMory:LOAD:CONTinue command/query

Continues the load operation of time capture and waterfall files saved on multiple disks.

Command Syntax: `MMEMory:LOAD:CONTinue`

Example Statements:

```
OUTPUT 711;":MMEMORY:LOAD:CONTINUE"
OUTPUT 711;"mmem:load:continue"
```

Query Syntax: `MMEMory:LOAD:CONTinue?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads split files which were saved with the MMEM:STOR:CONT commands. MMEM:LOAD:CONT is valid only with time capture and waterfall files.

Use the MMEM:LOAD:TCAP or MMEM:LOAD:WAT command to begin the load operation.

When the analyzer has completed loading the first split file, filename_1, it generates a message, "Media full; Insert next disk with 'filename_2'. Insert the disk containing filename_2. Send this command to continuing loading the file.

You can use the MMEM:LOAD:CONT? to verify that the time capture or waterfall file has been transferred. If the MMEM:LOAD:CONT? query returns a +1, the analyzer has not completed the transfer. The query returns a 0 when the analyzer has transferred the entire time capture or waterfall file.

For additional information on the generation of split files, see the MMEM:STOR:CONT command.

MMEMory:LOAD:LIMit:LOWer:TRACe[1|2] command

Loads a lower limit for the specified trace from a file on the specified disk.

Command Syntax:

```
MMEMory:LOAD:LIMit:LOWer:TRACe[1|2] <FILENAME>

<FILENAME> ::= [<MSIS>]<filename>
    <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
<filename> ::= ASCII characters (see beginning of chapter
               for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"Mmem:Load:Lim:Low:Trace2 'limit1'"
OUTPUT 711;"MMEM:LOAD:LIM:LOW:TRACE 'INT:LIM2'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads the contents of a file into the lower limit register of the specified trace. The file must have been saved either with the MMEM:STOR:LIM:UPP, MMEM:LIM:LOW, or MMEM:STOR:TRAC command.

The first parameter selects the trace—TRAC1 for trace A or TRAC2 for trace B. Trace A is the default if you do not specify a trace. The second parameter specifies the source.

Additional limit commands are available under CALCulate:LIMit and DISPlay[:WINDow[1|2]]:LIMit.

MMEMory:LOAD:LIMit:UPPer:TRACe[1|2] command

Loads an upper limit for the specified from a file on the specified disk.

Command Syntax:

```
MMEMory:LOAD:LIMit:UPPer:TRACe[1|2] <FILENAME>

    <FILENAME> ::= [<MSIS>]<filename>
        <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
    <filename> ::= ASCII characters (see beginning of chapter for
                   <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":mmem:load:lim:upp:trace 'TRACE1'"
OUTPUT 711;"Mmem:Load:Lim:Upp:Trace2 'INT:MYTRACE'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads the contents of a file into the upper limit register of the specified trace. The file must have been saved either with the MMEM:STOR:LIM:UPP, MMEM:LIM:LOW, or MMEM:STOR:TRAC command.

The first parameter selects the trace—TRAC1 for trace A or TRAC2 for trace B. Trace A is the default if you do not specify a trace. The second parameter specifies the source.

Additional limit commands are available under CALCulate:LIMit and DISPlay[:WINDow[1|2]]:LIMit.

MMEMory:LOAD:MATH **command**

Loads a complete set of math definitions into the analyzer from the specified disk.

Command Syntax:

```
MMEMory:LOAD:MATH <FILENAME>

<FILENAME> ::= [<MSIS>]<filename>
    <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
<filename> ::= ASCII characters (see beginning of chapter
               for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"MMEM:LOAD:MATH 'EXT:MATHF1'"
OUTPUT 711;"mmem:load:math 'mymath'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command uses the contents of a file to the analyzer's function registers (F1 through F5) with math functions and the analyzer's constant registers (K1 through K5) with values. The file must have been saved with the MMEM:STOR:MATH command.

MMEMory:LOAD:PROGram command

Loads an HP Instrument BASIC program into the analyzer from a file on the specified disk.

Command Syntax:

```
MMEMory:LOAD:PROGram <FILENAME>

   <FILENAME> ::= [<MSIS>]<filename>
       <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
   <filename> ::= ASCII characters (see beginning of chapter
                  for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":Mmemory:Load:Prog 'INT:IBFILE'"
OUTPUT 711;"MMEMORY:LOAD:PROGRAM 'myprog'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads an HP Instrument BASIC program into the selected program buffer.

To specify the active program buffer, send the PROG:NAME command before sending this command. If a program buffer is not specified, the HP Instrument BASIC program loads into Program 1.

To load an HP Instrument BASIC program directly from your controller, use the PROGram[:SELected]:DEFine command.

MMEMory:LOAD:STATe command

Loads an instrument state into the analyzer from the specified disk.

Command Syntax:

```
MMEMory:LOAD:STATe <number>|<bound>, <FILENAME>

    <number> ::= a real number (NRf data)
                 limits:  1:1
     <bound> ::= MAX|MIN
  <FILENAME> ::= [<MSIS>]<filename>
      <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
  <filename> ::= ASCII characters (see beginning of chapter
                 for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"mmem:load:state 1, 'INT:STATE.STA'"
OUTPUT 711;"Mmem:Load:Stat 1, 'ext:mystate'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command uses the contents of a file to redefine the instrument state. The file must have been saved with the MMEM:STOR:STAT command.

MMEMory:LOAD:SYNThesis command

Loads a synthesis table into the analyzer from a file on the specified disk.

Command Syntax: `MMEMory:LOAD:SYNThesis <FILENAME>`

```
<FILENAME> ::= [<MSIS>]<filename>
    <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
<filename> ::= ASCII characters (see beginning of chapter
               for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":MMEM:LOAD:SYNT 'SYFILE'"
OUTPUT 711;"mmemory:load:synt 'int:file1'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads a synthesis table into the curve fit buffer. The file *must have been saved* with the MMEM:STOR:CFIT or MMEM:STOR:SYNT commands. The file appears with the file extension ".FIT" or ".SYN".

The current synthesis table is overwritten.

Note

The file extensions ".FIT" and ".SYN" are naming conventions. The file type is created with the MMEM:STOR:CFIT or MMEM:STOR:SYNT commands. This file type data is embedded within the file.

MMEMory:LOAD:TCAPture command

Loads a time capture file from the specified disk.

Command Syntax:

```
MMEMory:LOAD:TCAPture <FILENAME>

    <FILENAME> ::= [<MSIS>]<filename>
        <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
    <filename> ::= ASCII characters (see beginning of chapter
                   for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"Mmemory:Load:Tcapture 'int:tcapl'"
OUTPUT 711;"MMEM:LOAD:TCAPTURE 'MYTIME'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads the contents of a file into the time capture buffer. The file must have been saved with the MMEM:STOR:TCAP command. To display the time capture buffer use the CALC:FEED 'TCAP [1|2]' command.

Use the MMEM:LOAD:CONT if the file is saved on multiple disks. See the MMEM:LOAD:CONT command for more information about loading split files on multiple disks.

MMEMory:LOAD:TRACe command

Loads a trace into the analyzer from the specified disk.

Command Syntax:

```
MMEMory:LOAD:TRACe {D1|D2|D3|D4|D5|D6|D7|D8}, <FILENAME>

        <FILENAME> ::= [<MSIS>]<filename>
            <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
        <filename> ::= ASCII characters (see beginning of chapter
                       for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":mmem:load:trac D4, 'INT:TESTTR"
OUTPUT 711;"Mmem:Load:Trac D5, 'mytrace'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads the contents of a file into one of the analyzer's eight data registers (D1 through D8). The first parameter specifies the destination. The second parameter specifies the source.

The file must have been saved with the MMEM:STOR:TRAC command. After loading the data register you can display its contents with the CALC:FEED 'D{1|2... |8}' command.

MMEMory:LOAD:WATerfall command

Loads a waterfall file into the analyzer from a file on the specified disk.

Command Syntax:

```
MMEMory:LOAD:WATerfall (W1|W2|W3|W4|W5|W6|W7|W8),
<FILENAME>

    <FILENAME> ::= [<MSIS>]<filename>
        <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
    <filename> ::= ASCII characters (see beginning of chapter
                   for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"MMEMORY:LOAD:WAT W8, 'INT:MYWAT'"
OUTPUT 711;"mmemory:load:waterfall W3, 'testwat'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads the contents of a file into one of the analyzer's eight waterfall registers (W1 through W8). The file must have been saved with the MMEM:STOR:WAT command.

After loading the data register you can display its contents with the CALC:FEED 'W{1|2... |8}' command.

Use the MMEM:LOAD:CONT if the file is saved on multiple disks. See the MMEM:LOAD:CONT command for more information about loading split files on multiple disks.

MMEMory:MSIS — command/query

Specifies a default disk.

Command Syntax:

```
MMEMory:MSIS <MSINAME>

     <MSINAME> ::- <disk>
        <disk> ::-NVRAM:|RAM:|INT:|EXT[,<select_code>[,<unit_number>]]:
```

Example Statements:

```
OUTPUT 711;":Mmem:Msis 'INT:'"
OUTPUT 711;"MMEMORY:MSIS 'RAM:'"
```

Query Syntax:

```
MMEMory:MSIS?
```

Return Format:

```
STRING
```

Attribute Summary:

```
Option:  not applicable
Preset State:  saved in non-volatile memory
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If you omit disk specifiers from MMEMory commands, the commands are automatically directed to the default disk. This command uses the following mnemonics to select the default disk:

- NVRAM: — selects the non-volatile RAM disk.
- RAM: — selects the volatile RAM disk.
- INT: — selects the internal disk.
- EXT: — selects the external disk. The select code and unit number specifiers are valid only with the EXT: disk specifier. If the select code and unit number are specified with this command, they become the *default* for all external disks (EXT:) specifiers.

To determine the type of file system for the default disk, send the MMEM:FSYStem? query.

Note

When accessing the external mass storage device (EXT:); the active controller on the HP-IB must temporarily pass control to the analyzer. After the command has been executed, the analyzer must pass control back. See chapter 2, "Behavior in an HP-IB System," for more information on passing control.

MMEMory:REName command

Renames a file.

Command Syntax:

```
MMEMory:REName <FILENAME>, <FILE>

     <FILENAME> ::= [<MSIS>]<filename>
         <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
     <filename> ::= ASCII characters (see beginning of chapter
                    for <filename> restrictions)
         <FILE> ::= ASCII characters (see beginning of chapter
                    for <file> restrictions)
```

Example Statements:

```
OUTPUT 711;"mmemory:ren 'INT:FILE1', 'MYFILE'"
OUTPUT 711;"Mmem:Rename 'testfile', 'file3'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The <filename> is the old file name; <file> is the new file name.

Renaming only allows you to change a file's name on the current disk. It does not allow you to *move* a file by changing the file's name and disk specifier. To move a file, first copy it to another disk with the MMEM:COPY command, then delete it from the original disk with the MMEM:DEL command.

Note When accessing the external mass storage device (EXT:); the active controller on the HP-IB must temporarily pass control to the analyzer. After the command has been executed, the analyzer must pass control back. See chapter 2, "Behavior in an HP-IB System," for more information on passing control.

MMEMory:STORe:CFIT command

Stores a curve fit table to a file on the specified disk.

Command Syntax: `MMEMory:STORe:CFIT <FILENAME>`

```
<FILENAME> ::= [<MSIS>]<filename>
    <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
<filename> ::= ASCII characters (see beginning of chapter
               for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":MMEM:STORE:CFIT 'CFFILE'"
OUTPUT 711;"mmem:store:cfit 'int:newcve'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command stores the current curve fit table to a file. The disk catalog identifies the file type with the extension, ".FIT". See the DISP:CONT MMEM command for information about displaying the disk catalog.

If the file name matches the name of another file on the disk, this command overwrites the old file.

MMEMory:STORe:CONTinue command/query

Splits a large time capture file or a waterfall file over multiple disks.

Command Syntax: `MMEMory:STORe:CONTinue`

Example Statements:

```
OUTPUT 711;"Mmemory:Stor:Cont"
OUTPUT 711;"MMEMORY:STOR:CONTINUE"
```

Query Syntax: `MMEMory:STORe:CONTinue?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command splits files created by the MMEM:STOR:WAT and the MMEM:STOR:TCAP commands only.

If the buffer is too large for the disk when you send the MMEM:STOR:WAT or MMEM:STOR:TCAP command, an error, "Media full; File too large" is generated. Send the MMEM:STOR:CONT command to begin the save operation. (The error message is generated before the save operation is implemented).

MMEM:STOR:CONT adds a numeric specifier to the <filename>. Split files appear as filename_1, filename_2, etc. in the disk catalog (see DISP:CONT MMEM).

MMEM:STORE CONT? returns a +1 if the analyzer has not completely saved the time capture file or waterfall file. Insert the new disk and send the MMEM:STORE:CONT command to continue the save operation. The query returns a 0 when the entire time capture or waterfall has been saved.

MMEMory:STORe:LIMit:LOWer:TRACe[1|2] command

Saves the lower limit of the specified trace to a file on the specified disk.

Command Syntax:

```
MMEMory:STORe:LIMit:LOWer:TRACe[1|2] <FILENAME>

    <FILENAME> ::= [<MSIS>]<filename>
        <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
    <filename> ::= ASCII characters (see beginning of chapter
                   for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":mmem:stor:limit:low:trace2 'int:lim2'"
OUTPUT 711;"Mmem:Stor:Limit:Low:Trace 'newlim.lim'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command saves a lower limit to a file. The first parameter selects which lower limit you are saving—TRAC1 for trace A or TRAC2 for trace B. Trace A is the default if you do not specify a trace.

If the file name you specify matches the name of another file on the disk, this command overwrites the old file.

Note

If you plan to transfer this file to a PC, refer to the *Standard Data Format Utilities Users Guide.*

MMEMory:STORe:LIMit:UPPer:TRACe[1|2] command

Saves the upper limit of the specified trace to a file on the specified disk.

Command Syntax:

```
MMEMory:STORe:LIMit:UPPer:TRACe[1|2] <FILENAME>

<FILENAME> ::= [<MSIS>]<filename>
    <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
<filename> ::= ASCII characters (see beginning of chapter
               for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"MMEM:STOR:LIMIT:UPP:TRACE2 'LIMIT2.LIM'"
OUTPUT 711;"mmem:stor:limit:upp:trace2 'int:newlim'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command saves an upper limit to a file. The first parameter selects which upper limit you are saving—TRAC1 for trace A or TRAC2 for trace B. Trace A is the default if you do not specify a trace.

If the file name you specify matches the name of another file on the disk, this command overwrites the old file.

Note If you plan to transfer this file to a PC, refer to the *Standard Data Format Utilities Users Guide.*

MMEMory:STORe:MATH command

Saves a complete set of math definitions to a file on the specified disk.

Command Syntax: `MMEMory:STORe:MATH <FILENAME>`

```
<FILENAME> ::= [<MSIS>]<filename>
    <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
<filename> ::= ASCII characters (see beginning of chapter
               for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":Mmem:Stor:Math 'mymath.def'"
OUTPUT 711;"MMEM:STORE:MATH 'EXT:NEWMATH'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command saves the math functions in the analyzer's five function registers (F1 through F5) and the current values in the analyzer's constant registers (K1 through K5) to a file.

If the filename you specify matches the name of another file on the disk, this command overwrites the old file.

MMEMory:STORe:PROGram **command**

Saves an HP Instrument BASIC program to a file on the specified disk for the first time.

Command Syntax: `MMEMory:STORe:PROGram <FILENAME>`

```
     <FILENAME> ::= [<MSIS>]<filename>
         <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
     <filename> ::= ASCII characters (see beginning of chapter
                    for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"mmem:store:prog 'int:myprog'"
OUTPUT 711;"Mmemory:Stor:Prog 'IBPROG'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command saves the currently active HP Instrument Basic program to the specified disk. The program must be located in the active program buffer (see the PROG:NAME command). If the active program buffer does not contain a program the analyzer generates an error, "Program Error, No program exists."

If the filename you specify matches the name of another file on the disk, the analyzer generates an error, "Program Error, Duplicate file name." It aborts the save operation. This command will not overwrite an existing file. See the MMEM:DEL command for information about deleting existing files from a disk.

MMEMory:STORe:STATe command

Saves the instrument state to a file on the specified disk.

Command Syntax:

```
MMEMory:STORe:STATe <number>|<bound>, <FILENAME>

     <number> ::= a real number (NRf data)
                  limits:  1:1
      <bound> ::= MAX|MIN
   <FILENAME> ::= [<MSIS>]<filename>
       <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
   <filename> ::= ASCII characters (see beginning of chapter
                  for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":MMEMORY:STOR:STATE 1, 'EXT:STATE.STA'"
OUTPUT 711;"mmem:stor:state 1, 'NEWSTATE'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance: confirmed
```

Description:

If the filename you specify matches the name of another file on the disk, this command overwrites the old file.

MMEMory:STORe:SYNThesis command

Stores a synthesis table to a file on the specified disk.

Command Syntax:

```
MMEMory:STORe:SYNThesis <FILENAME>

<FILENAME> ::= [<MSIS>]<filename>
    <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
<filename> ::= ASCII characters (see beginning of chapter
               for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"Mmem:Store:Synt 'INT:SYFILE'"
OUTPUT 711;"MMEM:STORE:SYNT 'NEWSYNTH'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command stores the current synthesis table to a file. The disk catalog identifies the file type with the extension, ".SYN". See the DISP:CONT MMEM command for information about displaying the disk catalog.

If the file name matches the name of another file on the disk, this command overwrites the old file.

MMEMory:STORe:TCAPture **command**

Saves the time capture buffer to a file on the specified disk.

Command Syntax:

```
MMEMory:STORe:TCAPture <FILENAME>

 <FILENAME> ::= [<MSIS>]<filename>
     <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
 <filename> ::= ASCII characters (see beginning of chapter
                for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":mmemory:stor:tcap 'EXT:TCAP1'"
OUTPUT 711;"Mmemory:Stor:Tcapture 'Mytcap'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command saves the current time capture buffer to a file on the specified disk.

If the buffer is too large for the disk, an error, “Media full; File too Large” is generated. Use the MMEM:STORE CONT command to split the file over multiple disks. See the MMEM:STORE CONT command for more information about splitting a time capture file on multiple disks.

If the filename you specify matches the name of another file on the disk, this command overwrites the old file.

Note If you plan to transfer this file to a PC, refer to the *Standard Data Format Utilities Users Guide.*

MMEMory:STORe:TRACe command

Saves the specified trace to a file on the specified disk.

Command Syntax:

```
MMEMory:STORe:TRACe {TRACe1|TRACe2}, <FILENAME>

<FILENAME> ::= [<MSIS>]<filename>
    <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
<filename> ::= ASCII characters (see beginning of chapter
               for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;"MMEM:STOR:TRACE TRACe1, 'NEW.TRC'"
OUTPUT 711;"mmem:store:trac TRACe1, 'INT:MYTRAC'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance: confirmed
```

Description:

The first parameter specifies which trace you are saving—TRAC1 for the trace appearing in trace box A or TRAC2 for the trace appearing in trace box B. The second parameter specifies the disk and filename. If the disk is not specified, the file is saved to the default disk.

If the filename you specify matches the name of another file on the disk, this command overwrites the old file.

This command differs from the TRAC:DATA command as the MMEM:STOR:TRAC command only saves trace data to a file. TRAC:DATA saves trace data to one of the data registers.

Note If you plan to transfer this file to a PC, refer to the *Standard Data Format Utilities Users Guide*.

MMEMory:STORe:WATerfall **command**

Saves the current waterfall display to a file on the specified disk.

Command Syntax:

```
MMEMory:STORe:WATerfall {TRACe1|TRACe2}, <FILENAME>

     <FILENAME> ::= [<MSIS>]<filename>
         <MSIS> ::= RAM:|NVRAM:|INT:|EXT:
     <filename> ::= ASCII characters (see beginning of chapter
                    for <filename> restrictions)
```

Example Statements:

```
OUTPUT 711;":Mmem:Store:Wat TRACe1, 'LASTWAT'"
OUTPUT 711;"MMEMORY:STOR:WAT TRACe1, 'INT:MYWAT'"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command saves a waterfall display to a file. The waterfall buffer *must* contain more than one trace.

The first parameter specifies which trace you are saving—TRAC1 for the trace appearing in trace box A or TRAC2 for the waterfall appearing in trace box B. The second parameter specifies the disk and filename. If the disk is not specified, the file is saved to the default disk.

If the filename you specify matches the name of another file on the disk, this command overwrites the old file.

If the waterfall display is too large for the disk, an error, "Media full; File too large". Use the MMEM:STORE CONT command to split the file over multiple disks. See the MMEM:STORE CONT command for more information about splitting a time capture file on multiple disks.

Note If you plan to transfer this file to a PC, refer to the *Standard Data Format Utilities Users Guide.*

21

OUTPut

This subsystem contains one command which enables and disables the analyzer's source output. See the SOURce subsystem for commands which define the analyzer's source output.

OUTPut[:STATe] command/query

Enables the analyzer's internal source.

Command Syntax:

```
OUTPut[:STATe] {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"output:stat ON"
OUTPUT 711;"Output OFF"
```

Query Syntax:

```
OUTPut[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

This command is not available for the swept sine analysis instrument mode (INST:SEL SINE).

22

PROGram

The commands in this subsystem are only available when the HP Instrument BASIC option is installed (Option 1C2). The commands in the PROGram subsystem allow you to generate and control HP Instrument BASIC programs in the analyzer.

The commands grouped under the SELected mnemonic operate on the active program buffer. Since SELected is an implied mnemonic, you can omit it from the PROGram commands. See "Implied Mnemonics" in chapter 3 for more information.

The command under the EXPLicit mnemonic operates on any one of the analyzer's five program buffer—not just the active program buffer.

PROGram:EXPLicit:DEFine command/query

Loads an HP Instrument BASIC program into the specified program buffer from an external controller.

Command Syntax:

```
PROGram:EXPLicit:DEFine
{PROGram1|PROGram2|PROGram3|PROGram4|PROGram5}, <PROGRAM>

<PROGRAM> ::= <BLOCK>
```

Examples:

```
 Indefinite Block

RSUBTXT2 =  OUTPUT 711;":PROG:DEF #0";
OUTPUT 711;"10 PRINT ""HELLO WORLD"""&CHR$(10);
OUTPUT 711;"20 END"&CHR$(10);
OUTPUT 711;CHR$(10) END

 Definite Block

OUTPUT 711;":PROG:DEF #230";
OUTPUT 711;"10 PRINT ""HELLO WORLD"""&CHR$(10);
OUTPUT 711;"20 END"&CHR$(10);
```

Query Syntax: `PROGram:EXPLicit:DEFine?`

Return Format: `definite length <BLOCK>`

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command transfers a program between the analyzer and your controller. This allows you to develop a program on your controller and then load it into the analyzer. The first parameter specifies the program buffer. This becomes the active program buffer. The second parameter is the HP Instrument BASIC program.

Two example programs in chapter 6, Download and Upload, demonstrate transferring a program file between the analyzer and an external controller.

When you transfer a program to the analyzer, you can use either the definite or the indefinite length block syntax. The simplest way to load an HP Instrument BASIC program into the analyzer is to send this command followed by #0, followed by all the characters making up the program (including line numbers and line feeds at the end of each program statement). Terminate the entire command with line feed character (ASCII decimal 10) and < ^ END> (the HP-IB END message, EOI set true).

When the analyzer returns the program to your controller, it always uses the definite length block syntax. See "Block Data" in chapter 4 for more information.

PROGram[:SELected]:DEFine command/query

Loads an HP Instrument BASIC program from an external controller into the active program buffer.

Command Syntax: `PROGram[:SELected]:DEFine <PROGRAM>`

```
<PROGRAM> ::= <BLOCK>
```

Examples:

```
 Indefinite Block

OUTPUT 711;":PROG:DEF #0";
OUTPUT 711;"10 PRINT ""HELLO WORLD"""&CHR$(10);
OUTPUT 711;"20 END"&CHR$(10);
OUTPUT 711;CHR$(10) END

 Definite Block

OUTPUT 711;":PROG:DEF #230";
OUTPUT 711;"10 PRINT ""HELLO WORLD"""&CHR$(10);
OUTPUT 711;"20 END"&CHR$(10);
```

Query Syntax: `PROGram[:SELected]:DEFine?`

Return Format: `definite length <BLOCK>`

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command transfers a program between the analyzer and your controller. This allows you to develop a program on your controller and then load it into the analyzer.

Two example programs in chapter 6, Download and Upload, demonstrate transferring a program file between the analyzer and an external controller.

Use the PROG[:SEL]:NAME to select the active program buffer.

When you transfer a program to the analyzer, you can use either the definite or the indefinite length block syntax. The *simplest* way to load an HP Instrument BASIC program into the analyzer is to send this command followed by #0, followed by all the characters making up the program (including line numbers and line feeds at the end of each program statement). Terminate the entire command with line feed character (ASCII decimal 10) and < ^ END> (the HP-IB END message, EOI set true).

When the analyzer returns the program to your controller, it always uses the definite length block syntax. See "Block Data" in chapter 4 for more information.

PROGram[:SELected]:DELete:ALL command

Deletes all HP Instrument BASIC programs stored in the analyzer.

Command Syntax: `PROGram[:SELected]:DELete:ALL`

Example Statements:

```
OUTPUT 711;":prog:delete:all"
OUTPUT 711;"Program:Sel:Del:All"
```

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  not applicable
SCPI Compliance: confirmed
```

Description:

In addition to deleting the active program, this command deletes *all* of the resident HP Instrument BASIC programs. Program variables—both those in COM and those not in COM are deleted as well.

This is equivalent to a "Scratch A" operation.

PROGram[:SELected]:DELete[:SELected] command

Deletes the active HP Instrument BASIC program.

Command Syntax:

```
PROGram[:SELected]:DELete[:SELected]
```

Example Statements:

```
OUTPUT 711;"PROG:SELECTED:DEL:SEL"
OUTPUT 711;"program:del"
```

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

In addition to deleting the active program, this command deletes all of the program variables—both those in COM and those not in COM. Specify the active program with the PROG:NAME command.

This is equivalent to a "Scratch A" operation.

PROGram[:SELected]:LABel command/query

Loads a softkey label for the active HP Instrument BASIC program.

Command Syntax:

```
PROGram[:SELected]:LABel <STRING>

<STRING> ::= maximum of 18 ASCII characters
                (2 lines of 9 characters)
```

Example Statements:

```
OUTPUT 711;":Program:Lab '  START     TEST'"
OUTPUT 711;"PROG:SELECTED:LAB '  PRINT    REPORT'"
```

Query Syntax: `PROGram[:SELected]:LABel?`

Return Format: `STRING`

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command allows you to customize the front panel softkey labels for HP Instrument BASIC programs.

Specify the active program with the PROG:NAME command.

PROGram[:SELected]:MALLocate command/query

Allocates memory space for HP Instrument BASIC programs.

Command Syntax:

```
PROGram[:SELected]:MALLocate (<number>|<bound>|DEFault)

     <number> ::= a real number (NRf data)
                    limits:  1200:500000
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"program:sel:mall 134987"
OUTPUT 711;"Program:Mall 250982"
```

Query Syntax:

```
PROGram[:SELected]:MALLocate?
```

Return Format: NR1

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

If you send PROG:MALL DEF, the analyzer resizes the stack space to fit the current active program. In some cases, the analyzer may allocate more memory than the HP Instrument BASIC program needs.

Note

You need to allocate more memory if you encounter the message, "ERROR 2 Memory overflow" while your program is running

PROGram[:SELected]:NAME command/query

Selects an HP Instrument BASIC program.

Command Syntax:

```
PROGram[:SELected]:NAME
{PROGram1|PROGram2|PROGram3|PROGram4|PROGram5}
```

Example Statements:

```
OUTPUT 711;":PROGRAM:NAME PROGram1"
OUTPUT 711;"prog:selected:name PROGram5"
```

Query Syntax:

```
PROGram[:SELected]:NAME?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Use this command to designate an HP Instrument BASIC program buffer as the "active" program buffer.

For example, use this command to select a program buffer when you load an HP Instrument BASIC program into the analyzer with the PROG:DEF command.

PROGram[:SELected]:NUMBer command/query

Loads a new value for the specified numeric variable in the active HP Instrument BASIC program.

Command Syntax:

```
PROGram[:SELected]:NUMBer '<VARIABLE>', <BLOCK>

          <VARIABLE> ::= name of a numeric variable
```

When data is ASCII-encoded, (FORM ASC) <BLOCK> takes the following form:

```
             <BLOCK> ::= <number>[,<number>] . . .
            <number> ::= a real number
                         (NRf data)
                         limits:  - 9.9e37:9.9e37
```

When data is binary-encoded, (FORM REAL) <BLOCK> takes the following form:

```
             <BLOCK> ::= #<byte>[<length_bytes>]<number>[,<number>] . . .
              <byte> ::= number of length bytes to follow (ASCII encoded)
      <length_bytes> ::  =number of data bytes to follow (ASCII encoded)
            <number> ::= a real number (32- or 64-bit binary floating point)
```

Example Statements:

```
OUTPUT 711;"Program:Sel:Numb 'Address',11"
OUTPUT 711;"PROGRAM:NUMB 'Scode',7"
```

Query Syntax:

```
PROGram[:SELected]:NUMBer?
```

Return Format:

```
definite length <BLOCK>
```

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

When you load an array with this command, values in the <BLOCK> parameter are loaded into the 1^{st} through n^{th} elements of the array (where *n* is number of values in the block).

The analyzer uses the format specified by the FORMat[:DATA] command for query responses. The analyzer generates an error if the specified variable is not defined in the active program. Use the PROG:NAME command to specify the active program.

Use the PROG:STR command to load string variables.

PROGram[:SELected]:STATe command/query

Selects the state of the active HP Instrument BASIC program.

Command Syntax: `PROGram[:SELected]:STATe {STOP|PAUSe|RUN|CONTinue}`

Example Statements:

```
OUTPUT 711;":program:stat RUN"
OUTPUT 711;"Prog:Selected:Stat CONTinue"
```

Query Syntax: `PROGram[:SELected]:STATe?`

Return Format: CHAR

Attribute Summary:

```
Option:  1C2 HP Instrument BASIC
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This command allows you to run, pause, stop or continue the active HP Instrument BASIC program.

The analyzer generates an error message, "Settings conflict; Invalid program state change requested," if you send RUN or CONT while a program is running. It also generates the error if you send CONT while a program is stopped.

Use the PROG:NAME command to select the active program.

PROGram[:SELected]:STRing command/query

Loads a new value for the specified string variable for the active HP Instrument BASIC program.

Command Syntax:

```
PROGram[:SELected]:STRing '<VARIABLE>','<STRING>'

  <VARIABLE> ::= name of string variable
                 (mandatory $ at the end of the name)
    <STRING> ::= ASCII characters - 0 through 255
                 maximum number of characters: 32766
```

Example Statements:

```
OUTPUT 711;"PROGRAM:SEL:STR 'A$','Done'"
OUTPUT 711;"program:str 'Message$','Measuring'"
```

Query Syntax: `PROGram[:SELected]:STRing?`

Return Format: `<STRING>`

Attribute Summary:

```
Option: 1C2 HP Instrument BASIC
Overlapped: no
Preset State: not applicable
SCPI Compliance: confirmed
```

Description:

This command sets or queries the contents of a string variable in the active HP Instrument BASIC program. Use the PROG:NAME to designate the active program.

Use the PROG[:SEL]:NUMB command to load or query numeric variables.

23

[SENSe:]

Commands in this subsystem determine how measurement data is acquired.

Commands grouped under the AVERage mnemonic define how the results of several measurements will be combined in one trace.

Commands grouped under the FREQuency mnemonic control the frequency characteristics of the analyzer.

Commands grouped under the TCAPture mnemonic define the time capture paramteters.

Commands grouped under the SWEep mnemonic define parameters for swept sine measurements.

Commands grouped under the VOLTage subsection control the amplitude characteristics of the input channels.

Commands under the WINDow mnemonic define windowing parameters.

SENSe is an implied mnemonic. Therefore, you can omit it from all SENSe commands. See "Implied Mnemonics" in chapter 3 for more information.

[SENSe:]AVERage:CONFidence command/query

Specifies the confidence level used in equal confidence averaging in octave measurements.

Command Syntax:

```
[SENSe:]AVERage:CONFidence (<num-
ber>[<unit>])|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  0.25:2.0
       <unit> ::= [DB]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Average:Conf 0.5"
OUTPUT 711;"SENS:AVERAGE:CONF 2"
```

Query Syntax:

```
[SENSe:]AVERage:CONFidence?
```

Return Format: NR3

Attribute Summary:

```
Option:  1D1 Realtime Octave
Overlapped:  no
Preset State:  +5.0E-001
SCPI Compliance:  instrument-specific
```

Description:

The analyzer varies the average time constant to provide a 68% probability that the measured results are within ± σ of the true mean value. There is a 95% probability that the results are within ± 2σ of the true value.

There are four values of σ available: .25 dB, .5 dB, 1 dB and 2 dB.

[SENSe:]AVERage:COUNt command/query

Specifies a count or a weighting factor for the averaged measurement data.

Command Syntax:

```
[SENSe:]AVERage:COUNt (<number>[<unit>])|<step>|<bound>

          <number> ::= a real number (NRf data)
                         limits:  1:9999999
            <unit> ::=
            <step> ::= UP|DOWN
           <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"sense:aver:coun 7299573"
OUTPUT 711;"Average:Coun 8425657"
```

Query Syntax: `[SENSe:]AVERage:COUNt?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +10
SCPI Compliance:  instrument-specific
```

Description:

As a counter, AVER:COUN determines the number of time records used to average the measurement data.. This command is only valid in FFT, order analysis and correlation instrument modes.

Once the specified number of time records have been averaged, the No Pending Operation (NPO) flag is set to 1. Use the *OPC command to determine when the specified number of time records have been combined. (See “Synchronization” in chapter 2 for more information on the NPO flag.)

When used with exponential averaging (AVER:TYPE RMS;TCON EXP or AVER:TYPE VECT;TCON EXP), this command determines how the results of the current measurement (new data) is combined with the averaged trace (old data). Data is combined, point by point, according to the following formula:

$$\left(\frac{1}{n}\,\text{new}\right) + \left(\frac{n-1}{n}\,\text{old}\right)$$

[SENSe:]AVERage:HOLD command/query

Specifies the type of hold used in averaging octave measurements.

Command Syntax:

```
[SENSe:]AVERage:HOLD (OFF|0|MAXimum|MINimum)
```

Example Statements:

```
OUTPUT 711;":AVERAGE:HOLD MAXimum"
OUTPUT 711;"sens:average:hold MINimum"
```

Query Syntax:

```
[SENSe:]AVERage:HOLD?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  1D1 Realtime Octave
Overlapped:  no
Preset State:  OFF
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the type of average hold used for octave measurements. It is valid when used with:

- Linear averaging (AVER:TYPE RMS).
- Exponential averaging (AVER:TYPE RMS;TCON EXP).
- Equal confidence averaging (AVER:TYPE ECON).

If you send AVER:HOLD MAX, the analyzer displays the maximum averaged spectrum value for each band.

If you send AVER:HOLD MIN, the analyzer displays the minimum averaged spectrum value for each band. This is useful for estimating background noise.

Note AVERage:HOLD differs from the peak hold function. See the AVER:TYPE MAX command description for information about the peak hold function.

[SENSe:]AVERage:IMPulse command/query

Enables impulse detection in octave measurements.

Command Syntax:

```
[SENSe:]AVERage:IMPulse (OFF|0|ON|1)
```

Example Statements:

```
OUTPUT 711;"Sense:Aver:Imp OFF"
OUTPUT 711;"AVERAGE:IMP ON"
```

Query Syntax:

```
[SENSe:]AVERage:IMPulse?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D1 Realtime Octave
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

When impulse detection is on (AVER:IMP ON), the analyzer computes and displays the IEC 651 impulse characteristics in the overall power band.

In linear averaging (AVER:TYPE RMS;TCON FRE), the analyzer calculates the value of the impulse output over the average time (specified with the AVER:TIME command).

In all other types of averaging, the analyzer calculates the instantaneous value of the impulse vector.

See the analyzer's online Help for additional information.

[SENSe:]AVERage:IRESult:RATE command/query

Specifies how often the display is updated when fast average mode is on.

Command Syntax:

```
[SENSe:]AVERage:IRESult:RATE (<number>[<unit>])|<step>|<bound>

   <number> ::= a real number (NRf data)
                limits:  1:99999
     <unit> ::=
     <step> ::= UP|DOWN
    <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":average:ires:rate 2939"
OUTPUT 711;"Sense:Aver:Iresult:Rate 60760"
```

Query Syntax:

```
[SENSe:]AVERage:IRESult:RATE?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +5
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the rate used by the analyzer when AVER:IRES is on.

The analyzer updates the display once for each N averages. N is the update rate specified with this command. The analyzer continues to update the display whenever it reaches a multiple of N.

[SENSe:]AVERage:IRESult[:STATe] command/query

Selects fast average mode.

Command Syntax:

```
[SENSe:]AVERage:IRESult[:STATe] (OFF|0|ON|1)
```

Example Statements:

```
OUTPUT 711;"SENS:AVERAGE:IRES:STATE OFF"
OUTPUT 711;"aver:ires ON"
```

Query Syntax:

```
[SENSe:]AVERage:IRESult[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

This command specifies whether the analyzer displays is in fast average mode.

In **FFT and correlation instrument mode**, AVER:IRES ON updates the display once for every N averages. N is the update rate specified with AVER:IRES:RATE. The preset AVER:IRES rate is 5. OFF updates the display after each average.

In **swept sine instrument mode**, AVER:IRES ON updates the display after the entire sweep is completed. OFF updates the display at each point in the sweep.

In **histogram instrument mode**, AVER:IRES ON updates the display as fast as it can without slowing down the measurement. OFF updates the display at the end of the measurement.

[SENSe:]AVERage:PREView command/query

Specifies the type of preview averaging.

Command Syntax:

```
[SENSe:]AVERage:PREView {OFF|0|MANual|TIMed}
```

Example Statements:

```
OUTPUT 711;":Average:Prev MANual"
OUTPUT 711;"SENSE:AVER:PREV TIMed"
```

Query Syntax:

```
[SENSe:]AVERage:PREView?
```

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  OFF
SCPI Compliance:  instrument-specific
```

Description:

This command is valid in FFT analysis mode only (INST:SEL FFT).

To enable manual previewing, send AVER:PREV MAN. The analyzer waits for a response before taking the next time record.

To enable timed previewing, send AVER:PREV TIM. The analyzer waits for a specified amount of time before accepting the time record. That is, if no response is sent the analyzer accepts the time record. Set the time period with the AVER:PREV:TIME command.

After each time record is collected, the Waiting for Accept/Reject bit in the Operational Status Register is set to 1. The bit is cleared when the analyzer receives an accept or reject command or when the analyzer receives a command that changes the measurement setup.

To accept the time record send AVER:PREV:ACC. To reject the time record send AVER:PREV:REJ. To turn off average previewing, send AVER:PREV OFF.

[SENSe:]AVERage:PREView:ACCept command

Accept the current time record during preview averaging.

Command Syntax: `[SENSe:]AVERage:PREView:ACCept`

Example Statements:

```
OUTPUT 711;"sense:aver:preview:acc"
OUTPUT 711;"Aver:Preview:Acc"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

[SENSe:]AVERage:PREView:REJect command

Reject the current time record during preview averaging.

Command Syntax: `[SENSe:]AVERage:PREView:REJect`

Example Statements:

```
OUTPUT 711;":AVERAGE:PREV:REJ"
OUTPUT 711;"sense:aver:preview:rej"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

[SENSe:]AVERage:PREView:TIME command/query

Specifies the amount of time the analyzer waits for a response in timed preview averaging.

Command Syntax:

```
[SENSe:]AVERage:PREView:TIME (<number>[<unit>])|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0.1:3600.0
  <unit> ::= [S]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Sens:Average:Prev:Time 1188.54"
OUTPUT 711;"AVER:PREV:TIME 1119.08"
```

Query Syntax: `[SENSe:]AVERage:PREView:TIME?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1.0E001
SCPI Compliance:  instrument-specific
```

Description:

The analyzer waits the specified amount of time for a response. If a response is *not* sent, the analyzer *accepts* the time record.

Time is specified in seconds. Specify timed preview averaging with the AVER:PREV:TIM command.

[SENSe:]AVERage[:STATe] command/query

Turns the selected averaging function (AVER:TYPE) on or off.

Command Syntax:

```
[SENSe:]AVERage[:STATe] {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":aver ON"
OUTPUT 711;"Sense:Aver:Stat ON"
```

Query Syntax:

```
[SENSe:]AVERage[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

When you select ON, each trace represents the combined results of several measurements, and the averaging function specified in AVER:TYPE determines how results are combined.

RMS averaging (AVER:TYPE RMS) provides a better estimate of the noise in measurement data. Vector averaging (AVER:TYPE VECT) reduces the amount of random noise and provides a better estimate of the repetitive signals in the measurement data. Maximum averaging (AVER:TYPE MAX) saves the maximum power value (power spectra) for each frequency bin.

When you select OFF, each trace represents the results of a single measurement. It is mathematically equivalent to exponential RMS (power) averaging with 1 average.

When averaging is ON and AVER:TYPE is MAXimum, RMS or VECTor, INIT:IMM sets the No Pending Operation (NPO) flag to 1 *after* the specified number (set with the AVER:COUN command) of measurement results have been combined. When averaging is ON and termination control is exponential (AVER:TCON EXPO), INIT:IMM sets the NPO flag to 1 each time a measurement is completed, *after* the initial N averages. When averaging is OFF, INIT:IMM sets the NPO flag to 1 each time a measurement is completed. It acts as if the average count is set to 1. See "Synchronization" in chapter 2 for more information on the NPO flag.

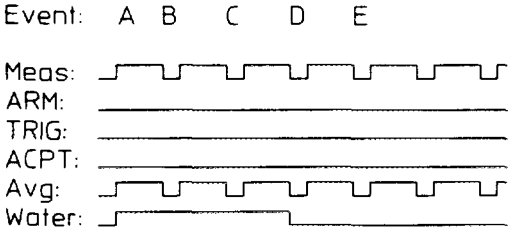
Figure 23-1 illustrates of transition of bits (Measuring, Averaging, Waiting for TRIG, and Waiting for ARM) in the Operational Status condition register.

Note

Trigger conditions must be met for each measurement—even when averaging is turned on.

Instrument Setting: PRESET state

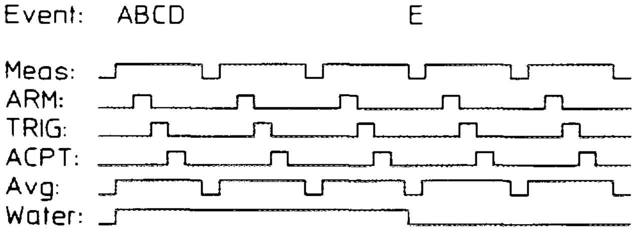
ARM:SOURce IMMediate
AVERage ON
AVERage:IRESult: RATE 4
CALCulate:WATerfall: COUNt 3
TRIGger:SOURce IMMediate



A - INIT command sent
B - 1st update
C - 2nd update
D - 3rd update, waterfall now full
E - 4th update

Instrument Setting: Manual arm, triggered, preview on

ARM:SOUR MAN
AVER:PREV MAN
CALC:WAT:COUN 3
TRIG:SOUR INT1

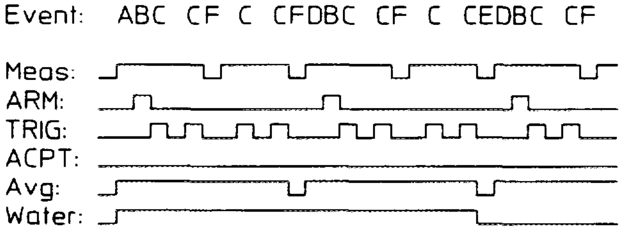


A - INIT command sent
B - wait for manual arm
C - wait for trigger
D - wait for accept/reject
E - 3rd update, waterfall filled

Instrument Setting: Fast average, repeat on, manual arm, input trigger

ARM:SOUR MAN
AVER ON
AVER:COUN 4
AVER:TCON REP
AVER:IRES OFF

AVER:IRES:RATE 2
CALC:WAT:COUN 4
TRIG:SOUR INT1



A - INIT command sent
B - wait for manual arm
C - wait for trigger, take average
D - repeat another set of averages
E - 4th update, waterfall filled
F - screen updated

Figure 23-1. Transition of Operational Status Condition Register When AVERage[:STATe] ON

[SENSe:]AVERage:TCONtrol command/query

Specifies how the analyzer behaves after the count (AVER:COUN) is reached.

Command Syntax:

```
[SENSe:]AVERage:TCONtrol (FREeze|REPeat|EXPonential)
```

Example Statements:

```
OUTPUT 711;"SENSE:AVER:TCONTROL FREeze"
OUTPUT 711;"aver:tcon REPeat"
```

Query Syntax:

```
[SENSe:]AVERage:TCONtrol?
```

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command specifies termination control during averaging.

To specify linear (normal) averaging, send TCON FRE with the AVER:TYPE command.

To specify exponential averaging, send TCON EXP with the AVER:TYPE command.

To specify repeat averaging ("autostart"), send TCON REP with the AVER:TYPE command. The analyzer takes N averages, clears the data, waits for arming conditions, and then takes another N averages. The analyzer continues taking measurements until you send one of the following commands:

- INIT:CONT OFF
- AVER:STAT OFF
- AVER:TCON FRE

Depending upon the instrument mode (specified with the INST:SEL command), some types of termination control and averaging are not valid. See table 23-1.

Table 23-1. Valid Types of Termination Control and Averaging for Instrument Mode

FFT instrument mode

AVERage:TYPE	AVERage:TCONTROL		
	FREeze	REPeat	EXPonential
MAX	valid	valid	not allowed
RMS	valid	valid	valid
VECTor	valid	valid	valid
ECONfidence	not allowed	not allowed	not allowed

Correlation instrument mode

AVERage:TYPE	AVERage:TCONTROL		
	FREeze	REPeat	EXPonential
MAX	not allowed	not allowed	not allowed
RMS	valid	valid	valid
VECTor	valid	valid	valid
ECONfidence	not allowed	not allowed	not allowed

Histogram instrument mode

AVERage:TYPE	AVERage:TCONTROL		
	FREeze	REPeat	EXPonential
MAX	not allowed	not allowed	not allowed
RMS	not allowed	not allowed	not allwoed
VECTor	valid	valid	not allowed
ECONfidence	not allowed	not allowed	not allowed

Octave instrument mode

AVERage:TYPE	AVERage:TCONTROL		
	FREeze	REPeat	EXPonential
MAX	valid	valid	not allowed
RMS	valid	valid	valid
VECTor	not allowed	not allowed	not allowed
ECONfidence	not allowed	not allowed	valid

Order instrument mode

AVERage:TYPE	AVERage:TCONTROL		
	FREeze	REPeat	EXPonential
MAX	not allowed	not allowed	not allowed
RMS	not allowed	not allowed	not allowed
VECTor (time)	valid	valid	valid
ECONfidence	not allowed	not allowed	not allowed

Swept Sine instrument mode

AVERage:TYPE	AVERage:TCONTROL		
	FREeze	REPeat	EXPonential
MAX	not allowed	not allowed	not allowed
RMS	not allowed	not allowed	not allwoed
VECTor	not allowed	not allowed	not allowed
ECONfidence	not allowed	not allowed	not allowed

[SENSe:]AVERage:TIME command/query

Specifies the time period used in averaging octave measurements and histograms.

Command Syntax:

```
[SENSe:]AVERage:TIME (<number>[<unit>])|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  0.0:9.9e37
       <unit> ::= [S|REC|PNT]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Average:Time 3.97722e+37"
OUTPUT 711;"SENSE:AVER:TIME 9.75315e+37"
```

Query Syntax: `[SENSe:]AVERage:TIME?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1.25E-001 (INST:SEL OCT)
1 REC (INST:SEL HIST)
SCPI Compliance:  instrument-specific
```

Description:

In **octave analysis instrument mode** (INST:SEL OCT):

The specified amount of time (in seconds) is used in linear and exponential averaging. It is also used in the peak hold function.

In linear averaging (AVER:TYPE RMS), the value is used for linear integration time. In exponential averaging (AVER:TYPE RMS;TCON EXP), the value is used as the time constant.

In peak hold (AVER:TYPE MAX), this value is used as the integration time over which to hold maximum values. The termination control (AVER:TCON) must be FREeze.

In **histogram analysis instrument mode** (INST:SEL HIST):

This command specifies the length of time averaging data for the histogram. This histogram length can be specified in time (S for seconds), records (REC), or points (PNT). The analyzer rounds the specified histogram length up to the nearest point.

An optimal histogram may be obtained by setting the number of points (specified by this command) to the number of bins^2 (specified with the HIST:BINS command).

$$\text{Optimal Histogram} \approx (\text{AVER:TIME}) = (\text{HIST:BINS})^{2}$$

Note

This command is only valid in histogram instrument mode (INST:SEL HIST) and octave analysis instrument mode (INST:SEL OCT; Option 1D1).

[SENSe:]AVERage:TYPE command/query

Specifjes the type of averaging the analyzer performs.

Command Syntax: `[SENSe:]AVERage:TYPE (MAXimum|RMS|VECTor|ECONfidence)`

Example Statements:

```
OUTPUT 711;"sense:aver:type VECTor"
OUTPUT 711;"Aver:Type ECONfidence"
```

Query Syntax: `[SENSe:]AVERage:TYPE?`

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  RMS
SCPI Compliance:  instrument-specific
```

Description:

The types of averaging available vary according to the instrument mode.

FFT analysis instrument mode (INST:SEL FFT):

To select rms (power) averaging, send AVER:TYPE RMS. The analyzer averages N time records where N is the number of averages you specify with the AVER:COUN command.

To select exponential rms (power) averaging, send AVER:TYPE RMS;TCON EXP. The number of averages specified with AVER:COUN determines the weighting of old versus new data.

To select vector averaging, send AVER:TYPE VECT. The analyzer averages complex values point-by-point in the frequency domain. The input signal must be periodic and a trigger signal from the analyzer's source or from an external signal must be provided.

To select exponential vector averaging, send AVER:TYPE VECT;TCON EXP. The number of averages specified with AVER:COUN determines the weighting of old versus new data.

Send AVER:TYPE MAX to select the peak hold function. The analyzer takes data continuously and mathematically compares each data point along the measured frequency span with the previous peak values. Only the largest value of each point is saved. The results are not mathematically averaged.

In correlation analysis instrument mode (INST:SEL CORR), the following commands are valid:

AVER:TYPE RMS

AVER:TYPE RMS;TCON EXP

AVER:TYPE VECT

AVER:TYPE VECT;TCON EXP

In histogram instrument mode (INST:SEL HIST):

This command is not valid.

Octave analysis instrument mode (INST:SEL OCT):

To select linear averaging, send AVER:TYPE RMS. Old and new data records are weighted equally to yield the arithmetic mean. Averaging is done for a specified amount of time rather than for a number of averages. The value specified with the AVER:TIME command is used for linear integration time.

To select exponential averaging, send AVER:TYPE RMS;TCON EXP. New data is weighted more than old data. The AVER:TIME command specifies the time constant.

To select equal confidence averaging, send AVER:TYPE ECON;TCON EXP. The averaging time for each band is proportional to the bandwidth product for that band. The relative confidence in the measurement is equal across bands. The AVER:CONF command determines the specifies the confidence level.

To specify the peak hold function, send AVER:TYPE MAX. The value specified with the AVER:TIME command is the integration time over which to hold maximum values when AVER:TCON is FRE (repeat is off). When repeat is on (AVER:TCON REP), the integration time is not used.

Order analysis instrument mode (INST:SEL ORD):

To select time averaging, send AVER:TYPE VECT. The analyzer averages N time records where N is the number of averages you specify with the AVER:COUN command.

To select time exponential averaging, send AVER:TYPE VECT;TCON EXP. The number of averages specified with AVER:COUN determines the weighting of old versus new data. As the number of averages increases, new data is weighted less.

In swept sine instrument mode (INST:SEL SINE):

This command is not valid. Averaging integrates a single data point at a time, as opposed to averaging complete time records.

[SENSe:]FEED command/query

Specifies the data source for a measurement; either from the input channels or from the time capture buffer.

Command Syntax: `[SENSe:]FEED {INPut|TCAPture}`

Example Statements:

```
OUTPUT 711;":FEED INPut"
OUTPUT 711;"sens:feed TCAPture"
```

Query Syntax: `[SENSe:]FEED?`

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  INP
SCPI Compliance:  instrument-specific
```

Description:

Data from the time capture buffer may be used for a measurement, replacing the use of the inputs, channel 1 and channel 2. This command directs the analyzer to use the data in the time capture buffer for the measurement.

Note The measurement does not begin until the INITiate command is sent.

[SENSe:]FREQuency:CENTer command/query

Specifies the center frequency for the current measurement.

Command Syntax:

```
[SENSe:]FREQuency:CENTer (<number>[<unit>])|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  0.0234375:115000.0
       <unit> ::= [HZ]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Sens:Freq:Center 6838.14"
OUTPUT 711;"FREQ:CENTER 15540.5"
```

Query Syntax: `[SENSe:]FREQuency:CENTer?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +5.12E+004
SCPI Compliance:  confirmed
```

Description:

FREQ:CENT and FREQ:SPAN work together to define the band of frequency you want to analyze. When you change the value of FREQ:CENT, the value of FREQ:SPAN remains constant.

Step size (FREQ:STEP) determines the change in frequency which results when you send UP or DOWN with this command.

In **swept sine instrument mode** (INST:SEL SINE; Option 1D2):

The allowable values are 15.625 mHz to 511199.984375 Hz. A value specified by this command is rounded to the next lower 15.625 mHz step.

[SENSe:]FREQuency:MANual command/query

Selects a discrete point to be measured during manual sweep mode.

Command Syntax:

```
[SENSe:]FREQuency:MANual {<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0.015625:51200.0
  <unit> ::= [HZ]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":freq:man 27843.8"
OUTPUT 711;"Sense:Freq:Manual 47639.1"
```

Query Syntax:

```
[SENSe:]FREQuency:MANual?
```

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +5.12E+001
SCPI Compliance:  confirmed
```

Description:

The frequency must fall within the start and stop frequencies (FREQ:STAR and FREQ:STOP). Multiple points can be measured by repeating this command with each value.

Note

This command is only used if the SWE:MODE MAN command has been sent.

[SENSe:]FREQuency:RESolution command/query

Sets the resolution of the frequency steps used in swept sine instrument mode.

Command Syntax:

```
[SENSe:]FREQuency:RESolution
(<number><unit>)|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0.015625:51200.0
  <unit> ::= HZ|PCT|PNT/SWP|PNT/DEC|PNT/OCT
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"SENS:FREQ:RESOLUTION 500 HZ"
OUTPUT 711;"freq:resolution 100 PNT/SWP"
```

Query Syntax: `[SENSe:]FREQuency:RESolution?`

Return Format: `NR3`

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +1.01E+002 PNT/SWP
SCPI Compliance:  confirmed
```

Description:

Resolution is limited as follows:

FREQ:SPAN ≥ Resolution ≥ (FREQ:SPAN / 400)

If linear spacing is specified (SWE:SPAC LIN), the resolution can be set using the following units:

- HZ hertz
- PNT/SWP points per sweep
- PCT percent

If logarithmic spacing is specified (SWE:SPAC LOG), the resolution can be set using the following units:

- PNT/SWP points per sweep
- PNT/DEC points per decade
- PNT/OCT points per octave
- PCT percent

To determine what the current unit setting is, send FREQ:RES? UNIT.

Note This command is not used if the FREQ:RES:AUTO ON command has been sent.

[SENSe:]FREQuency:RESolution:AUTO command/query

Selects auto resolution for swept sine instrument mode.

Command Syntax:

```
[SENSe:]FREQuency:RESolution:AUTO {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":Freq:Res:Auto OFF"
OUTPUT 711;"SENS:FREQUENCY:RES:AUTO ON"
```

Query Syntax:

```
[SENSe:]FREQuency:RESolution:AUTO?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

The frequency spacing between measurement points is adjusted automatically by the analyzer. The size of the step is incremented or decremented to accommodate varying frequency response changes.

The analyzer calculates the ratio of the frequency response of the current point to the frequency response of the previous point. If the ratio exceeds the maximum percentage change specified by the FREQ:RES:AUTO:MCH command, the analyzer adjusts the resolution to measure the next point.

The adjusted resolution value is never less than the minimum resolution value specified by the FREQ:RES:AUTO:MIN command. The initial resolution of the sweep between the first two points is specified by FREQ:RES:AUTO:MIN.

Note FREQ:RES:AUTO is ON after a reset (*RST).

[SENSe:]FREQuency:RESolution:AUTO:MCHange command/query

Specifies the maximum change permitted between the frequency response of the current measurement point and the frequency response of the previous measurement point.

Command Syntax:

```
[SENSe:]FREQuency:RESolution:AUTO:MCHange
{<number><unit>}|<step>|<bound>

  <number> ::= a real number (NRf data)
                 limits:  0.391:100
    <unit> ::= PCT
    <step> ::= UP|DOWN
   <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"sense:freq:resolution:auto:mch 30 PCT"
OUTPUT 711;"Frequency:Res:Auto:Mch 25 PCT"
```

Query Syntax: `[SENSe:]FREQuency:RESolution:AUTO:MCHange?`

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +2 PCT
SCPI Compliance:  instrument-specific
```

Description:

This command is used with auto resolution (FREQ:RES:AUTO ON).

The analyzer calculates the ratio of the frequency response of the current point to the frequency response of the previous point. The ratio exceeds the value specified by this command, the analyzer estimates a correction for the resolution and applies it to the next measurement point.

[SENSe:]FREQuency:RESolution:AUTO:MINimum command/query

Specifies the initial resolution of a swept sine measurement with automatic resolution.

Command Syntax:

```
[SENSe:]FREQuency:RESolution:AUTO:MINimum
{<number><unit>}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0.015625:51200.0
  <unit> ::= HZ|PCT|PNT/SWP|PNT/DEC|PNT/OCT
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":FREQ:RESOLUTION:AUTO:MINIMUM 10 HZ"
OUTPUT 711;"sens:freq:resolution:auto:minimum 200 PNT/SWP"
```

Query Syntax:

```
[SENSe:]FREQuency:RESolution:AUTO:MINimum?
```

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +4.01E+002 PNT/SWP
SCPI Compliance:  instrument-specific
```

Description:

If automatic resolution is specified (FREQ:RES:AUTO ON), this command sets the initial resolution used between the first and second measurement point. It also specifies the minimum resolution the analyzer uses if an adjustment in resolution is required.

The analyzer calculates the ratio of the frequency response of the current measurement point to the frequency response of the previous measurement point. If the ratio exceeds the specified limit (FREQ:RES:AUTO:MCH), the analyzer corrects the resolution and applies it to the next measurement point. The correction is never less than the value specified by this command.

If linear spacing is specified (SWE:SPAC LIN), the resolution can be set using the following units:

- HZ hertz
- PNT/SWP points per sweep
- PCT percent

If logarithmic spacing is specified (SWE:SPAC LOG), the resolution can be set using the following units:

- PNT/SWP points per sweep
- PNT/DEC points per decade
- PNT/OCT points per octave
- PCT percent

To determine what the current unit setting is, send FREQ:RES? UNIT.

[SENSe:]FREQuency:RESolution:OCTave command/query

Specifies the type of octave measurement.

Command Syntax: `[SENSe:]FREQuency:RESolution:OCTave (THIRd|FULL|TWELth)`

Example Statements:

```
OUTPUT 711;"Sens:Freq:Resolution:Oct THIRd"
OUTPUT 711;"FREQUENCY:RES:OCT TWELth"
```

Query Syntax: `[SENSe:]FREQuency:RESolution:OCTave?`

Return Format: CHAR

Attribute Summary:

```
Option:  1D1 Realtime Octave
Overlapped:  no
Preset State:  THIR
SCPI Compliance:  instrument-specific
```

Description:

To select 1/3 octave band measurements, send FREQ:RES:OCT:THIR. The center frequency of each 1/3 octave frequency band is located at a frequency of $2^{(1/3)}$ times the preceding 1/3 octave band. The analyzer displays 33 frequency bands and a total RMS power band.

To specify start and stop frequencies, send the FREQ:STAR and FREQ:STOP commands. The minimum start frequency is 80 mHz. The maximum stop frequency is 31.5 kHz. If you change one of these frequencies, the analyzer adjusts the other frequency as follows:

$$\text{stop frequency} = (2^{11} / 2^{(1/3)}) \times \text{start frequency}$$
or
$$\text{stop frequency} = 1625.53 \times \text{start frequency}$$

To select 1/12 octave band measurements, send FREQ:RES:OCT:TWEL. The center frequency of each 1/12 octave frequency band is located at a frequency of $2^{(1/12)}$ times the preceding 1/12 octave band. The analyzer displays 132 frequency bands and a total RMS power band.

To specify start and stop frequencies, send the FREQ:STAR and FREQ:STOP commands. The minimum start frequency is 91.45 mHz. The maximum stop frequency is 11.31 kHz. If you change one of these frequencies, the analyzer adjusts the other frequency as follows:

$$\text{stop frequency} = (2^{11} / 2^{(1/12)}) \times \text{start frequency}$$
or
$$\text{stop frequency} = 1933.05 \times \text{start frequency}$$

To select full octave band measurements, send FREQ:RES:OCT:FULL. The analyzer displays 11 frequency bands and a total RMS power band.

To specify start and stop frequencies, send the FREQ:STAR and FREQ:STOP commands. The minimum start frequency is 61.04 mHz. The maximum stop frequency is 16 kHz. If you change one of these frequencies, the analyzer adjusts the other frequency as follows:

$$\text{stop frequency} = (2^{11} / 2) \times \text{start frequency}$$

or

$$\text{stop frequency} = 1024 \times \text{start frequency}$$

[SENSe:]FREQuency:SPAN command/query

Specifies the frequency bandwidth to be measured.

Command Syntax: `[SENSe:]FREQuency:SPAN {<number>[<unit>]}|<step>|<bound>`

```
<number> ::= a real number (NRf data)
             limits:  0.015625:102400.0
  <unit> ::= [HZ]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":frequency:span 76604.9"
OUTPUT 711;"Sense:Freq:Span 41986.6"
```

Query Syntax: `[SENSe:]FREQuency:SPAN?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1.024E+005
SCPI Compliance:  confirmed
```

Description:

The value of FREQ:SPAN is used with either FREQ:CENT or FREQ:STAR to define the band of frequencies. The maximum frequency span is 102.4 kHz for a single channel measurement; 51.2 for two channel measurements. The minimum frequency span is 97.65625 mHz for a single channel measurement; 195.3125 mHz for two channel measurements. Allowable values for the frequency span are determined by the following formula:

$$(\text{maximum frequency span}) / 2^{19}$$

When you send this command, the value of the record length (SWE:TIME) is adjusted so the following formula is true:

$$\text{record length} = 400/\text{FREQ:SPAN}$$

The frequency span is narrowed if the value of the start frequency (FREQ:STAR) and half the frequency span exceeds 115 kHz for one channel measurements and 57.5 kHz for two channel measurements. In other words, the frequency span is limited by the following formulas:

for one channel measurements:

```
(FREQ:STAR) + (FREQ:SPAN/2) ≤ 115 kHz
```

for two channel measurements:

```
(FREQ:STAR) + (FREQ:SPAN/2) ≤ 57.5 kHz
```

The frequency resolution is determined by the frequency span.

frequency resolution = FREQ:SPAN/400

FREQ:SPAN UP increases the frequency span to the next largest allowable value. FREQ:SPAN DOWN decreases the frequency span to the next smallest allowable value.

In **swept sine instrument mode** (INST:SEL SINE; Option 1D2):

The value specified by this command is rounded to the next higher 15.625 mHz step. The frequency span is narrowed if the value of the start frequency and the value of the frequency span exceeds 51.2 kHz. That is:

$$(\text{FREQ:STAR}) + (\text{FREQ:SPAN}) \leq 51.2\ \text{kHz}$$

If logarithmic spacing is used (SWE:SPAC LOG), the value may be expressed in terms of hertz, decades or octaves.

To increment the size of the span in steps relative to 51.2 kHz, send FREQ:STEP UP. To decrement the size of the span in steps, send FREQ:STEP DOWN.

[SENSe:]FREQuency:SPAN:FULL command

Sets the analyzer to the widest frequency span available for the current instrument mode.

Command Syntax:

```
[SENSe:]FREQuency:SPAN:FULL
```

Example Statements:

```
OUTPUT 711;"SENSE:FREQ:SPAN:FULL"
OUTPUT 711;"freq:span:full"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The start frequency is set to 0 Hz; the stop frequency is set to 102.4 kHz for one- channel measurements and 51.2 kHz for two-channel measurements.

In swept sine instrument mode (INST:SEL SINE; Option 1D2):

> The start frequency is “fixed” at 0 Hz.

[SENSe:]FREQuency:SPAN:LINK command/query

Specifies the frequency parameter which remains constant if frequency span or record length is modified.

Command Syntax:

```
[SENSe:]FREQuency:SPAN:LINK {STARt|CENTer}
```

Example Statements:

```
OUTPUT 711;":Frequency:Span:Link STARt"
OUTPUT 711;"SENSE:FREQ:SPAN:LINK STARt"
```

Query Syntax:

```
[SENSe:]FREQuency:SPAN:LINK?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  STAR
SCPI Compliance:  confirmed
```

Description:

This command “anchors” or “fixes” the start frequency or the center frequency.

If FREQ:SPAN LINK STAR is sent, the start frequency does not change when the frequency span or record length changes.

Note FREQ:SPAN:LINK is set to CENTer after a reset (*RST).

[SENSe:]FREQuency:STARt command/query

Specifies the start (lowest) frequency for the frequency band of the current measurement.

Command Syntax: `[SENSe:]FREQuency:STARt (<number>[<unit>])|<step>|<bound>`

```
<number> ::= a real number (NRf data)
             limits:  0.0:114999.9023
  <unit> ::= [HZ]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"sens:frequency:star 47931.8"
OUTPUT 711;"Frequency:Star 2719.49"
```

Query Syntax: `[SENSe:]FREQuency:STARt?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  confirmed
```

Description:

The values of FREQ:STAR and FREQ:SPAN define the frequency bandwidth. The size of the bandwidth (FREQ:SPAN) remains constant if the start frequency changes.

Allowable start frequency values are defined by the following equation:

single channel measurements:

`0 ≤ start frequency ≤ (115 kHz - (frequency span/2))`

two channel measurements:

`0 ≤ start frequency ≤ (57.5 kHz - (frequency span/2))`

Step size (FREQ:STEP) determines the change in frequency which results when you send UP or DOWN with this command.

In **octave analysis instrument mode** (INST:SEL OCT; Option 1D1):

This command specifies the start frequency for the octave measurement. The frequency can be specified in Hz or as a band number.

In **swept sine instrument mode** (INST:SEL SINE; Option 1D2):

The allowable values are: 15.625 mHz to 511199.984375 Hz. A value specified by this command is rounded to the next lower 15.625 mHz step.

[SENSe:]FREQuency:STEP[:INCRement] command/query

Specifies the step size which is used when changing frequency parameters.

Command Syntax:

```
[SENSe:]FREQuency:STEP[:INCRement]
{<number>[<unit>]}|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  0.015625:10240.0
       <unit> ::= [HZ]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":FREQ:STEP 5384.31"
OUTPUT 711;"sens:frequency:step:incr 7351.21"
```

Query Syntax: `[SENSe:]FREQuency:STEP[:INCRement]?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +2.0E+003
SCPI Compliance:  confirmed
```

Description:

Step size determines the change in frequency which results when you send UP or DOWN with the FREQ:CENT or the FREQ:STAR commands.

In swept sine instrument mode (INST:SEL SINE; Option 1D2):

Step size determines the change in frequency which results when you send UP or DOWN with any of the following commands:

- FREQ:CENTer
- FREQ:STARt
- FREQ:STOP
- FREQ:MANual

[SENSe:]FREQuency:STOP command/query

Sets the stop frequency to the specified value.

Command Syntax:

```
[SENSe:]FREQuency:STOP (<number>[<unit>])|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0.03125:115000.0
  <unit> ::= [HZ]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Sense:Freq:Stop 27142.6"
OUTPUT 711;"FREQ:STOP 79909.5"
```

Query Syntax: `[SENSe:]FREQuency:STOP?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1.024E+005
SCPI Compliance:  confirmed
```

Description:

This command defines the upper limit of the frequency bandwidth. The start frequency remains fixed (FREQ:SPAN:LINK). The values for the center frequency, the frequency span, and the record length change to appropriate values.

In **octave measurement mode** (INST:SEL OCT; Option 1D1):

> This command specifies the stop frequency for the octave measurement. The frequency can be specified in Hz or as a band number.

In **swept sine instrument mode** (INST:SEL SINE; Option 1D2):

> The value specified by this command is rounded to the next higher 15.625 mHz step. The start frequency (FREQ:STAR) is held constant and selected as the new "anchor" for the measurement. The center frequency and frequency span are adjusted to appropriate values.

[SENSe:]HISTogram:BINS command/query

Specifies the number of bins in a histogram.

Command Syntax:

```
[SENSe:]HISTogram:BINS <number>|<step>|<bound>

        <number> ::= a real number (NRf data)
                     limits:  4:1024
          <step> ::= UP|DOWN
         <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":histogram:bins 799.55"
OUTPUT 711;"Sense:Hist:Bins 782.005"
```

Query Syntax: `[SENSe:]HISTogram:BINS?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +5.12E+002
SCPI Compliance:  instrument-specific
```

Description:

To obtain an optimal histogram, set the number of bins equal to the square root of the number of points specified with the [SENSe:]AVERage:TIME command.

$$\text{Optimal Histogram} \approx \text{(HIST:BINS)} = \sqrt{\text{([SENSe:]AVERage:TIME)}}$$

[SENSe:]ORDer:MAXimum command/query

Specifies the number of orders to be displayed.

Command Syntax:

```
[SENSe:]ORDer:MAXimum (<number>[<unit>])|<step>|<bound>

      <number> ::= a real number (NRf data)
                   limits:  3.125:200
        <unit> ::= [ORD]
        <step> ::= UP|DOWN
       <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"SENSE:ORD:MAXIMUM 48"
OUTPUT 711;"ord:max 148"
```

Query Syntax: `[SENSe:]ORDer:MAXimum?`

Return Format: `NR1`

Attribute Summary:

```
Option:  1D0 Computed Order Tracking
Overlapped:  no
Preset State:  +10
SCPI Compliance:  instrument-specific
```

Description:

The command defines the highest order to be tracked and is used with the ORD:RES command to specify the spacing between order lines. The allowable range is 3.125 to 200 orders.

ORD:MAX / ORD RES ≤ 200

[SENSe:]ORDer:RESolution command/query

Specifies order resolution.

Command Syntax:

```
[SENSe:]ORDer:RESolution (<number>[<unit>])|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  .0078125:1
       <unit> ::= [ORD]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Order:Res 0.928812"
OUTPUT 711;"SENSE:ORD:RES 0.767834"
```

Query Syntax:

```
[SENSe:]ORDer:RESolution?
```

Return Format: NR3

Attribute Summary:

```
Option:  1D0 Computed Order Tracking
Overlapped:  no
Preset State:  +1.0E-001
SCPI Compliance:  instrument-specific
```

Description:

This command defines the spacing of the order map lines as a ratio of the number of orders displayed (ORD:MAX) divided by the number of lines per order.

ORD:MAX / ORD RES ≤ 400

Values can range from .0078125 to 1.

[SENSe:]ORDer:RPM:MAXimum command/query

Specifies the maximum rotational speed range you want to analyze.

Command Syntax: `[SENSe:]ORDer:RPM:MAXimum <number>|<step>|<bound>`

```
<number> ::= a real number (NRf data)
             limits:  0:9.9e37
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"sense:ord:rpm:max 2.86684e+37"
OUTPUT 711;"Ord:Rpm:Max 9.73824e+37"
```

Query Syntax: `[SENSe:]ORDer:RPM:MAXimum?`

Return Format: NR3

Attribute Summary:

```
Option:  1D0 Computed Order Tracking
Overlapped:  no
Preset State:  +6.0E+003
SCPI Compliance:  instrument-specific
```

Description:

For runup measurements, the measurement stops at the speed specified by this command.

For rundown measurements, the measurement starts at the speed specified by this command and continues to the minimum RPM (ORD:RPM:MIN).

[SENSe:]ORDer:RPM:MINimum command/query

Specifies the minimum rotational speed range you want to analyze.

Command Syntax:

```
[SENSe:]ORDer:RPM:MINimum <number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits: 0:9.9e37
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":ORDER:RPM:MIN 4.23826e+37"
OUTPUT 711;"sense:ord:rpm:min 3.7435e+36"
```

Query Syntax:

```
[SENSe:]ORDer:RPM:MINimum?
```

Return Format: NR3

Attribute Summary:

```
Option:  1D0 Computed Order Tracking
Overlapped:  no
Preset State:  +6.0E+002
SCPI Compliance:  instrument-specific
```

Description:

For runup measurements, the measurement starts at the speed specified by this command and continues to the maximum RPM (ORD:RPM:MAX).

For rundown measurements, the measurement stops at the speed specified by this command.

[SENSe:]ORDer:TRACk[1|2|3|4|5] command/query

Specifies the order number for the selected track.

Command Syntax:

```
[SENSe:]ORDer:TRACk[1|2|3|4|5]
{<number>[<unit>]}|<step>|<bound>
```

```
<number> ::= a real number (NRf data)
                limits:  0:9.9e37
  <unit> ::= [ORD]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Sens:Order:Trac5 2147483647"
OUTPUT 711;"ORDER:TRAC 2147483647"
```

Query Syntax:

```
[SENSe:]ORDer:TRACk[1|2|3|4|5]?
```

Return Format: NR1

Attribute Summary:

```
Option:  1D0 Computed Order Tracking
Overlapped:  no
Preset State:  +1  TRACk 1
               +2  TRACk 2
               +3  TRACk 3
               +4  TRACk 4
               +5  TRACk 5
SCPI Compliance:  instrument-specific
```

Description:

This command assigns order values to each of the five possible orders to be tracked. The value must be between 0 and the programmed highest order (ORD:MAX) and can be specified in .0001 increments.

If an order value is *not* assigned, the default order number is used.

[SENSe:]ORDer:TRACk[1|2|3|4|5]:STATe command/query

Selects order track or order spectrum measurements.

Command Syntax:

```
[SENSe:]ORDer:TRACk[1|2|3|4|5]:STATe (OFF|0|ON|1)
```

Example Statements:

```
OUTPUT 711;":ord:track3:stat ON"
OUTPUT 711;"Sense:Ord:Trac5:State OFF"
```

Query Syntax:

```
[SENSe:]ORDer:TRACk[1|2|3|4|5]:STATe?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D0 Computed Order Tracking
Overlapped:  no
Preset State:  +0 (all tracks)
SCPI Compliance:  instrument-specific
```

Description:

The analyzer makes an order track measurement when ORD:TRAC:STAT ON is sent.

The analyzer makes an order spectrum measurement when ORD:TRAC:STAT OFF is sent.

Note

This command *ignores* the order track specifier. If the state is changed for one order track, the state changes for *all* order tracks.

[SENSe:]REJect:STATe command/query

Turns overload rejection on or off.

Command Syntax: `[SENSe:]REJect:STATe {OFF|0|ON|1}`

Example Statements:

```
OUTPUT 711;"SENS:REJECT:STAT OFF"
OUTPUT 711;"rej:state OFF"
```

Query Syntax: `[SENSe:]REJect:STATe?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

If overload rejection is off (the default condition), all time records are included in the measurement.

If AVER:REJ ON is sent, the time record from the overloaded input channel is not included in the measurement results. The concurrent time record from the other channel is rejected as well. The measurement continues until the analyzer has collected the specified number of non-distorted time records.

[SENSe:]SWEep:DIRection command/query

Specifies the direction of the sweep.

Command Syntax:

```
[SENSe:]SWEep:DIRection (UP|DOWN)
```

Example Statements:

```
OUTPUT 711;":Swe:Direction DOWN"
OUTPUT 711;"SENS:SWE:DIRECTION UP"
```

Query Syntax:

```
[SENSe:]SWEep:DIRection?
```

Return Format: CHAR

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  UP
SCPI Compliance:  confirmed
```

Description:

To initiate a sweep that begins at the lowest frequency (FREQ:STARt) and ends at the highest frequency (FREQ:STOP), send SWE:DIR UP.

To initiate a sweep that begins at the highest frequency (FREQ:STOP) and ends at the lowest frequency (FREQ:STARt), send SWE:DIR DOWN.

This command is not used if a manual sweep is specified with the SWEep:MODE MAN command.

[SENSe:]SWEep:DWELl command/query

Specifies the integration time for swept sine measurements.

Command Syntax: `[SENSe:]SWEep:DWELl <dwell_time>`

```
<dwell_time> ::= (<number>[<unit>])|<step>|<bound>
    <number> ::= a real number (NRf data)
                 limits:  250.e-6:32768
      <unit> ::= [S|CYCLE]
      <step> ::= UP|DOWN
     <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"sens:sweep:dwel .005 s"
OUTPUT 711;"Swe:Dwell 1 cycle"
```

Query Syntax: `[SENSe:]SWEep:DWELl?`

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +5.0 CYCLE
SCPI Compliance:  instrument-specific
```

Description:

Integration time is the amount of time that each point is measured.

Sending SWE:DWEL in seconds, results in a constant integration scale.

Sending SWE:DWEL in cycles, results in a proportional integration scale. At higher frequencies the same number of cycles occurs in a shorter time. The integrate time must be a minimum of 1 cycle long. The analyzer takes any value less than one as one complete cycle.

[SENSe:]SWEep:MODE command/query

Specifies automatic or manual sweep modes.

Command Syntax:

```
[SENSe:]SWEep:MODE (AUTO|MANual)
```

Example Statements:

```
OUTPUT 711;":SWE:MODE AUTO"
OUTPUT 711;"sens:swe:mode AUTO"
```

Query Syntax:

```
[SENSe:]SWEep:MODE?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  AUTO
SCPI Compliance:  confirmed
```

Description:

To select automatic sweep mode, send SWE:MODE AUTO. The instrument controls the sweep according to the following parameters:

- FREQ:STARt
- FREQ:STOP
- FREQ:RESolution
- SWE:DIRection
- SWE:SPACe

The values of the parameters are specified by default or by the appropriate command.

Send SWE:MODE MAN to select a discrete sweep; the measurement occurs only at the frequency points specified by the FREQ:MANual command.

[SENSe:]SWEep:OVERlap command/query

Specifies the maximum amount of time record overlap.

Command Syntax:

```
[SENSe:]SWEep:OVERlap (<number>[<unit>])|<step>|<bound>

      <number> ::= a real number (NRf data)
                   limits:  0:99
        <unit> ::= [PCT]
        <step> ::= UP|DOWN
       <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Sens:Sweep:Over 84 PCT"
OUTPUT 711;"SWE:OVERLAP .44"
```

Query Syntax: `[SENSe:]SWEep:OVERlap?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

Overlap processing is not used with triggered measurements. The analyzer must be in:

- real time.
- automatic arming (ARM:SOUR IMM).
- frequency span ≤ 12.8 kHz (1 channel)
  frequency span ≤ 6.4 kHz (2 channel)
  (set with [SENSe:]FREQ:SPAN command).
- continuous (free run) trigger mode (TRIG:SOUR IMM).
- overload rejection off ([SENSe:]REJect:STATe OFF).

Data points from the end of one time record can be reused at the beginning of the next time record. This results in the overlapping of time records. Use this command to specify the amount of the block size which should be common to two consecutive time records.

As the frequency span decreases, the corresponding time record length increases. Overlap processing becomes possible when the instrument takes more time to collect time records than it does to process them. This allows you to make a faster measurement especially with narrow frequency spans. Overlap processing also reduces statistical variance caused by windowing.

You can specify overlap either as a percentage or as a fraction of the time record length. SWE:OVER 0.22 is the same as SWE:OVER 22 PCT. The value you send is rounded to the nearest allowable percentage (an integer between 0 an 99).

The query returns a value that indicates the amount of overlap currently specified. The value is returned as a percent.

[SENSe:]SWEep:SPACing command/query

Selects linear or logarithmic spacing between measurement data points.

Command Syntax: `[SENSe:]SWEep:SPACing {LINear|LOGarithmic}`

Example Statements:

```
OUTPUT 711;":swe:spacing LOGarithmic"
OUTPUT 711;"Sens:Swe:Spacing LINear"
```

Query Syntax: `[SENSe:]SWEep:SPACing?`

Return Format: CHAR

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  LIN
SCPI Compliance:  confirmed
```

Description:

Send SWE:SPAC LIN for linearly spaced frequency points. The frequency step size does not change during the sweep of the frequency points; it remains constant over the entire spectrum.

Send SWE:SPAC LOG for logarithmically or proportionately spaced frequency points. The ratio of the location of adjacent points is constant. In addition to hertz, decade or octave units may be used with the FREQ:SPAN command. The following units may be used with the FREQ:RES command:

- PNT/SWP
- PNT/DEC
- PNT/OCT
- PCT

[SENSe:]SWEep:STIMe command/query

Specifies the settling time for a swept sine measurement.

Command Syntax:

```
[SENSe:]SWEep:STIMe {<number>[<unit>]}|<step>|<bound>

   <number> ::= a real number (NRf data)
                limits:  0:9.9e37
     <unit> ::= [S|CYCLE]
     <step> ::= UP|DOWN
    <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"SENS:SWEEP:STIM 0.005 s"
OUTPUT 711;"swe:stime 1 cycle"
```

Query Syntax:

```
[SENSe:]SWEep:STIMe?
```

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +5.0 CYCLE
SCPI Compliance:  instrument-specific
```

Description:

Settling time is the delay between changing the source frequency and starting the measurement at each point. This allows the device under test to stabilize after the frequency changes.

Units can be specified in seconds (S) or cycles (CYCLE).

[SENSe:]SWEep:TIME command/query

Specifies the length of the time record in seconds.

Command Syntax: `[SENSe:]SWEep:TIME (<number>[<unit>])|<step>|<bound>`

```
<number> ::= a real number (NRf data)
             limits:  3.90625e-3:4096.0
  <unit> ::= [S]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Swe:Time 1019.39"
OUTPUT 711;"SENS:SWE:TIME 1461.43"
```

Query Syntax: `[SENSe:]SWEep:TIME?`

Return Format: `NR3`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +3.90625E- 003
SCPI Compliance:  confirmed
```

Description:

When you send this command, two other values may be adjusted. The value of the frequency span (FREQ:SPAN) is adjusted to 400 / record length hertz. If the start frequency is "fixed" (FREQ:SPAN:LINK STAR), the center and stop frequencies are adjusted accordingly. If the center frequency is "fixed" (FREQ:SPAN:LINK CENT), the start and stop frequencies are adjusted.

If you change the frequency span, the time record length is adjusted to 400 / frequency span.

In **correlation analysis instrument mode** (INST:SEL CORR):

This command specifies the record length, T. T represents the length of the raw time record collected. T is used in correlation windowing functions as follows:

-T/4 to T/4
0 to T/2
-T/2 to T/2

See the *HP 35665A Concepts Guide* for additional information about record length in correlation analysis.

[SENSe:]TCAPture:ABORt command

Stops the time capture process.

Command Syntax:

```
[SENSe:]TCAPture:ABORt
```

Example Statements:

```
OUTPUT 711;"sens:tcapture:abor"
OUTPUT 711;"Tcap:Abort"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The amount of data in the time capture buffer is less than the amount specified by the TCAP:LENG command. The analyzer aborts the time capture process *immediately*. Any partial time record is discard.

[SENSe:]TCAPture:DELete command

Removes the time capture buffer.

Command Syntax:

```
[SENSe:]TCAPture:DELete
```

Example Statements:

```
OUTPUT 711;":TCAP:DELETE"
OUTPUT 711;"sens:tcap:delete"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The analyzer removes the memory allocation for the time capture buffer, effectively “clearing” the buffer.

[SENSe:]TCAPture[:IMMediate] command

Starts the collection of data for the time capture process.

Command Syntax:

```
[SENSe:]TCAPture[:IMMediate]
```

Example Statements:

```
OUTPUT 711;"Sens:Tcapture:Imm"
OUTPUT 711;"TCAP"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The analyzer collects data from the input channels and stores it in the time capture buffer.

If the command (TCAP:MALL) which allocates memory has not been sent; the analyzer automatically allocates memory based on the current settings for the frequency span, the size of the time capture buffer (TCAP:LENG) and the size of the tachometer buffer if it is enabled with the [SENSe:]TCAP:TACH[:STATe] ON command.

The size of the tachometer buffer is determined by the number of revolutions per minute specified with the [SENSe:]TCAP:TACH:RPM:MAX command and the number of tachometer pulses per revolution specified with the TRIGger:TACH:PCOunt command.

[SENSe:]TCAPture:LENGth command/query

Specifies the length of the time capture buffer.

Command Syntax:

```
[SENSe:]TCAPture:LENGth {<number>[<unit>]}|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits: 0:9.9e37
       <unit> ::= [S|REC|PNT]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":tcapture:leng 4.13692e+37"
OUTPUT 711;"Sense:Tcap:Leng 4.92083e+37"
```

Query Syntax: `[SENSe:]TCAPture:LENGth?`

Return Format: NR3

Attribute Summary:

```
Option: not applicable
Overlapped: no
Preset State: +0.0
SCPI Compliance: instrument-specific
```

Description:

If length is specified in seconds, the size of the buffer is relative to the frequency span.

If length is specified in time records or points, the size of the buffer is absolute. There are 1024 points per time record.

Any specified length is rounded up to the nearest time record.

The maximum capture length is dependent upon the memory configuration:

Maximum Capture Length (in time records)		
	1 Channel	2 Channel
2 Mbytes	200	100
4 Mbytes (Option 1C1)	1000	500
8 Mbytes (Option ANA)	3000	1500

The minimum is 1 time record. If the TCAP:ABOR command is sent, the analyzer aborts the process and discards the partially filled time record.

Note

The analyzer's memory is *not* allocated until TCAP:MALL or TCAP[:IMM] has been sent.

[SENSe:]TCAPture:MALLocate command

Allocates memory for the time capture buffer.

Command Syntax:

```
[SENSe:]TCAPture:MALLocate
```

Example Statements:

```
OUTPUT 711;"SENSE:TCAP:MALLOCATE"
OUTPUT 711;"tcap:mall"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The analyzer automatically allocates memory when the TCAP[:IMM] command is sent. So although this command is not necessary, it ensures you have sufficient memory allocated for the time capture buffer.

[SENSe:]TCAPture:STARt[1|2] command/query

Specifies the beginning of the time capture data used in a measurement.

Command Syntax:

```
[SENSe:]TCAPture:STARt[1|2] (<number>[<unit>])|<step>|<bound>

   <number> ::= a real number (NRf data)
                limits:  -9.9e37:9.9e37
     <unit> ::= [S|REC|PNT]
     <step> ::= UP|DOWN
    <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Tcapture:Star 7.6288e+37"
OUTPUT 711;"SENSE:TCAP:STAR 1.25067e+37"
```

Query Syntax: `[SENSe:]TCAPture:STARt[1|2]?`

Return Format: `NR3`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

This command allows to select a portion of the time capture data to be used for the measurement. You can set the "analysis region" for each channel independently.

Use the TCAP:STOP command to specify the end of the time capture data.

The analyzer sets this value to the beginning of the capture data upon receiving any of the following commands:

- MMEMory:LOAD:STATe
- MMEMory:LOAD:TCAPture
- [SENSe:]TCAPture[:IMMediate]
- SYSTem:PRESet

[SENSe:]TCAPture:STOP[1|2] command/query

Specifies the end of the time capture data used in a measurement.

Command Syntax:

```
[SENSe:]TCAPture:STOP[1|2] (<num-
ber>[<unit>])|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
  <unit> ::= [S|REC|PNT]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"sense:tcap:stop 3.6729e+37"
OUTPUT 711;"Tcap:Stop 9.26164e+37"
```

Query Syntax: `[SENSe:]TCAPture:STOP[1|2]?`

Return Format: `NR3`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  end of data
SCPI Compliance:  instrument-specific
```

Description:

This command allows you to select a portion of the time capture data to be used for the measurement. You can set the "analysis region" for each channel independently.

Use the TCAP:STAR command to specify the beginning of the time capture data.

The analyzer sets this value to the end of the capture data upon receiving any of the following commands:

- MMEMory:LOAD:STATe
- MMEMory:LOAD:TCAPture
- [SENSe:]TCAPture[:IMMediate]
- SYSTem:PRESet

[SENSe:]TCAPture:TACHometer:RPM:MAXimum command/query

Specifies the tachometer's maximum RPM when included in the time capture buffer.

Command Syntax:

```
[SENSe:]TCAPture:TACHometer:RPM:MAXimum
<number>|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  5:491519
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":TCAPTURE:TACH:RPM:MAX 89641.8"
OUTPUT 711;"sens:tcapture:tach:rpm:max 279089"
```

Query Syntax: `[SENSe:]TCAPture:TACHometer:RPM:MAXimum?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +6000
SCPI Compliance:  instrument-specific
```

Description:

This command sets the upper limit of the rotation speed range you want to monitor for measurements using the time capture buffer.

In a runup measurement, this value specifies when the measurement stops. In a rundown measurement, this value specifies when the measurement starts.

Note This command is *not* used when the analyzer is in order analysis (INST:SEL ORD; Option 1D0). The analyzer's tachometer is always ON in this instrument mode. Specify the tachometer's maximum RPM with the [SENSe:]ORDer:RPM:MAXimum command when the analyzer is in order analysis instrument mode.

To include the tachometer input signal in the time capture buffer, send the TCAPture:TACHometer[:STATe] command.

Caution The value specified with this command is used by the analyzer to allocate memory for the tachometer buffer. If the value specified is too low, the analyzer aborts the time capture when it fills the tachometer buffer.

[SENSe:]TCAPture:TACHometer[:STATe] command/query

Directs the analyzer to include the tachometer input signal in the time capture buffer.

Command Syntax:

```
[SENSe:]TCAPture:TACHometer[:STATe] {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"Sens:Tcapture:Tach:State OFF"
OUTPUT 711;"TCAP:TACH OFF"
```

Query Syntax:

```
[SENSe:]TCAPture:TACHometer[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

This command determines if the tachometer input signal is included in the time capture buffer.

The tachometer parameters must be setup before capturing the data. See the TCAP:TACH:RPM:MAX and TRIGger:TACHometer commands for more information.

Note This command is *not* required for order analysis (INST:SEL ORD; Option 1D0). The analyzer's tachometer is always ON in order analysis instrument mode.

[SENSe:]VOLTage[1|2]:RANGe:AUTO command/query

Automatically selects the best range on the specified channel for the current input signal.

Command Syntax:

```
[SENSe:]VOLTage[1|2]:RANGe:AUTO {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":voltage2:rang:auto OFF"
OUTPUT 711;"Sens:Volt:Range:Auto ON"
```

Query Syntax:

```
[SENSe:]VOLTage[1|2]:RANGe:AUTO?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

At the start of a measurement, the analyzer selects the lowest input range. It steps the input up through successive range values until the input is no longer in an overload condition. The analyzer continues to adjust the range upward in response to increased signal amplitude.

The analyzer *never* adjusts the range downward in response to a decrease in signal amplitude. If the range is too large for the current input signal, send VOLT[1|2]:RANG:AUTO ON to restart autorange.

If you use command for a swept sine measurement, the analyzer adjusts the input range upward or downward at each measurement point, depending on the signal level at the measurement point.

To turn off the autorange feature:

- Set the input range by specifying a value with the VOLT:RANG command.

OR

- Send VOLT[1|2]:RANG:AUTO OFF. The range is fixed at the last autorange value.

Note The analyzer does not autorange during a time capture or during an averaged measurement.

If the channel specifier is not used, the command defaults to channel 1.

[SENSe:]VOLTage[1|2]:RANGe:UNIT:USER:LABel command/query

Assigns a name to the engineering units for the specified input channel.

Command Syntax:

```
[SENSe:]VOLTage[1|2]:RANGe:UNIT:USER:LABel <STRING>

<STRING> ::= ASCII characters - 32 through 126
             maximum number of characters:  4
```

Example Statements:

```
OUTPUT 711;"SENSE:VOLT2:RANG:UNIT:USER:LABEL 'SPL'"
OUTPUT 711;"volt:rang:unit:user:label 'g'"
```

Query Syntax:

```
[SENSe:]VOLTage[1|2]:RANGe:UNIT:USER:LABel?
```

Return Format:

```
STRING
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  EU
SCPI Compliance:  instrument-specific
```

Description:

The name assigned with this command labels the display's Y-axis. The label appears only when VOLT:RANG:UNIT:USER is ON.

If the channel specifier is not used, the command defaults to channel 1.

The query returns the last entered engineering unit name for the specified channel.

Note

An engineering unit label is *not* applied until the display is uppdated with new measurement results.

[SENSe:]VOLTage[1|2]:RANGe:UNIT:USER:SFACtor command/query

Specifies a scaling factor for engineering units (EU).

Command Syntax:

```
[SENSe:]VOLTage[1|2]:RANGe:UNIT:USER:SFACtor
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:9.9e37
  <unit> ::= [V/EU|EU/V]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Volt:Rang:Unit:User:Sfactor 0.1"
OUTPUT 711;"SENS:VOLT:RANGE:UNIT:USER:SFAC 1.8e-06"
```

Query Syntax: `[SENSe:]VOLTage[1|2]:RANGe:UNIT:USER:SFACtor?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command sets the number of volts per engineering units (V/EU).

The scaling factor calibrates engineering unit values to volts measured at the specified input channel. The EU scaling factor is only used when the VOLT:RANG:UNIT:USER ON command is sent.

If the channel specifier is not used, the command defaults to channel 1.

[SENSe:]VOLTage[1|2]:RANGe:UNIT:USER[:STATe] command/query

Enables the use of engineering units (EU).

Command Syntax:

```
[SENSe:]VOLTage[1|2]:RANGe:UNIT:USER[:STATe] {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"sens:voltage2:rang:unit:user:stat OFF"
OUTPUT 711;"Voltage2:Rang:Unit:User OFF"
```

Query Syntax:

```
[SENSe:]VOLTage[1|2]:RANGe:UNIT:USER[:STATe]?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

This command allows you to specify the unit for each input channel.

When VOLT:RANG:UNIT:USER is OFF, the unit is Volts.

When VOLT:RANG:UNIT:USER is ON, the unit is an engineering unit. To define the label for the engineering unit, use the VOLT:RANG:UNIT:USER:LAB command. To specify a conversion factor, use the VOLT:RANG:UNIT:USER:SFAC command. If the channel specifier is not used, the command defaults to channel 1.

Refer to the appendix E, “Determining Units” for more information about setting engineering units.

[SENSe:]VOLTage[1|2]:RANGe[:UPPer] command/query

Specifies the input range for the selected channel.

Command Syntax:

```
[SENSe:]VOLTage[1|2]:RANGe[:UPPer]
{<number>[<unit>]}|<step>|<bound>
```

```
<number> ::= a real number (NRf data)
             limits:  -51:31.66
  <unit> ::= [DBVRMS|VPK|DBVPK|V|DBV|EU|DBEU|VRMS]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":VOLT2:RANGE 23.5198"
OUTPUT 711;"sens:voltage:rang:upp 13.6442"
```

Query Syntax:

```
[SENSe:]VOLTage[1|2]:RANGe[:UPPer]?
```

Return Format:

```
NR3
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  -5.1E+001
SCPI Compliance:  confirmed
```

Description:

This command sets the range for the input channel. Valid input ranges are from 27 through -51 dBVrms in 2 dB steps. If you send a value that is not allowed, it is rounded up to the next higher value. If you do not send specify units when you send a new value, the default unit is used, DBVRMS.

If the channel specifier is not used, the command defaults to channel 1.

To increment the value of the input range to the next higher value (+2 dB), send VOLT:RANG UP.

To decrement the value of the input range to the next lower value (-2 dB), send VOLT:RANG DOWN.

To set the input range to a value near to the amplitude of the main marker value, send VOLT[1|2]:RANG (CALC[1|2]:MARK:Y?).

Note

You can specify a new range for the channel 2 range while you are in one channel measurement mode. However, the value is not used to set the channel 2 range until you enter the two channel mode.

To determine units send the query, VOLT[1|2]:RANG? UNIT.

[SENSe:]WINDow[1|2]:EXPonential **command/query**

Specifies the time constant for the exponential window function.

Command Syntax:

```
[SENSe:]WINDow[1|2]:EXPonential
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  3.8147E-6:9.9999E6
  <unit> ::= [S]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Sense:Wind:Exponential .1"
OUTPUT 711;"WIND:EXP 2.5"
```

Query Syntax:

```
[SENSe:]WINDow[1|2]:EXPonential?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +9.999E+003
SCPI Compliance:  confirmed
```

Description:

The time constant is used to calculate the exponential decay for the exponential window according to the following formula:

$$e^{-t/\tau}$$

where
t is the position (in time) in the record length
τ is the time constant

Note The first point in the time record is always considered to be time t=0.

[SENSe:]WINDow[1|2]:FORCe command/query

Specifies the width of the force window.

Command Syntax:

```
[SENSe:]WINDow[1|2]:FORCe
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  3.8147E-6:9.9999E6
  <unit> ::= [S]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":window:forc 0.1"
OUTPUT 711;"Sense:Wind:Forc 2"
```

Query Syntax:

```
[SENSe:]WINDow[1|2]:FORCe?
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +9.999E+003
SCPI Compliance:  confirmed
```

Description:

This command specifies the length of the force window in seconds.

The force window passes the first part of the time record (specified by the length of the width of the force window) and sets the remaining part to the average value of the time record's remaining data.

Note The first point in the time record is always considered to be time t=0.

[SENSe:]WINDow[1|2]:ORDer:DC command/query

Directs the analyzer to include the DC bin in the composite power calculation (order track measurements).

Command Syntax:

```
[SENSe:]WINDow[1|2]:ORDer:DC (OFF|0|ON|1)
```

Example Statements:

```
OUTPUT 711;"SENSE:WIND:ORDER:DC OFF"
OUTPUT 711;"wind2:order:dc OFF"
```

Query Syntax:

```
[SENSe:]WINDow[1|2]:ORDer:DC?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D0 Computed Order Tracking
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

The composite power calculation (CALC:FEED 'XFR:POW:COMP') sums the power of the order spectrum at each RPM step.

To exclude the power of the order spectrum in the DC bin, send WIND:ORD:DC OFF.

The first bin is excluded from the composite power calculation if the measurement is using a uniform windowing function (WIND UNIF).

The first two bins are excluded from the calculation if the measurement is using a hanning window (WIND HANN).

The first five bins are excluded from the calculation if the measurement is using a flat top window (WIND FLAT).

Note This command is not trace specific. It ignores the trace specifier.

[SENSe:]WINDow[1|2][:TYPE] command/query

Selects the type of windowing function for the specified channel.

Command Syntax:

```
[SENSe:]WINDow[1|2][:TYPE]
{HANNing|FLATtop|UNIForm|FORCe|EXPonential|LAG|LLAG}
```

Example Statements:

```
OUTPUT 711;":Window FLATtop"
OUTPUT 711;"SENS:WIND:TYPE LAG"
```

Query Syntax:

```
[SENSe:]WINDow[1|2][:TYPE]?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  FLAT
SCPI Compliance:  confirmed
```

Description:

To select a hanning window, send WIND HANN. The beginning and end of the time record have a zero value which forces a periodic form on the data. It is commonly used to measure random noise and provides better frequency resolution.

To select a flat top window, send WIND FLAT. This window function is similar to the a hanning window, but is optimized for narrow band signals with a flatter passband. It has increased amplitude accuracy but less frequency resolution.

To select a uniform window, send WIND UNIF. The entire time record is weighted uniformly. This window function should be used for signals which may be considered self-windowing, such as transients, bursts and periodic waveforms.

To select a force window, send WIND FORC. This window function is a modified uniform window. It passes the input signal for the specified amount of time (WIND FORC) then attenuates it to the average value of the remaining data for the remainder of the time record.

To select an the exponential window, send WIND EXP. This function attenuates the input signal at a decaying exponential rate determined by the specified time constant (WIND EXP).

If you specify the force window on channel 1 and the exponential window on channel 2, the force window on channel 1 is multiplied by the exponential window. The default combination of force/exponential windowing at Preset is force window for channel 1 and exponential window for channel 2.

In **order analysis instrument mode** (INST:SEL ORD; Option 1D0) and **swept sine instrument mode** (INST:SEL SINE; Option 1D2), you may specify one of the following windowing functions:

- hanning
- flat top
- uniftorm

The windowing function is not available in **octave analysis instrument mode** (INST:SEL OCT; Option 1D1).

In **correlation analysis instrument mode** (SEL:INST CORR), this command specifies the correlation weighting function.

To select the uniform function, send WIND UNIF. The uniform weighting function (- T/2, T/2) does not suppress any part of the time record. This function should be used for signals which may be considered self-windowing, such as transients, bursts and periodic waveforms.

To select the Zero Pad 0, T/2 function, send WIND LAG. The function suppresses the last half of the time record and passes only the first half.

To select the Zero Pad -T/4, T/4 function, send WIND LLAG. The function suppresses the first quarter and the last quarter of the time record, and passes the center part of the time record (the second and third quarters).

See the *HP 35665A Concepts Guide* for additional information about correlation's weighting functions.

24

SOURce

Commands in this subsystem control the analyzer's source output. See the OUTPut subsystem for commands which enable the analyzer's source output.

SOURce:BURSt command/query

Sets the burst length for the burst source types.

Command Syntax:

```
SOURce:BURSt {<number><unit>}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:100
  <unit> ::= PCT
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"sour:burst 53.3507 PCT"
OUTPUT 711;"Sour:Burs 66.4736 PCT"
```

Query Syntax: `SOURce:BURSt?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +5.0E+001
SCPI Compliance:  instrument-specific
```

Description:

This command is used with the SOUR:FUNC BRAN and SOUR:FUNC BCH commands.

The active time of the burst cycle, "burst length," is set as a percentage of the total time record. The burst starts at the beginning of the time record.

The query returns a value in percent.

SOURce:FREQuency[:CW] command/query

Sets the frequency of the sine source.

Command Syntax:

```
SOURce:FREQuency[:CW] (<number>[<unit>])|<step>|<bound>

      <number> ::= a real number (NRf data)
                   limits:  0.0:115000.0
        <unit> ::= [HZ]
        <step> ::= UP|DOWN
       <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":SOURCE:FREQ 23210.6"
OUTPUT 711;"source:freq:cw 53914.7"
```

Query Syntax: `SOURce:FREQuency[:CW]?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1.024E+004
SCPI Compliance:  confirmed
```

Description:

This command sets the frequency of the fixed sine source type. To select a fixed sine source output, use the SOUR:FUNC SIN command.

The allowable range is 0 to 115 kHz. The frequency may be set to 15.625 mHz increments.

This is an alias for the SCPI command SOURce:FREQuency:FIXed.

SOURce:FREQuency:FIXed

command/query

Sets the frequency of the sine source type.

Command Syntax:

```
SOURce:FREQuency:FIXed (<number>[<unit>])|<step>|<bound>

           <number> ::= a real number (NRf data)
                        limits:  0.0:115000.0
             <unit> ::= [HZ]
             <step> ::= UP|DOWN
            <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Source:Freq:Fixed 65147.2"
OUTPUT 711;"SOUR:FREQ:FIXED 11208.3"
```

Query Syntax: `SOURce:FREQuency:FIXed?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1.024E+004
SCPI Compliance:  confirmed
```

Description:

This command sets the frequency of the fixed sine source type. To select a fixed sine source output, use the SOUR:FUNC SIN command.

The allowable range is 0 to 115 kHz. The frequency may be set to 15.625 mHz increments.

This is an alias for the SCPI command SOURce:FREQuency[:CW].

SOURce:FUNCtion[:SHAPe] command/query

Specifies the source output.

Command Syntax:

```
SOURce:FUNCtion[:SHAPe]
{SINusoid|RANDom|BRANdom|PCHirp|BCHirp|PINK|USER}
```

Example Statements:

```
OUTPUT 711;":sour:function RANDom"
OUTPUT 711;"Sour:Func:Shape SINusoid"
```

Query Syntax:

```
SOURce:FUNCtion[:SHAPe]?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  SIN
SCPI Compliance:  confirmed
```

Description:

To select a sinusoidal waveform, send SOUR:FUNC SIN. Refer to the SOURce:FREQuency commands for information about setting the frequency for the sine waveform.

To select random noise, send SOUR:FUNC RAND. Random noise is a continuous gaussian distributed noise signal. The signal is band-limited and band-translated to concentrate the energy in the frequency span defined by the commands in the [SENse:]FREQ subsystem. This is an approved SCPI command.

To select burst random noise, send SOUR:FUNC BRAN. Burst random noise is a gaussian distributed noise signal in successive bursts. The SOUR:BURS command is used to specify the burst length as a percentage of the time record. This is an instrument- specific command.

To select periodic chirp, send SOUR:FUNC PCH. Periodic chirp is a fast sine sweep across the current frequency span. The sweep repeats with the same period as the current time record. This is an approved SCPI command.

To select burst chirp, send SOUR:FUNC BCH. Burst chirp is a fast sine sweep over the current frequency span for a portion of the time record. The SOUR:BURS command is used to specify the burst length as a percentage of the time record. This is an instrument-specific command.

To select pink noise, send SOUR:FUNC PINK. Pink noise is noise whose spectral density is inversely proportional to frequency. This is an instrument-specific command.

To select arbitrary source data, send SOUR:FUNC USER. The data must be stored in a data register specified with the SOURce:USER[:REGister] command.

SOURce:USER[:REGister] command/query

Specifies the data register which contains the data for the arbitrary source.

Command Syntax:

```
SOURce:USER[:REGister] (D1|D2|D3|D4|D5|D6|D7|D8)
```

Example Statements:

```
OUTPUT 711;"SOUR:USER:REG D5"
OUTPUT 711;"sour:user D8"
```

Query Syntax:

```
SOURce:USER[:REGister]?
```

Return Format: CHAR

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  D1
SCPI Compliance:  instrument-specific
```

Description:

This command is used with the SOUR:FUNC USER command which sets the source to output arbitrary data. Send SOUR:USER to identify the data register location of the arbitrary waveform.

Refer to the MMEMory:LOAD:TRACe, and TRACe[:DATA] commands for additional information about saving arbitrary source data.

SOURce:USER:REPeat command/query

Enables the source repeat function.

Command Syntax:

```
SOURce:USER:REPeat {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;":Sour:User:Rep OFF"
OUTPUT 711;"SOUR:USER:REP OFF"
```

Query Syntax:

```
SOURce:USER:REPeat?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

To output the arbitrary source data only during data collection of the measurement, send SOURCE:REPEAT OFF. One "waveform" is present in each measurement record.

To output the arbitrary source data continuously, send SOURCE:USER:REPEAT ON. The source "repeats" the arbitrary waveform. This is the selection at Preset.

Note

This command has *no* effect when the analyzer is in free run trigger mode (TRIG:SOUR IMM).

SOURce:VOLTage[:LEVel]:AUTO command/query

Enables the autolevel feature in swept sine measurements (INST:SEL SINE).

Command Syntax:

```
SOURce:VOLTage[:LEVel]:AUTO {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"source:volt:lev:auto OFF"
OUTPUT 711;"Sour:Voltage:Auto OFF"
```

Query Syntax:

```
SOURce:VOLTage[:LEVel]:AUTO?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

If autolevel is enabled with the SOUR:VOLT:AUTO ON command, the analyzer adjusts the source output level to keep the amplitude of an input channel within a specified range.

See the following commands for additional information about setting the range:

- SOURce:VOLTage:LIMit[:AMPLitude]
- SOURce:VOLTage:LIMit:INPut
- SOURce:VOLTage[:LEVel]:REFerence
- SOURce:VOLTage[:LEVel]:REFerence:CHANnel
- SOURce:VOLTage[:LEVel]:REFerence:TOLerance

To turn off the autolevel function, send SOUR:VOLT:AUTO OFF.

Refer to the *HP 35665A Concepts Guide* for detailed information about the autolevel feature.

SOURce:VOLTage[:LEVel][:IMMediate][:AMPLitude] command/query

Specifies the source output level.

Command Syntax:

```
SOURce:VOLTage[:LEVel][:IMMediate][:AMPLitude]
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:13.9794
  <unit> ::= [DBVRMS|VPK|DBVPK|V|DBV|VRMS]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":SOUR:VOLTAGE 0 DBV"
OUTPUT 711;"sour:voltage:lev:imm:amplitude 5 VPK"
```

Query Syntax: `SOURce:VOLTage[:LEVel][:IMMediate][:AMPLitude]?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance: confirmed
```

Description:

The source output level can be expressed in terms of peak values, Vpk, or rms values, Vrms. If the source output level is specified in DBVRMS, the smallest non-zero level is -74.912 DBVRMS.

To determine if the output level is set to PEAK or RMS values, send SOUR:VOLT? UNIT.

SOURce:VOLTage[:LEVel]:REFerence command/query

Specifies the amplitude of the reference input channel for the autolevel feature in swept sine measurements (INST:SEL SINE).

Command Syntax:

```
SOURce:VOLTage[:LEVel]:REFerence
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:   -69.276:31.66
  <unit> ::= [DBVRMS|VPK|DBVPK|V|DBV|EU|DBEU|VRMS]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Sour:Voltage:Lev:Ref .1 VPK"
OUTPUT 711;"SOURCE:VOLT:REFERENCE 2 VRMS"
```

Query Syntax:

```
SOURce:VOLTage[:LEVel]:REFerence?
```

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +2.0 VPK
SCPI Compliance:  instrument-specific
```

Description:

This command specifies the amplitude which the analyzer tries to maintain for the input reference channel when autolevel is enabled (SOUR:VOLT:AUTO ON).

SOURce:VOLTage[:LEVel]:REFerence:CHANnel command/query

Selects the reference input channel for the autolevel feature in swept sine measurements (INST:SEL SINE).

Command Syntax:

```
SOURce:VOLTage[:LEVel]:REFerence:CHANnel (INPut1|INPut2)
```

Example Statements:

```
OUTPUT 711;":sour:volt:reference:chan INPut1"
OUTPUT 711;"Source:Volt:Lev:Reference:Chan INPut2"
```

Query Syntax:

```
SOURce:VOLTage[:LEVel]:REFerence:CHANnel?
```

Return Format: CHAR

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  INP2
SCPI Compliance:  instrument-specific
```

Description:

This command specifies which input channel the analyzer monitors when the autolevel is enabled (SOUR:VOLT:AUTO ON). The analyzer adjusts the source output to keep the amplitude of this input channel within the range specified with the SOURce:VOLTage[:LEVel]:REFerence.

SOURce:VOLTage[:LEVel]:REFerence:TOLerance command/query

Specifies the sensitivity of the autolevel algorithm in swept sine measurements (INST:SEL SINE).

Command Syntax:

```
SOURce:VOLTage[:LEVel]:REFerence:TOLerance
(<number>[<unit>])|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0.1:20
  <unit> ::= [DB]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"SOURCE:VOLT:LEV:REFERENCE:TOL 8.92904"
OUTPUT 711;"source:volt:ref:tolerance 8.48152"
```

Query Syntax: `SOURce:VOLTage[:LEVel]:REFerence:TOLerance?`

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +2.0
SCPI Compliance:  instrument-specific
```

Description:

This command sets a tolerance band (in dB) for the autolevel algorithm.

If the amplitude of the reference input channel falls outside of the specified range (relative to the value set with the SOUR:VOLT[:LEV]:REF command), the analyzer adjusts the amplitude of the source output when set to SOUR:VOLT:AUTO ON.

SOURce:VOLTage:LIMit[:AMPLitude] command/query

Sets the maximum limit used by the autolevel algorithm to adjust the source's amplitude in swept sine measurements (INST:SEL SINE).

Command Syntax:

```
SOURce:VOLTage:LIMit[:AMPLitude]
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  -9.9e37:13.9794
  <unit> ::= [DBVRMS|VPK|DBVPK|V|DBV|VRMS]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Sour:Voltage:Lim 0 DBV"
OUTPUT 711;"SOUR:VOLTAGE:LIM:AMPLITUDE 1 VRMS"
```

Query Syntax: `SOURce:VOLTage:LIMit[:AMPLitude]?`

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +2.0
SCPI Compliance:  confirmed
```

Description:

This command controls the analyzer's autolevel algorithm in adjusting the source's output level during a sweep. The limit constrains any autolevel adjustment to the source's amplitude.

SOURce:VOLTage:LIMit:INPut command/query

Sets the maximum amplitude of the *non-reference* input channel for the autolevel feature in swept sine measurements (INST:SEL SINE).

Command Syntax:

```
SOURce:VOLTage:LIMit:INPut
{<number>[<unit>]}|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  486e-06:31.66
       <unit> ::= [DBVRMS|VPK|DBVPK|V|DBV|EU|DBEU|VRMS]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"sour:volt:limit:inp -56.5358"
OUTPUT 711;"Source:Volt:Lim:Input -44.2379"
```

Query Syntax: `SOURce:VOLTage:LIMit:INPut?`

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +2.0
SCPI Compliance:  instrument-specific
```

Description:

This command controls the amplitude of the "other" input channel during a sweep. The "other" input channel is the input channel that is *not* selected as the reference channel with the SOURce:VOLTage[:LEVel]:REFerence:CHANnel command.

The limit constrains any adjustment to the source's output attempted by autolevel.

SOURce:VOLTage:SLEW command/query

Specifies the source amplitude ramp rate in swept sine measurements (INST:SEL SINE).

Command Syntax:

```
SOURce:VOLTage:SLEW {<number>[<unit>]}|<step>|<bound>

    <number> ::= a real number (NRf data)
                 limits:  0.0:10000.0
      <unit> ::= [V/S|VPK/S|VRMS/S]
      <step> ::= UP|DOWN
     <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":SOUR:VOLTAGE:SLEW 0 VRMS/S"
OUTPUT 711;"sour:voltage:slew .1 VPK/S"
```

Query Syntax: `SOURce:VOLTage:SLEW?`

Return Format: NR3

Attribute Summary:

```
Option:  1D2 Swept Sine
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  confirmed
```

Description:

This command allows you to specify how fast the source amplitude changes when you start or stop a swept sine measurement.

STATus

Commands in this subsystem provide access to most of the HP 35665A's status reporting structures (register sets). Some of the common commands described in chapter 8 provide access to the other register sets.

Most of the commands in this subsystem are used to set bits in registers. Most of the queries are used to read registers. Decimal weights are assigned to bits according to the following formula:

$$\text{weight} = 2^n$$

where *n* is the bit number with acceptable values of 0 through 14.

To set a single register bit to 1, send the decimal weight of that bit with the command that writes the register. To set more than one bit to 1, send the *sum* of the decimal weights of all the bits. Queries that read registers always return the sum of the decimal weights of all bits that are currently set to 1.

See chapter 5, "Using Status Registers" for more information on the HP 35665A status register sets.

Note The STATus commands are listed alphabetically. Therefore, the STATus:QUESTionable:NTR command and the STATus:QUESTionable:PTR command follow the STATus:QUESTionable:LIMit commands.

STATus:DEVice:CONDition? query

Reads and clears the Device State condition register.

Query Syntax: `STATus:DEVice:CONDition?`

Example Statements:

```
OUTPUT 711;"Status:Dev:Cond?"
OUTPUT 711;"STATUS:DEV:CONDITION?"
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Device State condition register. (The decimal weight of a bit is 2^n, where n is the bit number.)

See "Device State Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of condition registers in register sets.

STATus:DEVice:ENABle — command/query

Sets and queries bits in the Device State enable register.

Command Syntax:

```
STATus:DEVice:ENABle <number>|<bound>

   <number> ::= a real number (NRf data)
                  limits:  0:32767
    <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":stat:dev:enable 4"
OUTPUT 711;"Stat:Device:Enab 1"
```

Query Syntax:

```
STATus:DEVice:ENABle?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

To set a single bit in the Device State enable register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Device State Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of enable registers in register sets.

STATus:DEVice[:EVENt]? query

Reads and clears the Device State event register.

Query Syntax: `STATus:DEVice[:EVENt]?`

Example Statements:

```
OUTPUT 711;"STAT:DEVICE:EVEN?"
OUTPUT 711;"status:dev?"
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Device State event register. (The decimal weight of a bit is 2^n, where n is the bit number.)

Note The Device State event register is automatically cleared after it is read by this query.

See "Device State Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of event registers in register sets.

STATus:DEVice:NTRansition command/query

Sets and queries bits in the Device Status negative transition register.

Command Syntax: `STATus:DEVice:NTRansition <number>|<bound>`

```
<number> ::= a real number (NRf data)
             limits:  0:32767

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Stat:Device:Ntr 4"
OUTPUT 711;"STATUS:DEV:NTR 17"
```

Query Syntax: `STATus:DEVice:NTRansition?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

To set a single bit in the Device Status negative transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Device Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of negative transition registers in register sets.

STATus:DEVice:PTRansition command/query

Sets and queries bits in the Device State positive transition register.

Command Syntax: `STATus:DEVice:PTRansition <number>|<bound>`

```
<number> ::= a real number (NRf data)
                limits:  0:32767
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"status:dev:ptransition 17"
OUTPUT 711;"Stat:Dev:Ptransition 20"
```

Query Syntax: `STATus:DEVice:PTRansition?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

To set a single bit in the Device State positive transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 1 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Device State Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of positive transition registers in register sets.

STATus:OPERation:CONDition? query

Reads the Operational Status condition register.

Query Syntax: `STATus:OPERation:CONDition?`

Example Statements:

```
OUTPUT 711;":STAT:OPERATION:COND?"
OUTPUT 711;"stat:operation:cond?"
```

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Operational Status condition register. (The decimal weight of a bit is 2^n, where n is the bit number.)

See "Operational Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of condition registers in register sets.

STATus:OPERation:ENABle command/query

Sets and queries bits in the Operational Status enable register.

Command Syntax:

```
STATus:OPERation:ENABle <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:32767

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Status:Oper:Enab 96"
OUTPUT 711;"STATUS:OPER:ENABLE 2"
```

Query Syntax:

```
STATus:OPERation:ENABle?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Operational Status enable register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

See "Operational Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of enable registers in register sets.

STATus:OPERation[:EVENt]? query

Reads and clears the Operational Status event register.

Query Syntax:

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

Example Statements:

```
OUTPUT 711;":stat:oper?"
OUTPUT 711;"Status:Oper:Event?"
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Operational Status event register. (The decimal weight of a bit is 2^n, where n is the bit number.)

Note The Operational Status event register is automatically cleared after it is read by this query.

See "Operational Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of event registers in register sets.

STATus:OPERation:NTRansition command/query

Sets and queries bits in the Operational Status negative transition register.

Command Syntax:

```
STATus:OPERation:NTRansition <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:32767

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"STAT:OPER:NTRANSITION 260"
OUTPUT 711;"stat:operation:ntr 16386"
```

Query Syntax: `STATus:OPERation:NTRansition?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Operational Status negative transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Operational Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of negative transition registers in register sets.

STATus:OPERation:PTRansition command/query

Sets bits in the Operational Status positive transition register.

Command Syntax: STATus:OPERation:PTRansition <number>|<bound>

```
<number> ::= a real number (NRf data)
             limits:  0:32767
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Stat:Operation:Ptr 1536"
OUTPUT 711;"STATUS:OPER:PTR 2048"
```

Query Syntax: STATus:OPERation:PTRansition?

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Operational Status positive transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 1 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Operational Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of positive transition registers in register sets.

STATus:PRESet command

Sets bits in most enable and transition registers to their default state.

Command Syntax: `STATus:PRESet`

Example Statements:

```
OUTPUT 711;"status:pres"
OUTPUT 711;"Status:Pres"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

STAT:PRES has the following effect on the Limit Fail and Questionable Voltage register sets:

- Sets all enable register bits to 1.
- Sets all positive transition register bits to 1.
- Sets all negative transition register bits to 0.

STATUS:PRESet has the effect of bringing all events to the second level register sets (the Device State, Questionable Status, and Operation Status) *without* creating an SRQ or reflecting events in a serial poll.

It also affects these register sets (the Device State, Questionable Status, and Operation Status) as follows:

- Sets all enable register bits to 0.
- Sets all positive transition register bits to 1.
- Sets all negative transition register bits to 0.

STAT:PRES sets all bits in the User Defined enable register to 0. It does not affect any other register.

STATus:QUEStionable:CONDition? query

Reads and clears the Questionable Status condition register.

Query Syntax: `STATus:QUEStionable:CONDition?`

Example Statements:

```
OUTPUT 711;":STAT:QUESTIONABLE:COND?"
OUTPUT 711;"status:ques:cond?"
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Questionable Status condition register. (The decimal weight of a bit is 2^n, where n is the bit number.)

See “Questionable Status Register Set” in chapter 5 for a definition of bits in the register set. See “General Status Register Model” in chapter 5 for information about the role of condition registers in register sets.

STATus:QUEStionable:ENABle command/query

Sets and queries bits in the Questionable Status enable register.

Command Syntax:

```
STATus:QUEStionable:ENABle <number>|<bound>

<number> ::= a real number (NRf data)
                 limits:  0:32767

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Status:Ques:Enable 1"
OUTPUT 711;"STAT:QUES:ENABLE 513"
```

Query Syntax: `STATus:QUEStionable:ENABle?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Status enable register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Questionable Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of enable registers in register sets.

STATus:QUEStionable[:EVENt]? query

Reads and clears the Questionable Status event register.

Query Syntax: `STATus:QUEStionable[:EVENt]?`

Example Statements:

```
OUTPUT 711;":stat:questionable?"
OUTPUT 711;"Stat:Ques:Event?"
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance: confirmed
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Questionable Status event register. (The decimal weight of a bit is 2^n, where n is the bit number.)

Note The Questionable Status event register is automatically cleared after it is read by this query.

See "Questionable Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of event registers in register sets.

STATus:QUEStionable:LIMit:CONDition? query

Reads and clears the Limit Fail condition register.

Query Syntax: `STATus:QUEStionable:LIMit:CONDition?`

Example Statements:

```
OUTPUT 711;"STAT:QUESTIONABLE:LIM:COND?"
OUTPUT 711;"status:ques:limit:cond?"
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Limit Fail condition register. (The decimal weight of a bit is 2^n, where n is the bit number.)

See "Limit Fail Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of condition registers in register sets.

STATus:QUEStionable:LIMit:ENABle command/query

Sets and queries bits in the Limit Fail enable register.

Command Syntax:

```
STATus:QUEStionable:LIMit:ENABle <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:32767
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Stat:Questionable:Lim:Enable 3"
OUTPUT 711;"STAT:QUES:LIMIT:ENAB 15"
```

Query Syntax: `STATus:QUEStionable:LIMit:ENABle?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

To set a single bit in the Limit Fail enable register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Limit Fail Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of enable registers in register sets.

STATus:QUEStionable:LIMit[:EVENt]? query

Reads and clears the Limit Fail event register.

Query Syntax: `STATus:QUEStionable:LIMit[:EVENt]?`

Example Statements:

```
OUTPUT 711;"status:ques:lim:event?"
OUTPUT 711;"Stat:Questionable:Lim?"
```

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Limit Fail event register. (The decimal weight of a bit is 2^n, where n is the bit number.)

Note The Limit Fail event register is automatically cleared after it is read by this query.

See "Limit Fail Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of event registers in register sets.

STATus:QUEStionable:LIMit:NTRansition command/query

Sets and queries bits in the Limit Fail negative transition register.

Command Syntax: `STATus:QUEStionable:LIMit:NTRansition <number>|<bound>`

```
<number> ::= a real number (NRf data)
             limits:  0:32767
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":STAT:QUESTIONABLE:LIM:NTRANSITION 2"
OUTPUT 711;"stat:ques:limit:ntr 7"
```

Query Syntax: `STATus:QUEStionable:LIMit:NTRansition?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

To set a single bit in the Limit Fail negative transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Limit Fail Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of negative transition registers in register sets.

STATus:QUEStionable:LIMit:PTRansition command/query

Sets queries bits in the Limit Fail positive transition register.

Command Syntax: STATus:QUEStionable:LIMit:PTRansition <number>|<bound>

```
<number> ::= a real number (NRf data)
             limits:  0:32767
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Status:Ques:Lim:Ptransition 4"
OUTPUT 711;"STAT:QUESTIONABLE:LIM:PTR 12"
```

Query Syntax: STATus:QUEStionable:LIMit:PTRansition?

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

To set a single bit in the Limit Fail positive transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 1 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Limit Fail Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of positive transition registers in register sets.

STATus:QUEStionable:NTRansition command/query

Sets and queries bits in the Questionable Status negative transition register.

Command Syntax:

```
STATus:QUEStionable:NTRansition <number>|<bound>

   <number> ::= a real number (NRf data)
                  limits:  0:32767
    <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":status:ques:ntransition 1"
OUTPUT 711;"Stat:Ques:Ntransition 256"
```

Query Syntax:

```
STATus:QUEStionable:NTRansition?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Status negative transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Questionable Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of negative transition registers in register sets.

STATus:QUEStionable:PTRansition command/query

Sets and queries bits in the Questionable Status positive transition register.

Command Syntax:

```
STATus:QUEStionable:PTRansition <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:32767

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"STAT:QUESTIONABLE:PTR 256"
OUTPUT 711;"stat:questionable:ptr 768"
```

Query Syntax: `STATus:QUEStionable:PTRansition?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Status positive transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 1 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Questionable Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of positive transition registers in register sets.

STATus:QUEStionable:VOLTage:CONDition? query

Reads the Questionable Voltage condition register.

Query Syntax: `STATus:QUEStionable:VOLTage:CONDition?`

Example Statements:

```
OUTPUT 711;":Status:Ques:Volt:Condition?"
OUTPUT 711;"STAT:QUESTIONABLE:VOLT:COND?"
```

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Questionable Voltage condition register. (The decimal weight of a bit is 2^n, where n is the bit number.)

See “Questionable Voltage Register Set” in chapter 5 for a definition of bits in the register set. See “General Status Register Model” in chapter 5 for information about the role of condition registers in register sets.

STATus:QUEStionable:VOLTage:ENABle command/query

Sets and queries bits in the Questionable Voltage enable register.

Command Syntax:

```
STATus:QUEStionable:VOLTage:ENABle <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:32767

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"status:ques:voltage:enab 1"
OUTPUT 711;"Stat:Questionable:Volt:Enable 3"
```

Query Syntax:

```
STATus:QUEStionable:VOLTage:ENABle?
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Voltage enable register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Questionable Voltage Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of enable registers in register sets.

STATus:QUEStionable:VOLTage[:EVENt]? query

Reads and clears the Questionable Voltage event register.

Query Syntax: `STATus:QUEStionable:VOLTage[:EVENt]?`

Example Statements:

```
OUTPUT 711;":STAT:QUES:VOLTAGE?"
OUTPUT 711;"stat:questionable:volt:even?"
```

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance: confirmed
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the Questionable Voltage event register. (The decimal weight of a bit is 2^n, where n is the bit number.)

Note The Questionable Voltage event register is automatically cleared after it is read by this query.

See "Questionable Voltage Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of event registers in register sets.

STATus:QUEStionable:VOLTage:NTRansition command/query

Sets and queries bits in the Questionable Voltage negative transition register.

Command Syntax:

```
STATus:QUEStionable:VOLTage:NTRansition <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:32767

 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Status:Ques:Voltage:Ntr 2"
OUTPUT 711;"STAT:QUESTIONABLE:VOLT:NTRANSITION 1"
```

Query Syntax: `STATus:QUEStionable:VOLTage:NTRansition?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Voltage negative transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Questionable Voltage Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of negative transition registers in register sets.

STATus:QUEStionable:VOLTage:PTRansition command/query

Sets bits in the Questionable Voltage positive transition register.

Command Syntax:

```
STATus:QUEStionable:VOLTage:PTRansition <number>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:32767
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":stat:ques:voltage:ptr 2"
OUTPUT 711;"Status:Ques:Volt:Ptransition 3"
```

Query Syntax:

```
STATus:QUEStionable:VOLTage:PTRansition?
```

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Voltage positive transition register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 1 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "Questionable Voltage Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of positive transition registers in register sets.

STATus:USER:ENABle command/query

Sets and queries bits in the User Status enable register.

Command Syntax:

```
STATus:USER:ENABle <number>|<bound>

    <number> ::= a real number (NRf data)
                    limits:  0:32767

     <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"STAT:USER:ENAB 3272"
OUTPUT 711;"stat:user:enab 23996"
```

Query Syntax: `STATus:USER:ENABle?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

To set a single bit in the User Status enable register to 1, send the bit's decimal weight with this command. To set more than one bit to 1, send the sum of the decimal weights of all the bits. (The decimal weight of a bit is 2^n, where n is the bit number.)

All bits are initialized to 0 when the analyzer is turned on. However, the current setting of bits is *not* modified when you send the *RST command.

See "User Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of enable registers in register sets.

STATus:USER[:EVENt]? query

Reads and clears the User Status event register.

Query Syntax:

```
STATus:USER[:EVENt]?
```

Example Statements:

```
OUTPUT 711;":Status:User?"
OUTPUT 711;"STAT:USER:EVEN?"
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the sum of the decimal weights of all bits currently set to 1 in the User Status event register. (The decimal weight of a bit is 2^n, where n is the bit number.)

Note The User Status event register is automatically cleared after it is read by this query.

See "User Status Register Set" in chapter 5 for a definition of bits in the register set. See "General Status Register Model" in chapter 5 for information about the role of event registers in register sets.

STATus:USER:PULSe command

Sets bits in the User Status event register.

Command Syntax:

```
STATus:USER:PULSe <number>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  0:32767

      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"status:user:puls 17664"
OUTPUT 711;"Status:User:Pulse 1856"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Each bit in the User Status event register is set to 1 when you send the bit's decimal weight with the STAT:USER:PULS command. (The decimal weight of a bit is 2^n, where n is the bit number.)

See the "User Status Register Set" in chapter 5 for more information.

26

SYSTem

Commands in this subsystem are not related to analyzer performance. Instead, the SYSTem commands control global functions such as instrument preset, time and date.

SYSTem:BEEPer[:IMMediate] command

Sets the frequency and duration for the analyzer's beeper.

Command Syntax: `SYSTem:BEEPer[:IMMediate] <frequency>,<time>,<volume>`

```
<frequency> ::= [<number>|<bound>]
   <number> ::= a real number (NRf data)
                limits:  0:20000
    <bound> ::= MAX|MIN
     <time> ::= [<number>|<bound>]
   <number> ::= a real number (NRf data)
                limits:  0:20000
    <bound> ::= MAX|MIN
   <volume> [<number>|<bound>]
   <number> ::= a real number (NRf data)
                limits:  0:20000
    <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":SYST:BEEP 5729.99, 18946.4, 8522.73"
OUTPUT 711;"system:beep:immediate 9885.9, 14049.5,
5164.45"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance: confirmed
```

Description:

The frequency is specified in Hertz. The duration is specified in seconds. The volume parameter is accepted but is not used.

This command also activates the analyzer's beeper.

SYSTem:BEEPer:STATe command/query

Enables the analyzer's beeper.

Command Syntax:

```
SYSTem:BEEPer:STATe {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"Syst:Beep:State ON"
OUTPUT 711;"SYST:BEEPER:STAT ON"
```

Query Syntax:

```
SYSTem:BEEPer:STATe?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

When the beeper is enabled, it emits an audible tone when some messages are either displayed or placed in the error queue. It also emits an audible tone when a trace falls outside its specified limits if limit testing and the limit-fail beeper is turned on (CALCulate:LIMit:STATe ON and CALCulate:LIMit:BEEP ON).

SYSTem:COMMunicate:GPIB:ADDRess command/query

Sets the analyzer's HP-IB address.

Command Syntax:

```
SYSTem:COMMunicate:GPIB:ADDRess
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)
             limits:  0:30
  <unit> ::=
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":syst:communicate:gpib:address 0"
OUTPUT 711;"Syst:Comm:Gpib:Addr 3"
```

Query Syntax: `SYSTem:COMMunicate:GPIB:ADDRess?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

The analyzer's address is saved in non-volatile memory, so it is retained when you turn the analyzer off and on.

Note

When you use this command, wait at least 5 seconds before sending another command to the new address.

SYSTem:DATE command/query

Sets the date in the analyzer's battery-backed clock.

Command Syntax: `SYSTem:DATE <year>,<month>,<day>`

```
    <year> ::= [<number>|<bound>]
  <number> ::= a real number (NRf data)
               limits:  0:9999
   <bound> ::= MAX|MIN
   <month> ::= [<number>|<bound>]
  <number> ::= a real number (NRf data)
               limits:  1:12
   <bound> ::= MAX|MIN
     <day> [<number>|<bound>]
  <number> ::= a real number (NRf data)
               limits:  1:31
   <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"SYSTEM:DATE 1991, 9, 27"
OUTPUT 711;"syst:date 1991, 12, 25"
```

Query Syntax: `SYSTem:DATE?`

Return Format: NR1, NR1, NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

You must enter the year as a four-digit number, including century and millennium information (1991, not 91).

SYSTem:ERRor? query

Returns one error message from the analyzer's error queue.

Query Syntax: `SYSTem:ERRor?`

Example Statements:

```
OUTPUT 711;":Syst:Error?"
OUTPUT 711;"SYST:ERR?"
```

Return Format: `NR1, STRING`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

The error queue temporarily stores up to 5 error messages. When you send the SYST:ERR query, one message is moved from the error queue to the output queue so your controller can read the message. The error queue delivers messages to the output queue in the order received.

If more than 5 error messages are reported before any are read from the queue, the oldest error messages are saved. The last error message indicates too many error messages were received for the queue.

Note The error queue is cleared when you turn on the analyzer and when you send the *CLS command.

SYSTem:FLOG:CLEar command

Clears the fault log of all entries.

Command Syntax: `SYSTem:FLOG:CLEar`

Example Statements:

```
OUTPUT 711;"system:flog:clear"
OUTPUT 711;"Syst:Flog:Clear"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The fault log lists any hardware failures. This command deletes all lines in the fault log.

If any test fails, contact your local Hewlett-Packard sales and service office or have a qualified service technician refer to the *HP 35665A Service Guide.*

SYSTem:KEY command/query

Writes or queries front-panel key presses.

Command Syntax:

```
SYSTem:KEY <keycode>

<keycode> ::= <number>|<step>|<bound>
 <number> ::= a real number (NRf data)
              limits:  0:255
  <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":SYST:KEY 21"
OUTPUT 711;"syst:key 53"
```

Query Syntax: `SYSTem:KEY?`

Return Format: NR1

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The query returns the keycode for the last key pressed.

Sending the command with a keycode simulates pressing of that front-panel key. See table 26-1 for the front-panel keycodes.

*RST clears the queue of keys.

Table 26-1. Front Panel Keycodes

DISPLAY	Scale	1
	Disp Format	2
	Analys	3
	Trace Coord	4
	Active Trace	5
	Meas Data	6
MEASUREMENT	Window	7
	Freq	8
	Inst Mode	9
	Trigger	10
	Source	11
	Input	12
	Avg	13
	Pause/Cont	14
	Start	15
SYSTEM	System Utility	16
	Disk Utility	17
	Save/Recall	18
	Help	19
	Preset	20
	Local/HP-IB	21
	Plot/Print	22
	BASIC	23
SOFTKEYS	F1	46
	F2	47
	F3	48
	F4	49
	F5	50
	F6	51
	F7	52
	F8	53
	F9	54
	F10	55

Table 26-1. Front Panel Keycodes (continued)

Numeric Keypad	0	30
	1	31
	2	29
	3	27
	4	32
	5	38
	6	36
	7	39
	8	37
	9	35
	. (decimal point)	28
	+/-	26
	Back Space	24
	Marker Value	25
	↑	33
	↓	34
	Marker	40
	Marker Fctn	43

SYSTem:KLOCk command/query

Disables the keyboard.

Command Syntax:

```
SYSTem:KLOCk {OFF|0|ON|1}
```

Example Statements:

```
OUTPUT 711;"System:Kloc OFF"
OUTPUT 711;"SYSTEM:KLOC ON"
```

Query Syntax:

```
SYSTem:KLOCk?
```

Return Format:

```
NR1
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

This command allows your controller to disable the keyboard. This provides local lockout capability during the running of HP Instrument BASIC programs.

The query returns 1 if the keyboard is disabled.

SYSTem:PRESet command

Returns most of the analyzer's parameters to their preset states.

Command Syntax:

```
SYSTem:PRESet
```

Example Statements:

```
OUTPUT 711;":syst:preset"
OUTPUT 711;"Syst:Preset"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance: confirmed
```

Description:

In addition to returning parameters to their preset states, this command does all of the following:

- Aborts any HP-IB operations.
- Cancels any pending *OPC command or query.
- Clears the error queue.
- Clears all event registers (sets all bits to 0).

Note

The preset state of each parameter is listed under the Attribute Summary of the associated command.

This command is equivalent to the front panel **Preset** hardkey.

SYST:PRES does not affect the following parameters:

- The state of the Power-on Status Clear flag.
- The state of all enable and transition registers.
- The HP-IB input and output queues.
- The time and date (SYST:TIME and SYST:DATE).
- The HP-IB address settings (SYST:COMM:GPIB:ADDR, HCOP:PLOT:ADDR, HCOP:PRIN:ADDR).
- The HP-IB controller capability setting.
- The default disk selection (MMEM:MSIS).
- Contents of limit, data, and waterfall registers.
- Contents of the math function and constant registers.
- Contents of the RAM disks.
- Calibration constants.
- Contents of the time capture buffer.

SYSTem:SET command/query

Transfers an instrument state between the analyzer and an external controller.

Command Syntax:

```
SYSTem:SET <STATE>

        <STATE> ::= #<byte>[<length_bytes>]<data_bytes>
         <byte> ::= one byte specifying the number of length bytes
                    to follow (ASCII-encoded)
 <length_bytes> ::= bytes specifying the number of data bytes
                    to follow (ASCII-encoded)
   <data_bytes> ::= the bytes that define an instrument state
```

Example Statements:

```
OUTPUT 711;"SYST:SET STATE"
OUTPUT 711;"system:set STATE"
```

Query Syntax:

```
SYSTem:SET?
```

Return Format:

```
definite length <STATE>
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command transfers a complete instrument state—the same information contained in a state file—between the analyzer and your controller. This allows you to store an instrument state on you controller's file system. The state cannot be altered.

When you transfer an instrument state to the analyzer, you can use either the definite or indefinite length block syntax. When the analyzer returns the state to a controller, it always uses the definite length block syntax. See "Block Data" in chapter 4 for more information.

SYSTem:TIME command/query

Sets the time in the analyzer's battery-backed clock.

Command Syntax: `SYSTem:TIME <hour>,<minute>,<second>`

```
   <hour> ::= [<number>|<bound>]
 <number> ::= a real number (NRf data)
              limits:  0:23
  <bound> ::= MAX|MIN
 <minute> ::= [<number>|<bound>]
 <number> ::= a real number (NRf data)
              limits:  0:59
  <bound> ::= MAX|MIN
 <second> ::= [<number>|<bound>]
 <number> ::= a real number (NRf data)
              limits:  0:60
  <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":System:Time 15, 5, 0"
OUTPUT 711;"SYST:TIME 9, 30, 0"
```

Query Syntax: `SYSTem:TIME?`

Return Format: `NR1, NR1, NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command sets the time using a 24-hour format. For example, 3:05 pm becomes 15:05 and is sent as SYST:TIME 15, 5, 0.

27

TEST

Most of the commands in this subsystem are used to invoke service tests. Since these tests should be used only by qualified service personnel, the commands are not described here. See the *HP 35665A Service Guide* for descriptions.

Two commands in the TEST subsystem allow you to run the long confidence test and to determine whether the test passed or failed.

TEST:LOG:CLEar command

Clears the test log.

Command Syntax: `TEST:LOG:CLEar`

Example Statements:

```
OUTPUT 711;"test:log:cle"
OUTPUT 711;"Test:Log:Cle"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The test log lists the results of the long confidence test. This command deletes all lines in the test log.

If any test fails, contact your local Hewlett-Packard sales and service office or have a qualified service technician refer to the *HP 35665A Service Guide.*

TEST:LONG command

Executes the long confidence test.

Command Syntax:	`TEST:LONG`

Example Statements:	`OUTPUT 711;":TEST:LONG"`
	`OUTPUT 711;"test:long"`

Attribute Summary:	`Option:  not applicable`
	`Overlapped:  no`
	`Preset State:  not applicable`
	`SCPI Compliance:  instrument-specific`

Description:

The long confidence test is a series of individual tests that check various analyzer functions.

The overall result of the long confidence test is available by sending TEST:LONG:RESult?.

If the long confidence test fails, contact your local Hewlett-Packard sales and service office or have a qualified service technician refer to the *HP 35665A Service Guide*.

TEST:LONG:RESult? query

Returns the overall result of the long confidence test.

Query Syntax: `TEST:LONG:RESult?`

Example Statements:

```
OUTPUT 711;"Test:Long:Res?"
OUTPUT 711;"TEST:LONG:RES?"
```

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query tells you whether or not the analyzer passed the last long confidence test. The query returns +0 if the analyzer failed, +1 if it passed.

To display the results of each test to the analyzer's screen, send the DISPlay:CONTents TTAB command. To clear the test log, send TEST:LOG:CLEar. To display the fault log table (which lists hardware failures) send DISPlay:CONTents FTAB. To clear the fault log, send SYSTem:FLOG:CLEar.

If the long confidence test fails, contact your local Hewlett-Packard sales and service office or have a qualified service technician refer to the *HP 35665A Service Guide.*

28

TRACe

This subsystem contains commands which provide access to the raw measurement data (data that has not been transformed into the current display coordinates). The commands, TRAC:DATA and TRAC:WAT:DATA, allow you to transfer measurement data between the analyzer and an external controller.

Figure 28-1 shows you the position of TRAC:DATA in the data flow. It also illustrates the difference between data available in the TRACe subsystem and the CALCulate subsystem.

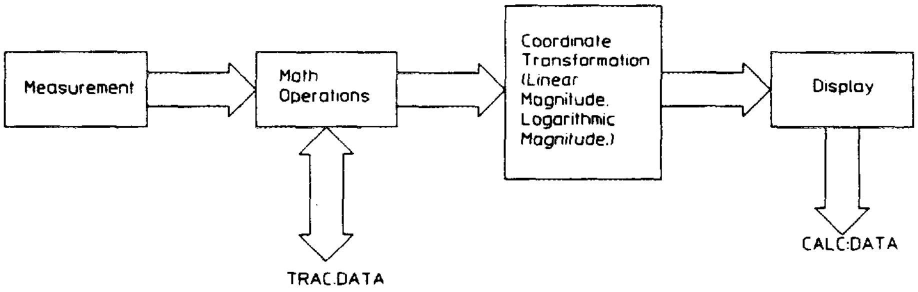

Measurement
Math Operations
Coordinate Transformation (Linear Magnitude, Logarithmic Magnitude.)
Display
TRAC:DATA
CALC:DATA

Figure 28-1. Flow of Measurement Data in the HP 35665A

After measurement data is collected, any specified math operations are performed. Data is then transformed into the specified coordinate system and sent to the display. TRAC:DATA gives you access to the raw measurement data after math operations have been performed. This data can be either complex or real. CALC:DATA gives you access to the display data—after the coordinate transformation.

Note

Both TRAC:DATA and CALC:DATA allow you to transfer measurement data *from* the analyzer. Only TRAC:DATA, however, allows you to transfer measurement data *to* the analyzer.

TRACe[:DATA] command/query

Stores data to the specified data register.

Command Syntax:

```
TRACe[:DATA] (D1|D2|D3|D4|D5|D6|D7|D8), <DATA>

        <DATA> ::= D1|D2|D3|D4|D5|D6|D7|D8
               ::= TRAC1|TRAC2
               ::= CAL1|CAL2
                   Calibration trace channel 1 or channel 2
               ::= <BLOCK> (definite and indefinite length)
               ::= <NRf>,<NRf>,<NRf>,<NRf> . . .
```

Example Statements:

```
OUTPUT 711;":trace D5, TRAC1"
OUTPUT 711;"Trac:Data D8, 1,2,3"
```

Query Syntax:

```
TRACe[:DATA]?
```

Return Format:

```
DEF_BLOCK
```

If FORMat[:DATA] REAL:

```
     <DEF_BLOCK> ::= #<byte><length_bytes><1st_value>
                     [. . .<last_value>]
          <byte> ::= one byte specifying the number of length bytes
                     to follow (ASCII encoded)
  <length_bytes> ::= number of data bytes to follow (ASCII encoded)
```

If FORMat[:DATA] ASCii:

```
     <DEF_BLOCK> ::= <1st_value>
                     [. . .<last_value>]
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command copies a selected trace into one of eight data registers, copies data between data registers, and *replaces* existing data in the specified data register with block data.

The first parameter specifies the destination. The second parameter specifies the source of the data.

This command differs from the MMEM:STOR:TRAC command. The MMEM:STOR:TRAC command saves trace data to a file. TRAC:DATA saves trace data to one of the data registers.

The query form of this command transfers data from the analyzer over the bus to your controller.

Note

Alias data is included in frequency domain data. In baseband, the first 401 points are alias protected; the remaining 112 are not. In a zoomed measurement (start frequency > 0 Hz) the first 55 points and the last 56 points are *not* alias protected.

Sending block data to a data register is valid *only* if the data register contains data. That is, you can only replace existing data in the data register. You *cannot* transfer block data to an empty data register. For more information about transferring block data, see chapter 4, "Block Data."

TRACe:WATerfall[:DATA] command/query

Stores data to the specified waterfall register.

Command Syntax:

```
TRACe:WATerfall[:DATA] (W1|W2|W3|W4|W5|W6|W7|W8), <WDATA>

       <WDATA> ::= TRAC1|TRAC2
               ::= <BLOCK> (definite and indefinite length)
               ::= <NRf>,<NRf>,<NRf>,<NRf> . . .
```

Example Statements:

```
OUTPUT 711;"TRACE:WAT:DATA W2, TRAC1"
OUTPUT 711;"trac:wat W4, 1,2,3"
```

Query Syntax: `TRACe:WATerfall[:DATA]?`

Return Format: `DEF_BLOCK`

If FORMat[:DATA] REAL:

```
   <DEF_BLOCK> ::= #<byte><length_bytes><1st__value>
                   [. . .<last_value>]
        <byte> ::= one byte specifying the number of length bytes
                   to follow (ASCII encoded)
<length_bytes> ::= number of data bytes to follow (ASCII encoded)
```

If FORMat[:DATA] ASCii:

```
   <DEF_BLOCK> ::= <1st_value>[. . .<last_value>]
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command copies a waterfall display into one of eight waterfall registers and *replaces* data in the specified waterfall register with block data. The waterfall display *must* contain more than 1 trace.

The first parameter specifies the destination. The second parameter specifies the source of the data.

This command differs from the MMEM:STOR:WAT command. The MMEM:STOR:WAT command saves a waterfall display to a file. TRAC:WAT:DATA saves a waterfall to one of the waterfall registers.

The query form of this command transfers data from the analyzer over the bus to your controller.

Note

Alias data is included in frequency domain data. In baseband, the first 401 points are alias protected; the remaining 112 are not. In a zoomed measurement (start frequency > 0 Hz) the first 55 points and the last 56 points are *not* alias protected.

Sending block data to a waterfall register is valid *only* if the waterfall register contains data. That is, you can only replace existing data in the waterfall register. You *cannot* transfer block data to an empty waterfall register. For more information about transferring block data, see chapter 4, “Block Data.”

TRACe:X[:DATA]? query

Returns the X-axis data for trace displays.

Query Syntax:

```
TRACe:X[:DATA]?
{TRACe1|TRACe2|D1|D2|D3|D4|D5|D6|D7|D8|W1|W2|W3|W4|W5|W6|W7
|W8}
```

Example Statements:

```
OUTPUT 711;":Trace:X? W1"
OUTPUT 711;"TRACE:X:DATA? W5"
```

Return Format:

```
DEF_BLOCK
```

If FORMat[:DATA] REAL:

```
        <DEF_BLOCK> ::= #<byte><length_bytes><1st_X- axis_value>
                        [. . .<last_X-axis_value>]
             <byte> ::= one byte specifying the number of length bytes
                        to follow (ASCII encoded)
     <length_bytes> ::= number of data bytes to follow (ASCII encoded)
```

If FORMat[:DATA] ASCii:

```
        <DEF_BLOCK> ::= <1st_X- axis_value>
                        [. . .<last_X-axis_value>]
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the values along the X-axis for any display. The values identify each bin in the trace.

Note

Alias data is included in frequency domain data. In baseband, the first 401 points are alias protected; the remaining 112 are not. In a zoomed measurement (start frequency > 0 Hz) the first 55 points and the last 56 points are *not* alias protected.

To determine the units for the X-axis send TRAC:X:UNIT?.

TRACe:X:UNIT? query

Returns the unit for the X-axis for trace displays.

Query Syntax:

```
TRACe:X:UNIT?
{TRACe1|TRACe2|D1|D2|D3|D4|D5|D6|D7|D8|W1|W2|W3|W4|W5|W6|W7
|W8}
```

Example Statements:

```
OUTPUT 711;"trace:x:unit? D2"
OUTPUT 711;"Trac:X:Unit? W7"
```

Return Format: STRING

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The unit for the X-axis is dependent upon the type of measurement data selected.

To query the values along the X-axis, the TRAC:X?.

TRACe:Z[:DATA]? query

Returns the Z-axis data for waterfall displays.

Query Syntax:

```
TRACe:Z[:DATA]?
(TRACe1|TRACe2|D1|D2|D3|D4|D5|D6|D7|D8
|W1|W2|W3|W4|W5|W6|W7|W8)
```

Example Statements:

```
OUTPUT 711;":TRAC:Z? D1"
OUTPUT 711;"trac:z:data? W7"
```

Return Format:

```
<DEF_BLOCK>

        <DEF_BLOCK> ::= #<byte><length_bytes><1st_Z- axis_value>
                        [. . .<last_Z-axis_value>]
             <byte> ::= one byte specifying number of length bytes
                        to follow (ASCII encoded)
     <length_bytes> ::= number of data bytes to follow (ASCII encoded)
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the values along the Z-axis in waterfall displays. The values identify each trace in a waterfall. It tells you when the measurement data was armed.

If the query is to trace box A (TRAC1), trace box B (TRAC2), or one of the waterfall registers (W1 - W8), the values are returned as an array.

Note

The array returned from a waterfall register contains a value for each waterfall step. This can be a maximum of 32767 values. The number of values returned is determined by the quantity specified with the CALC:WAT:COUN command.

If the query is to a data register (D1 - D8), a single value is returned. It is the Z-axis value of the trace saved to that data register.

To determine the units for the Z-axis, send TRAC:Z:UNIT?.

TRACe:Z:UNIT? query

Returns the unit for the Z-axis in waterfall displays.

Query Syntax:

```
TRACe:Z:UNIT?
 {TRACe1|TRACe2|D1|D2|D3|D4|D5|D6|D7|D8
|W1|W2|W3|W4|W5|W6|W7|W8}
```

Example Statements:

```
OUTPUT 711;"Trac:Z:Unit? W4"
OUTPUT 711;"TRAC:Z:UNIT? D7"
```

Return Format:

```
STRING
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The unit for the Z-axis is dependent upon the type of arming used to trigger the measurement.

The query returns "COUNTS" if manual trigger arming or automatic trigger arming was used. If the measurement data is averaged and manual or automatic arming was used, the query returns "AVG".

The query returns "SEC" if time step arming was used.

The query returns "RPM" if RPM step arming was used.

To query the values along the Z-axis, send TRAC:Z?.

TRIGger

This subsystem contains commands that control the analyzer's triggering function. See the ARM subsystem for commands that control the trigger arming functions.

Figure 29-1 shows the model for the HP 35665A's ARM-INITiate-TRIGger functions.

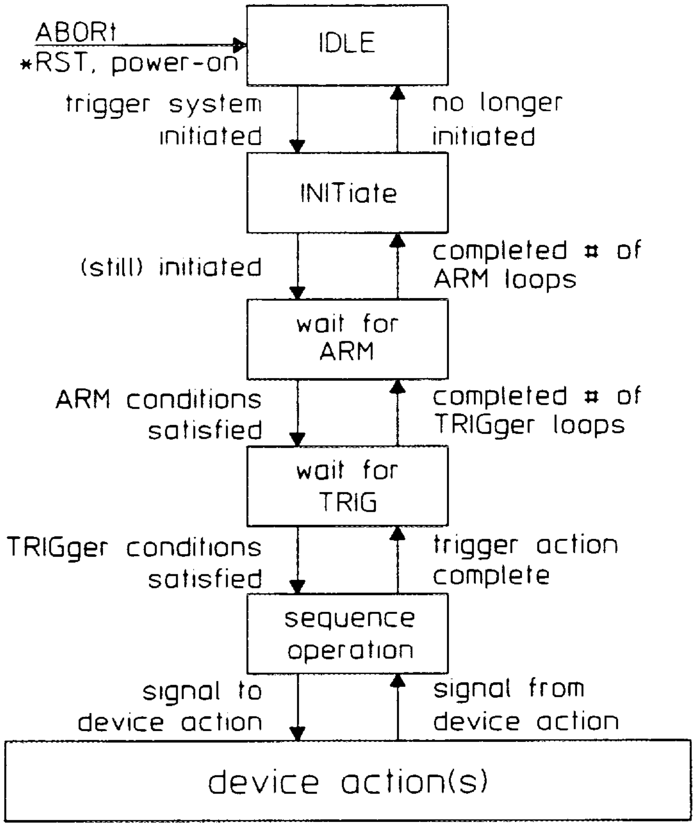


Figure 29-1. The HP 35665A's ARM-INITiate-TRIGger Functions

TRIGger[:IMMediate] command

Triggers the analyzer if TRIG:SOUR is BUS.

Command Syntax:

```
TRIGger[:IMMediate]
```

Example Statements:

```
OUTPUT 711;"TRIGGER:IMM"
OUTPUT 711;"trigger"
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

This command triggers the analyzer when the following two conditions are met:

- The HP-IB is designated as the trigger source. (See the TRIG:SOUR BUS command.)
- The analyzer is waiting to trigger. (Bit 5 of the Operational Status register must be set).

The *TRG command has the same effect as TRIG:IMM. It also has the same effect as the HP-IB bus management command Group Execute Trigger (GET).

TRIGger:LEVel command/query

Specifies the level of the input signal which causes the analyzer to trigger.

Command Syntax:

```
TRIGger:LEVel (<number>[<unit>])|<step>|<bound>

     <number> ::= a real number (NRf data)
                  limits:  -100:100
       <unit> ::= [PCT]
       <step> ::= UP|DOWN
      <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"TRIGGER:LEV .10"
OUTPUT 711;"trigger:lev -50 PCT"
```

Query Syntax: `TRIGger:LEVel?`

Return Format: `NR1`

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

You can specify the trigger level either as a percentage or as a fraction of the trigger channel's current input range. TRIG:LEV 0.25 is the same as TRIG:LEV 25 PCT.

To query the analyzer's current input range send [SENSe:]VOLT[1|2]:RANG?. To determine the units, send VOLT[1|2]:RANG? UNIT.

The trigger source must be one of the analyzer's two input channels, TRIG:SOUR INT1 or TRIG:SOUR INT2.

TRIGger:LEVel:TTL command/query

Specifies the TTL level of the signal which triggers the analyzer in order analysis and octave analysis instrument mode.

Command Syntax: `TRIGger:LEVel:TTL {HIGH|LOW}`

Example Statements:

```
OUTPUT 711;":Trig:Level:Ttl HIGH"
OUTPUT 711;"TRIGGER:LEV:TTL low"
```

Query Syntax: `TRIGger:LEVel:TTL?`

Return Format: `CHAR`

Attribute Summary:

```
Option:  1D0 Computed Order Tracking
         1D1 Realtime Octave
Overlapped:  no
Preset State:  LOW
SCPI Compliance:  instrument-specific
```

Description:

Use this command to trigger the analyzer when in order analysis instrument mode (INST:SEL ORD) or when in octave analysis instrument mode (INST:SEL OCT).

The analyzer is triggered either by a low TTL signal (LOW) or by a high TTL signal (HIGH).

This command is only valid with external triggering (TRIG:SOUR EXT).

The query returns the currently specified TTL level.

TRIGger:SLOPe **command/query**

Specifies the slope of the signal which triggers the analyzer.

Command Syntax:	`TRIGger:SLOPe {POSitive|NEGative}`

Example Statements:	`OUTPUT 711;"trigger:slop NEGative"`
	`OUTPUT 711;"Trigger:Slop NEGative"`

Query Syntax:	`TRIGger:SLOPe?`
Return Format:	CHAR

Attribute Summary:	`Option:  not applicable`
	`Overlapped:  no`
	`Preset State: POS`
	`SCPI Compliance:  confirmed`

Description:

The analyzer is triggered either by a low-to-high transition of the TTL signal (POS) or by the high-to-low transition of the TTL signal (NEG).

This command is only valid with the following selections:

- TRIG:SOUR:EXT
- TRIG:SOUR:INT1
- TRIG:SOUR:INT2

The query returns the currently specified slope.

Note

This command is *not* valid in order or octave instrument modes.
Use the TRIGger:LEVel:TTL command.

TRIGger:SOURce command/query

Selects the source of the trigger event.

Command Syntax:

```
TRIGger:SOURce
{IMMediate|EXTernal|INTernal1|INTernal2|OUTPut|BUS}
```

Example Statements:

```
OUTPUT 711;":TRIG:SOURCE OUTPut"
OUTPUT 711;"trig:source INTernal2"
```

Query Syntax:

```
TRIGger:SOURce?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  IMM
SCPI Compliance: confirmed
```

Description:

To select free run triggering send IMM. The analyzer automatically triggers as soon as it is armed.

To select the analyzer's EXT TRIG connector (on the rear panel) as the trigger source, send EXT. If EXT is selected, the analyzer is triggered either by a low-to- high transition or by a high-to-low transition of the TTL signal applied to this connector. The setting of TRIG:SLOP determines which transition triggers the analyzer. TRIG:SLOP POS is the default value.

To select one of the analyzer's input channels as the trigger source, send INT[1|2]. If the input channel specifier is not sent, the selection defaults to Channel 1. The analyzer is triggered when the selected channel's input signal matches the settings of TRIG:LEV and TRIG:SLOP.

To select the analyzer's signal source, send OUTP. The analyzer is triggered synchronously with the source.

To select the analyzer's HP-IB connector (also on the rear panel) send BUS. The analyzer is triggered when you send any of the following HP-IB commands:

- *TRG
- TRIG:IMM
- Group Execute Trigger (GET)

GET is a bus management command. See "Response to Bus Management Commands" in chapter 2 for more information.

Note

The analyzer must be waiting to trigger when it receives an external trigger signal or bus trigger command, otherwise the signal or command is ignored.
Bit 5 of the Operational Status condition register is set to 1 when the analyzer is waiting to trigger.

TRIGger:STARt[1|2] command/query

Specifies a pre-trigger value or a post-trigger value for the specified channel.

Command Syntax:

```
TRIGger:STARt[1|2] (<number>[<unit>])|<step>|<bound>

        <number> ::= a real number (NRf data)
                     limits:  -9.9e37:9.9e37
          <unit> ::= [S]
          <step> ::= UP|DOWN
         <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"Trig:Star2 -0.1"
OUTPUT 711;"TRIGGER:STAR 1.5"
```

Query Syntax: `TRIGger:STARt[1|2]?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

This command is only valid in FFT, **correlation** and **histogram** instrument modes. It is *not* valid in order analysis or swept sine instrument mode.

This command specifies the amount of time between two points: the point at which the analyzer is triggered and the point at which the specified channel starts collecting data.

Pre-trigger values are entered as a negative (–) quantity. The channel starts collecting data before the trigger point.

Post-trigger values are entered as a positive (+) quantity. The channel starts collecting data after the trigger point.

Pre-trigger and post-trigger values are specified in seconds, with resolution equal to 1 sample.

In FFT analysis and correlation analysis instrument modes:

The maximum pre-trigger value is 8191 samples. The maximum post-trigger delay is 8192 seconds. The difference between the delay specified for Channel 1 and the delay specified for Channel 2 is no larger than ± 7168 samples. Trigger resolution and sample period is:

$$\frac{1}{\text{frequency span} \times 2.56}$$

In histogram analysis instrument mode:

The maximum pre-trigger value is 3.90625 E-003 seconds for one channel and 7.8125 E-003 seconds for two channels. The maximum post-trigger delay is $(2^{32} - 1)$ points (at maximum frequency span). Trigger resolution is:

$$\frac{1}{\text{maximum frequency span} \times 2.56}$$

maximum frequency span = 102.4 kHz 1 channel
51.2 kHz 2 channel

TRIGger:TACHometer:HOLDoff command/query

Specifies a time interval in which the tachometer trigger is inhibited.

Command Syntax:

```
TRIGger:TACHometer:HOLDoff
{<number>[<unit>]}|<step>|<bound>

<number> ::= a real number (NRf data)limits:  0.0:0.052224
  <unit> ::= [S]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":trigger:tach:hold 0.0156011"
OUTPUT 711;"Trigger:Tach:Holdoff 0.0276145"
```

Query Syntax: `TRIGger:TACHometer:HOLDoff?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

This command "holds off" the trigger for the tachometer for a specified amount of time.

See the ARM:SOURce commands for additional information about RPM step arming. See the [SENSe:]ORD commands for additional information about order tracking.

TRIGger:TACHometer:LEVel — command/query

Specifies the level of the tachometer's input signal which causes the analyzer to trigger.

Command Syntax: `TRIGger:TACHometer:LEVel (<number>[<unit>])|<step>|<bound>`

```
<number> ::= a real number (NRf data)
             limits:  -25.0:+25.0
  <unit> ::= [V]
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;"TRIG:TACH:LEVEL 20.2506"
OUTPUT 711;"trig:tachometer:lev -3.13454"
```

Query Syntax: `TRIGger:TACHometer:LEVel?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +0.0
SCPI Compliance:  instrument-specific
```

Description:

The ± 4 volt range has 0.1 V resolution. The ± 20 volt range has 0.5 V resolution.

See the ARM:SOURce commands for additional information about RPM step arming. See the [SENSe:]ORD commands for additional information about order tracking.

TRIGger:TACHometer:PCOunt command/query

Specifies the number of tachometer pulses that occur in one revolution of the shaft.

Command Syntax:

```
TRIGger:TACHometer:PCOunt
<number>|<step>|<bound>

<number> ::= a real number (NRf data)
               limits:  0.5:2048.0
  <step> ::= UP|DOWN
 <bound> ::= MAX|MIN
```

Example Statements:

```
OUTPUT 711;":Trig:Tachometer:Pco 60"
OUTPUT 711;"TRIGGER:TACH:PCO 10"
```

Query Syntax: `TRIGger:TACHometer:PCOunt?`

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  +1.0
SCPI Compliance:  instrument-specific
```

Description:

See the ARM:SOURce commands for additional information about RPM step arming. See the [SENSe:]ORDer commands for additional information about order tracking.

TRIGger:TACHometer:RANGe command/query

Specifies the input range of the analyzer's tachometer.

Command Syntax:

```
TRIGger:TACHometer:RANGe {HIGH|LOW}
```

Example Statements:

```
OUTPUT 711;"trigger:tach:range LOW"
OUTPUT 711;"Trig:Tach:Range LOW"
```

Query Syntax:

```
TRIGger:TACHometer:RANGe?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  LOW
SCPI Compliance:  instrument-specific
```

Description:

To specify the input ranges as ± 4 volts, send LOW.

To specify the input range as ± 20 volts, send HIGH.

See the ARM:SOURce commands for additional information about RPM step arming. See the [SENSe:]ORD commands for additional information about order tracking.

TRIGger:TACHometer[:RPM]? query

Reads and returns the current RPM value for the analyzer's tachometer.

Query Syntax: `TRIGger:TACHometer[:RPM]?`

Example Statements:

```
OUTPUT 711;":TRIG:TACHOMETER?"
OUTPUT 711;"trig:tach:rpm?"
```

Return Format: NR3

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

TRIGger:TACHometer:SLOPe command/query

Specifies the slope of the tachometer input signal to be used in RPM step arming or order tracking.

Command Syntax:

```
TRIGger:TACHometer:SLOPe {POSitive|NEGative}
```

Example Statements:

```
OUTPUT 711;"Trig:Tachometer:Slop POSitive"
OUTPUT 711;"TRIG:TACHOMETER:SLOP NEGative"
```

Query Syntax:

```
TRIGger:TACHometer:SLOPe?
```

Return Format:

```
CHAR
```

Attribute Summary:

```
Option:  not applicable
Overlapped:  no
Preset State:  POS
SCPI Compliance:  instrument-specific
```

Description:

To specify a rising slope (low-to-high transition), send TRIG:TACH:SLOP POS.

To specify a falling slope (high-to-low transition), send TRIG:TACH:SLOP NEG.

The query returns the currently specified slope.

See the ARM:SOURce commands for additional information about RPM step arming. See the [SENSe:]ORD commands for additional information about order tracking.

HP 35665A Command Summary

Introduction

This appendix contains of all the HP-IB commands recognized by the HP 35665A and a brief description. All commands have a query form unless specified as command only or query only.

The appendix lists *common commands* and then lists the *subsystem commands* in alphabetical order.

Command List

Command Commands	Form	Description
*CAL?	query only	Calibrates the analyzer and returns the result
*CLS	command only	Clears the Status Byte by emptying the error queue and clearing all event registers
*ESE	‡	Sets bits in the Standard Event enable register
*ESR?	query only	Reads and clears the Standard Event event register
*IDN?	query only	Returns a string that uniquely identifies the analyzer
*OPC	‡	Sets or queries completion of all pending overlapped commands
*OPT?	query only	Returns a string that identifies the analyzer's option configuration
*PCB	command only	Sets the pass-control-back address
*PSC	‡	Sets the state of the Power-on Status Clear flag
*RST	command only	Executes a device reset
*SRE	‡	Sets bits in the Service Request enable register
*STB?	query only	Reads the Status Byte register
*TRG	command only	Triggers the analyzer if TRIG:SOUR is BUS
*TST?	query only	Tests the analyzer hardware and returns the results
*WAI	command only	Holds off processing of subsequent commands until all preceding commands have been processed

‡ Command form; add "?" for query form.

Subsystem Commands	Form	Description
ABORt		
ABORt	command only	Stops the current measurement in progress
ARM		
ARM[:IMMediate]	command only	Arms the trigger if ARM:SOUR is MAN
ARM:RPM:INCRement	‡	Specifies the number of RPM in a step for RPM step arming
ARM:RPM:MODE	‡	Enables the Start RPM Arming qualifier
ARM:RPM:THReshold	‡	Specifies the starting RPM value
ARM:SOURce	‡	Specifies the type of arming for the analyzer's trigger
ARM:TIMer	‡	Specifies the size of the step used in time step arming
CALCulate		
CALCulate[1 | 2]:ACTive	‡	Selects the active trace
CALC[1 | 2]:CFIT:ABORt	command only	Aborts the curve fit operation
CALC[1 | 2]:CFIT:COPY	command only	Copies the synthesis table into the curve fit table
CALC[1 | 2]:CFIT:DATA	‡	Loads values into the curve fit table
CALC[1 | 2]:CFIT:DESTination	‡	Selects the data register for the results of the curve fit operation
CALC[1 | 2]:CFIT:FREQuency[:AUTO]	‡	Specifies the region included in the curve fit operation
CALC[1 | 2]:CFIT:FREQuency:STARt	‡	Specifies the start frequency for a curve fit operation over a limited frequency span
CALC[1 | 2]:CFIT:FREQuency:STOP	‡	Specifies the stop frequency for a curve fit operation over a limited frequency span
CALC[1 | 2]:CFIT:FSCale	‡	Specifies the frequency scaling used in the curve fit operation
CALC[1 | 2]:CFIT[:IMMediate]	command only	Starts the curve fit process
CALC[1 | 2]:CFIT:ORDer:AUTO	‡	Determines the operation of the curve fitter
CALC[1 | 2]:CFIT:ORDer:POLes	‡	Specifies the number of poles used in the curve fit operation
CALC[1 | 2]:CFIT:ORDer:ZERos	‡	Specifies the number of zeros used in the curve fit operation
CALC[1 | 2]:CFIT:TDELay	‡	Specifies a time delay value for the curve fit operation

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
CALC[1 | 2]:CFIT:WEIGht:AUTO	‡	Determines the weighting function used in the curve fit operation
CALC[1 | 2]:CFIT:WEIGht:REGister	‡	Selects the data register which contains the weighting function for the curve fit operation
CALC[1 | 2]:DATA?	query only	Returns trace data that has been transformed to the currently selected coordinate transform
CALC[1 | 2]:DATA:HEADer:POINts?	query only	Returns the number of points in the data block returned with the CALC:DATA? query
CALC[1 | 2]:FEED	‡	Selects the measurement data to be displayed in the specified trace
CALC[1 | 2]:FORMat	‡	Selects a coordinate system for displaying measurement data and for transferring coordinate transformed data to a controller
CALC[1 | 2]:LIMit:BEEP[:STATe]	‡	Turns the limit-fail beeper on and off
CALC[1 | 2]:LIMit:FAIL?	query only	Returns the result of the last limit test; 0 for pass or 1 for fail
CALC[1 | 2]:LIMit:LOWer:CLEar[:IMMediate]	command only	Deletes the lower limit line from the specified display
CALC[1 | 2]:LIMit:LOWer:MOVE:Y	command only	Moves all segments of the lower limit line up or down in the specified trace
CALC[1 | 2]:LIMit:LOWer:REPort[:DATA]?	query only	Returns the X-axis value of the failed points for the lower limit test
CALC[1 | 2]:LIMit:LOWer:REPort:YDATa?	query only	Returns the Y-axis value of the failed points for the lower limit test
CALC[1 | 2]:LIMit:LOWer:SEGMent	‡	Defines the lower limit as a series of line segments in the specified display
CALC[1 | 2]:LIMit:LOWer:SEGMent:CLEar	command only	Deletes a segment from the lower limit line
CALC[1 | 2]:LIMit:LOWer:TRACe[:IMMediate]	command only	Converts the specified trace into a lower limit line
CALC[1 | 2]:LIMit:STATe	‡	Turns limit testing on and off for the specified trace
CALC[1 | 2]:LIMit:UPPer:CLEar[:IMMediate]	command only	Deletes the upper limit line from the specified display
CALC[1 | 2]:LIMit:UPPer:MOVE:Y	command only	Moves all segments of the upper limit line up or down in the specified trace box
CALC[1 | 2]:LIMit:UPPer:REPort[:DATA]?	query only	Returns the X-axis value of the failed points for the upper limit test
CALC[1 | 2]:LIMit:UPPer:REPort:YDATa?	query only	Returns the Y-axis value of the failed points for the upper limit test
CALC[1 | 2]:LIMit:UPPer:SEGMent	‡	Defines the upper limit as a series of line segments in the specified display

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
CALC[1 | 2]:LIMit:UPPer:SEGMent:CLEar	command only	Deletes a segment from the upper limit line
CALC[1 | 2]:LIMit:UPPer:TRACe[:IMMediate]	command only	Converts the specified trace into an upper limit line
CALC[1 | 2]:MARKer:BAND:STARt	‡	Specifies the lowest frequency of the band in which power is calculated
CALC[1 | 2]:MARKer:BAND:STOP	‡	Specifies the highest frequency of the band in which power is calculated
CALC[1 | 2]:MARKer:COUPled[:STATe]	‡	Couples the marker on the inactive trace with the marker of the active trace
CALC[1 | 2]:MARKer:FUNCtion	‡	Selects one of the analyzer's marker functions
CALC[1 | 2]:MARKer:FUNCtion:RESult?	query only	Returns the result of the calculation for the currently selected marker function
CALC[1 | 2]:MARKer:HARMonic:COUNt	‡	Specifies the maximum number of harmonic markers for the display
CALC[1 | 2]:MARKer:HARMonic:FUNDamental	‡	Specifies the fundamental frequency for harmonic markers and calculations
CALC[1 | 2]:MARKer:MAXimum[:GLOBal]	command only	Moves the main marker to the highest point on the specified trace
CALC[1 | 2]:MARK:MAX[:GLOBal]:TRACk	‡	Turns the peak tracking function on or off
CALC[1 | 2]:MARKer:MAXimum:LEFT	command only	Moves the main marker one peak to the left of its current location on the specified trace
CALC[1 | 2]:MARKer:MAXimum:RIGHt	command only	Moves the main marker one peak to the right of its current location on the specified trace
CALC[1 | 2]:MARKer:MODE	‡	Selects absolute or relative marker values
CALC[1 | 2]:MARKer:POSition	‡	Specifies the main marker's independent axis position
CALC[1 | 2]:MARKer:POSition:POINt	‡	Moves the main marker to a specific display point
CALC[1 | 2]:MARKer:REFerence:X	‡	Specifies the marker reference X-axis position
CALC[1 | 2]:MARKer:REFerence:Y	‡	Specifies the marker reference Y-axis position
CALC[1 | 2]:MARKer:SIDeband:CARRier	‡	Specifies the carrier frequency used for sideband markers and calculations
CALC[1 | 2]:MARKer:SIDeband:COUNt	‡	Specifies the number of sideband markers for the display
CALC[1 | 2]:MARKer:SIDeband:INCRement	‡	Specifies the frequency increment (or delta) between sideband markers
CALC[1 | 2]:MARKer[:STATe]	‡	Turns on the main markers or turns off all markers and marker functions for the active trace
CALC[1 | 2]:MARKer:X[:ABSolute]	‡	Specifies the main marker's X-axis position

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
CALC[1 | 2]:MARKer:X:RELative	‡	Specifies the marker reference X-axis position relative to the main marker
CALC[1 | 2]:MARKer:Y[:ABSolute]?	query only	Returns the main marker's Y-axis position
CALC[1 | 2]:MARKer:Y:RELative	‡	Specifies the marker reference Y-axis position relative to the main marker
CALC[1 | 2]:MATH:CONStant[1 | 2 | 3 | 4 | 5]	‡	Defines the value of one of the constant registers
CALC[1 | 2]:MATH:DATA	‡	Loads a complete set of math definitions
CALC[1 | 2]:MATH[:EXPRession[1 | 2 | 3 | 4 | 5]]	‡	Defines a math function
CALC[1 | 2]:MATH:SELect	‡	Selects and displays the designated math function
CALC[1 | 2]:MATH:STATe	‡	Evaluates the currently selected math operation for the specified trace and displays the results
CALC[1 | 2]:SYNThesis:COPY	command only	Copies the contents of the curve fit table into the synthesis table
CALC[1 | 2]:SYNThesis:DATA	‡	Loads values into the synthesis table
CALC[1 | 2]:SYNThesis:DESTination	‡	Selects the data register for the results of the synthesis operation
CALC[1 | 2]:SYNThesis:FSCale	‡	Specifies a frequency scale for the synthesis operation
CALC[1 | 2]:SYNThesis:GAIN	‡	Specifies the gain constant
CALC[1 | 2]:SYNThesis[:IMMediate]	command only	Creates a frequency response curve from the synthesis table
CALC[1 | 2]:SYNThesis:SPACing	‡	Specifies a linear or logarithmic scale for the X-axis data spacing
CALC[1 | 2]:SYNThesis:TDELay	‡	Specifies a time delay value for the synthesis operation
CALC[1 | 2]:SYNThesis:TTYPe	‡	Converts the synthesis table to another table format
CALC[1 | 2]:UNIT:AMPLitude	‡	Selects the unit of amplitude for the Y-axis scale
CALC[1 | 2]:UNIT:ANGLe	‡	Specifies the unit for phase coordinates
CALC[1 | 2]:UNIT:VOLTage	‡	Selects the vertical unit for the specified display's Y-axis
CALC[1 | 2]:WATerfall:COUNt	‡	Specifies the number of traces stored for a waterfall display
CALC[1 | 2]:WATerfall[:DATA]?	query only	Returns waterfall data that has been transformed to the currently selected coordinate transform
CALC[1 | 2]:WATerfall:SLICe:COPY	command only	Copies the selected waterfall slice to the designated data register

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
CALC[1 | 2]:WATerfall:SLICe:SELect	‡	Selects the waterfall slice at the specified X-axis position
CALC[1 | 2]:WATerfall:SLICe:SELect:POINt	‡	Selects a waterfall slice by its display point value
CALC[1 | 2]:WATerfall:TRACe:COPY	command only	Saves the selected trace to the specified data register
CALC[1 | 2]:WATerfall:TRACe:SELect	‡	Selects a waterfall trace by its Z-axis value
CALC[1 | 2]:WATerfall:TRACe:SELect:POINt	‡	Selects a waterfall trace by its step value
CALibration[:ALL]?	query only	Calibrates the analyzer and returns the result
CALibration:AUTO	‡	Calibrates the analyzer or sets the state of the autocalibration function
DISPlay		
DISPlay:ANNotation	‡	Turns the display of screen annotation on or off
DISP:BODE	command only	Selects a Bode diagram format
DISP:CONTents	‡	Specifies what is displayed on the analyzer's screen
DISP:ENABle	‡	Enables the analyzer's display
DISP:FORMat	‡	Selects a format for displaying trace data
DISP:GPIB:ECHO	command only	Enables and disables the echoing of HP-IB mnemonics to the analyzer's screen
DISP:PROGram[:MODE]	‡	Selects the portion of the analyzer's screen to be used for HP Instrument BASIC program output
DISP:WATerfall:BASeline	‡	Specifies the percentage of each trace that is concealed in the waterfall display
DISP:WATerfall:COUNt	‡	Determines the number of traces displayed in the waterfall display
DISP:WATerfall:HEIGht	‡	Specifies the height of the waterfall trace box
DISP:WATerfall:HIDDen	‡	Turns on or off the removal of hidden waterfall traces
DISP[:WINDow[1 | 2]]:LIMit:STATe	‡	Turns limit lines on or off in the specified display
DISP[:WIND[1 | 2]]:TRAC:GRAT:GRID[:STATe]	‡	Turns the display's overlay grid on or off
DISP[:WINDow[1 | 2]]:TRACe:LABel	‡	Loads a label for the specified trace
DISP[:WIND[1 | 2]]:TRAC:LAB:DEFault[:STATe]	‡	Turns on or off the analyzer's default title for the specified trace
DISP[:WINDow[1 | 2]]:TRACe:X:MATCh[1 | 2]	command only	Modifies the X-axis scaling of a trace to match the X-axis scaling of the reference trace
DISP[:WIND[1 | 2]]:TRACe:X[:SCALe]:AUTO	‡	Scales the measurement data to fit the trace box

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
DISP[:WINDow[1 | 2]]:TRACe:X[:SCALe]:LEFT	‡	Specifies the first X-axis value on the display
DISP[:WIND[1 | 2]]:TRACe:X[:SCALe]:RIGHt	‡	Specifies the last X-axis value on the display
DISP[:WIND[1 | 2]]:TRACe:X:SPACing	‡	Specifies X-axis scaling
DISP[:WIND[1 | 2]]:TRACe:Y:MATCh[1 | 2]	command only	Modifies the Y-axis scaling of a trace to match the Y-axis scaling of the reference trace
DISP[:WIND[1 | 2]]:TRACe:Y[:SCALe]:AUTO	‡	Scales and repositions the trace vertically to provide the best display of trace data
DISP[:WIND[1 | 2]]:TRAC:Y[:SCALe]:BOTTom	‡	Specifies the value of the bottom reference point of the display's Y-axis scale
DISP[:WIND[1 | 2]]:TRACe:Y[:SCALe]:CENTer	‡	Specifies the value of the center reference point of the display's Y-axis scale
DISP[:WIND[1 | 2]]:TRAC:Y[:SCALe]:PDIVision	‡	Defines the height of each vertical division on the specified trace
DISP[:WIND[1 | 2]]:TRAC:Y[:SCAL]:REFerence	‡	Determines the Y-axis reference position for the specified display
DISP[:WINDow[1 | 2]]:TRACe:Y[:SCALe]:TOP	‡	Specifies the value of the top reference point of the display's Y-axis scale
DISP[:WINDow[1 | 2]]:TRACe:Y:SPACing	‡	Specifies scaling of the Y-axis for linear magnitude coordinate data
FORMat		
FORMat[:DATA]	‡	Specifies the data type and date encoding to be used during transfers of a data block
HCOPy		
HCOPy:DESTination	‡	Specifies where the print or plot operation is sent: either directly to the HP-IB device or to a file on the default disk
HCOP:DEVice	‡	Specifies the output device
HCOP:EJECt	‡	Turns the page-eject feature on or off
HCOP:FILename	‡	Specifies a filename for the output of a print or plot operation
HCOP[:IMMediate]	command only	Plots or prints the currently specified item
HCOP:PLOT:ADDRess	‡	Tells the analyzer which HP-IB address is assigned to your plotter
HCOP:PLOT:LTYPe:TRACe[1 | 2]	‡	Selects the line type for the specified trace
HCOP:PLOT:PEN:ALPHa	‡	Selects the pen used for plotting miscellaneous annotations

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
HCOP:PLOT:PEN:DEFault	command only	Specifies default values for the plotter pen assignments
HCOP:PLOT:PEN:GRID	‡	Selects the pen used to plot the overlay grid
HCOP:PLOT:PEN:MARKer[1 | 2]	‡	Selects the pen used to plot markers for the specified trace
HCOP:PLOT:PEN:TRACe[1 | 2]	‡	Selects the pen used to plot the specified trace and annotation
HCOP:PLOT:SPEed	‡	Specifies the plotting speed for all plotting operations initiated by the analyzer
HCOP:PRINt:ADDRess	‡	Tells the analyzer which HP-IB address is assigned to your printer
HCOP:SOURce	‡	Selects the portion of the analyzer's screen you want to plot
HCOP:TSTamp:MODE	‡	Specifies the format of the time stamp used for plotting and printing
HCOP:TSTamp[:STATe]	‡	Turns a time stamp on or off for print and plot operations
INITiate		
INITiate:CONTinuous	‡	Sets the trigger system to a continuously initiated state
INITiate[:IMMediate]	command only	Starts a measurement and forces the trigger system to exit the idle state
INPut		
INPut[1 | 2]:BIAS[:STATe]	‡	Enables/disables the ICP supply on the corresponding input channel
INPut[1 | 2]:COUPling	‡	Selects AC or DC coupling for the specified channel
INPut[1 | 2]:FILTer:AWEighting[:STATe]	‡	Enables/disables the A-weight filter on the specified input channel
INPut[1 | 2]:FILTer[:LPASs][:STATe]	‡	Enables/disables the anti-alias filter for the specified input channel
INPut[1 | 2]:LOW	‡	Sets the specified channel's input shield to float or to ground
INPut[1 | 2]:REFerence:DIRection	‡	Sets transducer direction for measurement point
INPut[1 | 2]:REFerence:POINt	‡	Sets the number for the transducer point
INPut2[:STATe]	‡	Turns two-channel measurements on or off

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
INSTrument		
INSTrument:NSELect	‡	Selects one of the analyzer's six major instrument modes
INSTrument:SELect	‡	Selects one of the analyzer's six major instrument modes
MEMory		
MEMory:CATalog[:ALL]?	query only	Returns information on the current contents and state of the analyzer's memory
MEMory:CATalog:NAME?	query only	Returns information about memory usage allocated for a specific item
MEMory:DELete:ALL	command only	Purges all allocated memory in the analyzer
MEMory:DELete[:NAME]	command only	Purges the memory allocated for a specific item
MEMory:FREE[:ALL]?	query only	Returns information on the state of the analyzer's memory
MMEMory		
MMEMory:COPY	command only	Copies the contents of one disk to another or one file to another
MMEM:DELete	command only	Deletes one file or the contents of an entire disk
MMEM:DISK:ADDRess	‡	Tells the analyzer which HP-IB address is assigned to your external disk
MMEM:DISK:UNIT	‡	Specifies the unit of the external disk drive
MMEM:FSYStem?	query only	Returns the type of file system for the default disk
MMEM:INITialize	command only	Formats the specified disk
MMEM:LOAD:CFIT	command only	Loads a curve fit table into the analyzer from a file on the specified disk
MMEM:LOAD:CONTinue	‡	Continues the load operation of time capture and waterfall files saved on multiple disks
MMEM:LOAD:LIM:LOWer:TRACe[1 | 2]	command only	Loads a lower limit for the specified trace from a file on the specified disk
MMEM:LOAD:LIM:UPPer:TRACe[1 | 2]	command only	Loads an upper limit for the specified from a file on the specified disk
MMEM:LOAD:MATH	command only	Loads a complete set of math definitions into the analyzer from the specified disk
MMEM:LOAD:PROGram	command only	Loads an HP Instrument BASIC program into the analyzer from a file on the specified disk
MMEM:LOAD:STATe	command only	Loads an instrument state into the analyzer from the specified disk

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
MMEM:LOAD:SYNThesis	command only	Loads a synthesis table into the analyzer from a file on the specified disk
MMEM:LOAD:TCAPture	command only	Loads a time capture file from the specified disk
MMEM:LOAD:TRACe	command only	Loads a trace into the analyzer from the specified disk
MMEM:LOAD:WATerfall	command only	Loads a waterfall file into the analyzer from a file on the specified disk
MMEM:MSIS	‡	Specifies a default disk
MMEM:REName	command only	Renames a file
MMEM:STORe:CFIT	command only	Stores a curve fit table to a file on the specified disk
MMEM:STORe:CONTinue	‡	Splits a large for a time capture file or a waterfall file over multiple disks
MMEM:STOR:LIM:LOWer:TRACe[1 | 2]	command only	Saves the lower limit of the specified trace to a file on the specified disk
MMEM:STOR:LIM:UPPer:TRACe[1 | 2]	command only	Saves the upper limit of the specified trace to a file on the specified disk
MMEM:STORe:MATH	command only	Saves a complete set of math definitions to a file on the specified disk
MMEM:STORe:PROGram	command only	Saves an HP Instrument BASIC program to a file on the specified disk for the first time
MMEM:STORe:STATe	command only	Saves the instrument state to a file on the specified disk
MMEM:STORe:SYNThesis	command only	Stores a synthesis table to a file on the specified disk
MMEM:STORe:TCAPture	command only	Saves the time capture buffer to a file on the specified disk
MMEM:STORe:TRACe	command only	Saves the specified trace to a file on the specified disk
MMEM:STORe:WATerfall	command only	Saves the current waterfall display to a file on the specified disk
OUTPut		
OUTPut[:STATe]	‡	Enables the analyzer's internal source

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
PROGram		
PROGram:EXPLicit:DEFine	‡	Loads an HP Instrument BASIC program into the specified program buffer from an external controller
PROG[:SELected]:DEFine	‡	Loads an HP Instrument BASIC program from an external controller into the active program buffer
PROG[:SELected]:DELete:ALL	command only	Deletes all HP Instrument BASIC programs stored in the analyzer
PROG[:SELected]:DELete[:SELected]	command only	Deletes the active HP Instrument BASIC program
PROG[:SELected]:LABel	‡	Loads a softkey label for the active HP Instrument BASIC program
PROG[:SELected]:MALLocate	‡	Allocates memory space for HP Instrument BASIC programs
PROG[:SELected]:NAME	‡	Selects an HP Instrument BASIC program
PROG[:SELected]:NUMBer	‡	Loads a new value for the specified numeric variable in the active HP Instrument BASIC program
PROG[:SELected]:STATe	‡	Selects the state of the active HP Instrument BASIC program
PROG[:SELected]:STRing	‡	Loads a new value for the specified string variable for the active HP Instrument BASIC program
[:SENSe]		
[SENSe:]AVERage:CONFidence	‡	Specifies the confidence level used in equal confidence averaging in octave measurements
[SENSe:]AVERage:COUNt	‡	Specifies a count or a weighting factor for the averaged measurement data
[SENSe:]AVERage:HOLD	‡	Specifies the type of hold used in averaging octave measurements
[SENSe:]AVERage:IMPulse	‡	Enables impulse detection in octave measurements
[SENSe:]AVERage:IRESult:RATE	‡	Specifies how often the display is updated when fast average mode is on
[SENSe:]AVERage:IRESult[:STATe]	‡	Selects fast average mode
[SENSe:]AVERage:PREView	‡	Specifies the type of preview averaging
[SENSe:]AVERage:PREView:ACCept	command only	Accept the current time record during preview averaging
[SENSe:]AVERage:PREView:REJect	command only	Reject the current time record during preview averaging

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
[SENSe:]AVERage:PREView:TIME	‡	Specifies the amount of time the analyzer waits for a response in timed preview averaging
[SENSe:]AVERage[:STATe]	‡	Turns the selected averaging function on or off
[SENSe:]AVERage:TCONtrol	‡	Specifies how the analyzer behaves after the count is reached
[SENSe:]AVERage:TIME	‡	Specifies the time period used in averaging octave measurements and histograms
[SENSe:]AVERage:TYPE	‡	Specifies the type of averaging the analyzer performs
[SENSe:]FEED	‡	Specifies the data source for a measurement; either from the input channels or from the time capture buffer
[SENSe:]FREQuency:CENTer	‡	Specifies the center frequency for the current measurement
[SENSe:]FREQuency:MANual	‡	Selects a discrete point to be measured during manual sweep mode
[SENSe:]FREQuency:RESolution	‡	Sets the resolution of the frequency steps used in swept sine instrument mode
[SENSe:]FREQuency:RESolution:AUTO	‡	Selects auto resolution for swept sine instrument mode
[SENSe:]FREQ:RESolution:AUTO:MCHange	‡	Specifies the maximum change permitted between the frequency response of the current measurement point and the frequency response of the previous measurement
[SENSe:]FREQ:RESolution:AUTO:MINimum	‡	Specifies the initial resolution of a swept sine measurement with automatic resolution
[SENSe:]FREQuency:RESolution:OCTave	‡	Specifies the type of octave measurement
[SENSe:]FREQuency:SPAN	‡	Specifies the frequency bandwidth to be measured
[SENSe:]FREQuency:SPAN:FULL	command only	Sets the analyzer to the widest frequency span available for the current instrument mode
[SENSe:]FREQuency:SPAN:LINK	‡	Specifies the frequency parameter which remains constant if frequency span or record length is modified
[SENSe:]FREQuency:STARt	‡	Specifies the start frequency for the frequency band of the current measurement
[SENSe:]FREQuency:STEP[:INCRement]	‡	Specifies the step size which is used when changing frequency parameters
[SENSe:]FREQuency:STOP	‡	Sets the stop frequency to the specified value
[SENSe:]HISTogram:BINS	‡	Specifies the number of bins in a histogram

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
[SENSe:]ORDer:MAXimum	‡	Specifies the number of orders to be displayed
[SENSe:]ORDer:RESolution	‡	Specifies order resolution
[SENSe:]ORDer:RPM:MAXimum	‡	Specifies the maximum rotational speed range you want to analyze
[SENSe:]ORDer:RPM:MINimum	‡	Specifies the minimum rotational speed range you want to analyze
[SENSe:]ORDer:TRACk[1 | 2 | 3 | 4 | 5]	‡	Specifies the order number for the selected track
[SENSe:]ORDer:TRACk[1 | 2 | 3 | 4 | 5]:STATe	‡	Selects order track or order spectrum measurements
[SENSe:]REJect:STATe	‡	Turns overload rejection on or off
[SENSe:]SWEep:DIRection	‡	Specifies the direction of the sweep
[SENSe:]SWEep:DWELl	‡	Specifies the integration time for swept sine measurements
[SENSe:]SWEep:MODE	‡	Specifies automatic or manual sweep modes
[SENSe:]SWEep:OVERlap	‡	Specifies the maximum amount of time record overlap
[SENSe:]SWEep:SPACing	‡	Selects linear or logarithmic spacing between measurement data points
[SENSe:]SWEep:STIMe	‡	Specifies the settling time for a swept sine measurement
[SENSe:]SWEep:TIME	‡	Specifies the length of the time record in seconds
[SENSe:]TCAPture:ABORt	command only	Stops the time capture process
[SENSe:]TCAPture:DELete	command only	Removes the time capture buffer
[SENSe:]TCAPture[:IMMediate]	command only	Starts the collection of data for the time capture process
[SENSe:]TCAPture:LENGth	‡	Specifies the length of the time capture buffer
[SENSe:]TCAPture:MALLocate	command only	Allocates memory for the time capture buffer
[SENSe:]TCAPture:STARt[1 | 2]	‡	Specifies the beginning of the time capture data used in a measurement
[SENSe:]TCAPture:STOP[1 | 2]	‡	Specifies the end of the time capture data used in a measurement
[SENSe:]TCAP:TACHometer:RPM:MAXimum	‡	Specifies the tachometer's maximum RPM when included in the time capture buffer
[SENSe:]TCAPture:TACHometer[:STATe]	‡	Directs the analyzer to include the tachometer input signal in the time capture buffer
[SENSe:]VOLTage[1 | 2]:RANGe:AUTO	‡	Automatically selects the best range on the specified channel for the current input signal

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
[SENSe:]VOLT[1 | 2]:RANG:UNIT:USER:LABel	‡	Assigns a name to the engineering units for the specified input channel
[SENS:]VOLT[1 | 2]:RANG:UNIT:USER:SFACtor	‡	Specifies a scaling factor for engineering units
[SENS:]VOLT[1 | 2]:RANG:UNIT:USER:STATe	‡	Enables the use engineering units
[SENSe:]VOLTage[1 | 2]:RANGe[:UPPer]	‡	Specifies the input range for the selected channel
[SENSe:]WINDow[1 | 2]:EXPonential	‡	Specifies the time constant for the exponential window function
[SENSe:]WINDow[1 | 2]:FORCe	‡	Specifies the width of the force window
[SENSe:]WINDow[1 | 2]:ORDer:DC	‡	Directs the analyzer to include the DC bin in the composite power calculation
[SENSe:]WINDow[1 | 2][:TYPE]	‡	Selects the type of windowing function for the specified channel
SOURce		
SOURce:BURSt	‡	Sets the burst length for the burst source types
SOUR:FREQuency[:CW]	‡	Sets the frequency of the sine source
SOUR:FREQuency:FIXed	‡	Sets the frequency of the sine source type
SOUR:FUNCtion[:SHAPe]	‡	Specifies the source output
SOUR:USER[:REGister]	‡	Specifies the data register which contains the data for the arbitrary source
SOUR:USER:REPeat	‡	Enables the source repeat function
SOUR:VOLTage[:LEVel]:AUTO	‡	Enables the autolevel feature in swept sine measurements
SOUR:VOLTage[:LEV][:IMMediate][:AMPLitude]	‡	Specifies the source output level
SOUR:VOLTage[:LEVel]:REFerence	‡	Specifies the amplitude of the reference input channel for the autolevel feature in swept sine measurements
SOUR:VOLTage[:LEVel]:REFerence:CHANnel	‡	Selects the reference input channel for the autolevel feature in swept sine measurements
SOUR:VOLTage[:LEVel]:REFerence:TOLerance	‡	Specifies the sensitivity of the autolevel algorithm in swept sine measurements
SOUR:VOLTage:LIMit[:AMPLitude]	‡	Sets the maximum limit used by the autolevel algorithm to adjust the source's amplitude in swept sine measurements
SOUR:VOLTage:LIMit:INPut	‡	Sets the maximum amplitude of the *non-reference* input channel for the autolevel feature in swept sine measurements
SOUR:VOLTage:SLEW	‡	Specifies the source amplitude ramp rate in swept sine measurements

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
STATus		
STATus:DEVice:CONDition?	query only	Reads and clears the Device State condition register
STAT:DEVice:ENABle	‡	Sets and queries bits in the Device State enable register
STAT:DEVice[:EVENt]?	query only	Reads and clears the Device State event register
STAT:DEVice:NTRansition	‡	Sets and queries bits in the Device Status negative transition register
STAT:DEVice:PTRansition	‡	Sets and queries bits in the Device State positive transition register
STAT:OPERation:CONDition?	query only	Reads the Operational Status condition register
STAT:OPERation:ENABle	‡	Sets and queries bits in the Operational Status enable register
STAT:OPERation[:EVENt]?	query only	Reads and clears the Operational Status event register
STAT:OPERation:NTRansition	‡	Sets and queries bits in the Operational Status negative transition register
STAT:OPERation:PTRansition	‡	Sets bits in the Operational Status positive transition register
STAT:PRESet	command only	Sets bits in most enable and transition registers to their default state
STAT:QUEStionable:CONDition?	query only	Reads and clears the Questionable Status condition register
STAT:QUEStionable:ENABle	‡	Sets and queries bits in the Questionable Status enable register
STAT:QUEStionable[:EVENt]?	query only	Reads and clears the Questionable Status event register
STAT:QUEStionable:LIMit:CONDition?	query only	Reads and clears the Limit Fail condition register
STAT:QUEStionable:LIMit:ENABle	‡	Sets and queries bits in the Limit Fail enable register
STAT:QUEStionable:LIMit[:EVENt]?	query only	Reads and clears the Limit Fail event register
STAT:QUEStionable:LIMit:NTRansition	‡	Sets and queries bits in the Limit Fail negative transition register
STAT:QUEStionable:LIMit:PTRansition	‡	Sets queries bits in the Limit Fail positive transition register
STAT:QUEStionable:NTRansition	‡	Sets and queries bits in the Questionable Status negative transition register

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
STAT:QUEStionable:PTRansition	‡	Sets and queries bits in the Questionable Status positive transition register
STAT:QUEStionable:VOLTage:CONDition?	query only	Reads the Questionable Voltage condition register
STAT:QUEStionable:VOLTage:ENABle	‡	Sets and queries bits in the Questionable Voltage enable register
STAT:QUEStionable:VOLTage[:EVENt]?	query only	Reads and clears the Questionable Voltage event register
STAT:QUEStionable:VOLTage:NTRansition	‡	Sets and queries bits in the Questionable Voltage negative transition register
STAT:QUEStionable:VOLTage:PTRansition	‡	Sets bits in the Questionable Voltage positive transition register
STAT:USER:ENABle	‡	Sets and queries bits in the User Status enable register
STAT:USER[:EVENt]?	query only	Reads and clears the User Status event register
STAT:USER:PULSe	command only	Sets bits in the User Status event register
SYSTem		
SYSTem:BEEPer[:IMMediate]	command only	Sets the frequency and duration for the analyzer's beeper
SYST:BEEPer:STATe	‡	Enables the analyzer's beeper
SYST:COMMunicate:GPIB:ADDRess	‡	Sets the analyzer's HP-IB address
SYST:DATE	‡	Sets the date in the analyzer's battery-backed clock
SYST:ERRor?	query only	Returns one error message from the analyzer's error queue
SYST:FLOG:CLEar	command only	Clears the fault log of all entries
SYST:KEY	‡	Writes or queries front-panel key presses
SYST:KLOCk	‡	Disables the keyboard
SYST:PRESet	command only	Returns most of the analyzer's parameters to their preset states
SYST:SET	‡	Transfers an instrument state between the analyzer and an external controller
SYST:TIME	‡	Sets the time in the analyzer's battery-backed clock

‡ Command form.; add "?" for query form.

Subsystem Commands	Form	Description
TEST		
TEST:LOG:CLEar	command only	Clears the test log
TEST:LONG	command only	Executes the long confidence test
TEST:LONG:RESult?	query only	Returns the overall result of the long confidence test
TRACe		
TRACe[:DATA]	‡	Stores data to the specified data register
TRACe:WATerfall[:DATA]	‡	Stores data to the specified waterfall register
TRACe:X[:DATA]?	query only	Returns the X-axis data for trace display
TRACe:X:UNIT?	query only	Returns the unit for the X-axis for trace display
TRACe:Z[:DATA]?	query only	Returns the Z-axis data for waterfall display
TRACe:Z:UNIT?	query only	Returns the unit for the Z-axis in waterfall display
TRIGger		
TRIGger[:IMMediate]	command only	Triggers the analyzer if TRIG:SOUR is BUS
TRIG:LEVel	‡	Specifies the level of the input signal which causes the analyzer to trigger
TRIG:LEVel:TTL	‡	Specifies the TTL level of the signal which triggers the analyzer in order analysis and octave analysis instrument mode
TRIG:SLOPe	‡	Specifies the slope of the signal which triggers the analyzer
TRIG:SOURce	‡	Selects the source of the trigger event
TRIG:STARt[1|2]	‡	Specifies a pre-trigger value or a post-trigger value for the specified channel
TRIG:TACHometer:HOLDoff	‡	Specifies a time interval in which the tachometer trigger is inhibited
TRIG:TACHometer:LEVel	‡	Specifies the level of the tachometer's input signal which causes the analyzer to trigger
TRIG:TACHometer:PCOunt	‡	Specifies the number of tachometer pulses that occur in one revolution of the shaft
TRIG:TACHometer:RANGe	‡	Specifies the input range of the analyzer's tachometer
TRIG:TACHometer[:RPM]?	query only	Reads and returns the current RPM value for the analyzer's tachometer
TRIG:TACHometer:SLOPe	‡	Specifies the slope of the tachometer input signal to be used in RPM step arming or order tracking

‡ Command form.; add "?" for query form.

Cross-Reference from Front-Panel Keys to HP-IB Commands

Introduction

This section lists analyzer hardkeys and softkeys and their equivalent HP-IB commands.

Softkeys are indented to indicate their position in the menu tree. Keys that do not have an equivalent HP-IB command are excluded from the list. Keys which appear in multiple menus may appear only once. For example, time capture is available in most instrument modes and appears in multiple Measurement Data hardkey menus. The HP-IB command for time capture is only listed once.

Multiple HP-IB commands may be required for a single softkey. In these cases, the entire command string is listed. The command string includes valid parameter values.

You can also determine the equivalent HP-IB command for any key sequence by turning on the GPIB echo facility. With GPIB echo on, the analyzer displays the equivalent HP-IB command for each front panel key sequence. To turn on GPIB echo, press the following keys:

[Local/HP-IB]
    [GPIB ECHO **ON** OFF]

Measurement Group

Front Panel Key	HP-IB Command
Inst Mode	
[FFT ANALYSIS]	INSTrument:SELect FFT
[OCTAVE ANALYSIS]	INSTrument:SELect OCTave
[ORDER ANALYSIS]	INSTrument:SELect ORDer
[SWEPT SINE]	INSTrument:SELect SINE
[CORRELATN ANALYSIS]	INSTrument:SELect CORRelation
[HISTOGRAM / TIME]	INSTrument:SELect HISTogram
[CAPTURE ON OFF]	[SENSe:]FEED INPut | TCAPture
[1 CHANNEL]	INPut2[:STATe] OFF
[2 CHANNEL]	INPut2[:STATe] ON
[CAPTURE SETUP]	
[START CAPTURE]	[SENSe:]TCAPture[:IMMediate]
[ABORT CAPTURE]	[SENSe:]TCAPture:ABORt
[CAPTURE LENGTH]	[SENSe:]TCAPture:LENGth (0~9.9e+37[S | REC | PNT])
[ALLOCATE CAPTURE]	
[CONFIRM ALLOCATE]	[SENSe:]TCAPture:MALLocate
[REMOVE CAPTURE]	
[CONFIRM REMOVE]	[SENSe:]TCAPture:DELete
[TACHOMETR SETUP]	
[TACH DATA ON OFF]	[SENSe:]TCAPture:TACHometer[:STATe] OFF | 0 | ON | 1
[MAX RPM]	[SENSe:]TCAPture:TACHometer:RPM:MAXimum (0~9.9e+37)
[ANALYSIS REGION]	
[STRT TIME CHANNEL 1]	[SENSe:]TCAPture:STARt1 (0~9.9e+37[S | REC | PNT])
[STOP TIME CHANNEL 1]	[SENSe:]TCAPture:STOP1 (0~9.9e+37[S | REC | PNT])
[STRT TIME CHANNEL 2]	[SENSe:]TCAPture:STARt2 (0~9.9e+37[S | REC | PNT])
[STOP TIME CHANNEL 2]	[SENSe:]TCAPture:STOP2 (0~9.9e+37[S | REC | PNT])

Front Panel Key	HP-IB Command
Freq	
[SPAN]	[SENSe:]FREQuency:SPAN (0.015625 ~ 102400[HZ])
[CENTER]	[SENSe:]FREQuency:CENTer (0.0234375 ~ 115000[HZ])
[START]	[SENSe:]FREQuency:STARt (0 ~ 115000[HZ])
[STOP]	[SENSe:]FREQuency:STOP (0.03125 ~ 115000[HZ])
[ZERO START]	[SENSe:]FREQuency:STARt 0
[FULL SPAN]	[SENSe:]FREQuency:SPAN:FULL
[ENTRY STEP SIZE]	[SENSe:]FREQuency:STEP[:INCRement] (0.015625 ~ 10240[HZ])
[RECORD LENGTH]	[SENSe:]SWEep:TIME (0.00390625 ~ 4096[S])
[FULL OCTAVE]	[SENSe:]FREQuency:RESolution:OCTave FULL
[1/3 OCTAVE]	[SENSe:]FREQuency:RESolution:OCTave THIRd
[1/12 OCTAVE]	[SENSe:]FREQuency:RESolution:OCTave TWELth
[MIN RPM]	[SENSe:]ORDer:RPM:MINimum (0 ~ 9.9e+37)
[MAX RPM]	[SENSe:]ORDer:RPM:MAXimum (0 ~ 9.9e+37)
[MAX ORDER]	[SENSe:]ORDer:MAXimum (0 ~ 9.9e+37[ORD])
[DELTA ORDER]	[SENSe:]ORDer:RESolution (0 ~ 9.9e+37[ORD])
[TRACK ON OFF]	[SENSe:]ORDer:TRACk:STATe OFF | 0 | ON | 1
[TRACK 1 ORDER]	[SENSe:]ORDer:TRACk1 (0 ~ 9.9e+37[ORD])
[TRACK 2 ORDER]	[SENSe:]ORDer:TRACk2 (0 ~ 9.9e+37[ORD])
[TRACK 3 ORDER]	[SENSe:]ORDer:TRACk3 (0 ~ 9.9e+37[ORD])
[TRACK 4 ORDER]	[SENSe:]ORDer:TRACk4 (0 ~ 9.9e+37[ORD])
[TRACK 5 ORDER]	[SENSe:]ORDer:TRACk5 (0 ~ 9.9e+37[ORD])
[SWEEP LIN LOG]	[SENSe:]SWEep:SPACing LINear | LOGarithmic
[SWEEP UP DOWN]	[SENSe:]SWEep:DIRection UP | DOWN
[SWEEP AUTO MAN]	[SENSe:]SWEep:MODE AUTO | MANual
[MANUAL FREQ]	[SENSe:]FREQuency:MANual (0.015625 ~ 51200[HZ])
[RESOLUTN SETUP]	
[RESOLUTN]	[SENSe:]FREQuency:RESolution (0.015625 ~ 51200 [HZ | PCT | PNT/SWP | PNT/DEC | PNT/OCT])
[AUTO RES ON OFF]	[SENSe:]FREQuency:RESolution:AUTO OFF | 0 | ON | 1
[MAXIMUM % CHANGE]	[SENSe:]FREQuency:RESolution:AUTO:MCHange (0.391 ~ 100)
[MINIMUM RESOLUTN]	[SENSe:]FREQuency:RESolution:AUTO:MINimum (0.015625 ~ 51200 [HZ | PCT | PNT/SWP | PNT/DEC | PNT/OCT])
[RECORD TIME]	[SENSe:]SWEep:TIME (0.00390625 ~ 4096[S])
[HISTOGRAM LENGTH]	[SENSe:]SWEep:DWELl (0 ~ 9.9e+37[S | REC | PNT])
[HISTOGRAM BINS]	[SENSe:]HISTogram:BINS (4 ~ 2048)
[RECORD LENGTH]	[SENSe:]SWEep:TIME (0.00390625 ~ 4096[S])

Front Panel Key	HP-IB Command
Window	
[HANNING]	[SENSe:]WINDow[:TYPE] HANNing
[FLAT TOP]	[SENSe:]WINDow[:TYPE] FLATtop
[UNIFORM]	[SENSe:]WINDow[:TYPE] UNIForm
[FORCE EXPO]	[SENSe:]WINDow1[:TYPE] FORCe;WINDow2[:TYPE] EXPO [SENSe:]WINDow1[:TYPE] EXPO;WINDow2[:TYPE] FORCe
[CHANNEL 1 FORC EXPO]	[SENSe:]WINDow1[:TYPE] FORCe|EXPO
[CHANNEL 2 FORC EXPO]	[SENSe:]WINDow2[:TYPE] FORCe|EXPO
[FORCE WIDTH]	[SENSe:]WINDow:FORCe (3.8147e-06 ~ 9999900[S])
[EXPO DECAY]	[SENSe:]WINDow:EXPonential (3.8147e-06 ~ 9999900[S])
[CP DC BIN ON OFF]	[SENSe:]WINDow:ORDer:DC OFF|0|ON|1
[ZERO PAD -T/4, T/4]	[SENSe:]WINDow[:TYPE] LLAG
[ZERO PAD 0, T/2]	[SENSe:]WINDow[:TYPE] LAG
Input	
[CHANNEL 1 RANGE]	[SENSe:]VOLTage1:RANGe[:UPPer] (-51 ~ 31.66 [DBVRMS|VPK|DBVPK|V|DBV|EU|DBEU|VRMS])
[CH1 FIXED RANGE]	[SENSe:]VOLTage1:RANGe:AUTO OFF
[CH1 AUTO RANGE]	[SENSe:]VOLTage1:RANGe:AUTO ON
[CHANNEL 1 SETUP]	
[INPUT LOW FLOAT GND]	INPut#:LOW GROund|FLOat
[COUPLING AC DC]	INPut#:COUPling AC|DC
[ANTIALIAS ON OFF]	INPut#:FILTer[:LPASs][:STATe] OFF|0|ON|1
[A WT FLTR ON OFF]	INPut#:FILTer:AWEighting[:STATe] OFF|0|ON|1
[ICP SUPLY ON OFF]	INPut#:BIAS[:STATe] OFF|0|ON|1
[ENG UNIT ON OFF]	[SENSe:]VOLTage#:RANGe:UNIT:USER [:STATe] OFF|0|ON|1
[ENG UNIT MULTIPLER]	[SENSe:]VOLTage#:RANGe:UNIT:USER:SFACtor (-9.9e+37 ~ 9.9e+37[V/EU|EU/V])
[ENG UNIT AT MKR]	[SENSe:]VOLTage#:RANGe:UNIT:USER:SFACtor (-9.9e+37 ~ 9.9e+37)
[ENG UNIT LABEL]	[SENSe:]VOLTage#:RANGe:UNIT:USER:LABel <STRING>
[CHANNEL 2 RANGE]	[SENSe:]VOLTage2:RANGe[:UPPer] (-51 ~ 31.66 [DBVRMS|VPK|DBVPK|V|DBV|EU|DBEU|VRMS])
[CH2 FIXED RANGE]	[SENSe:]VOLTage2:RANGe:AUTO OFF
[CH2 AUTO RANGE]	[SENSe:]VOLTage2:RANGe:AUTO ON
[CHANNEL 2 SETUP]	

Front Panel Key	HP-IB Command
[TACHOMETR SETUP]	
[TACH PULS PER REV]	TRIGger:TACHometer:PCOunt (0.5 ~ 2048)
[TRG RANGE +/- 20 4]	TRIGger:TACHometer:RANGe HIGH | LOW
[LEVEL]	TRIGger:TACHometer:LEVel (-20 ~ 20[V])
[HOLDOFF TIME]	TRIGger:TACHometer:HOLDoff (0 ~ 0.052224[S])
[SLOPE POS NEG]	TRIGger:TACHometer:SLOPe POSitive | NEGative
Source	
[SOURCE ON OFF]	OUTPut[:STATe] OFF | 0 | ON | 1
[LEVEL]	SOURce:VOLTage[:LEVel][:IMMediate][:AMPLitude] (-9.9e+37 ~ 13.9794[DBVRMS | VPK | DBVPK | V | DBV | VRMS])
[RANDOM NOISE]	SOURce:FUNCtion[:SHAPe] RANDom
[BURST RANDOM]	SOURce:BURSt (0 ~ 100[PCT])
[PERIODIC CHIRP]	SOURce:FUNCtion[:SHAPe] PCHirp
[BURST CHIRP]	SOURce:CHIRp (0 ~ 100[PCT])
[PINK NOISE]	SOURce:FUNCtion[:SHAPe] PINK
[FIXED SINE]	SOURce:FREQuency[:CW] (0 ~ 115000[HZ])
[ARBITRARY (D1-D8)]	SOURce:FUNCtion[:SHAPe] USER [D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8]
[ARB SRC SETUP]	
[REPEAT ON OFF]	SOURce:USER:REPeat OFF | 0 | ON | 1
[DATA REG D1]	SOURce:USER[:REGister] D1
[DATA REG D2]	SOURce:USER[:REGister] D2
[DATA REG D3]	SOURce:USER[:REGister] D3
[DATA REG D4]	SOURce:USER[:REGister] D4
[DATA REG D5]	SOURce:USER[:REGister] D5
[DATA REG D6]	SOURce:USER[:REGister] D6
[DATA REG D7]	SOURce:USER[:REGister] D7
[DATA REG D8]	SOURce:USER[:REGister] D8
[RAMP RATE]	SOURce:VOLTage:SLEW (0 ~ 7071 [V/S | VPK/S | VRMS/S])
[AUTOLEVEL ON OFF]	SOURce:VOLTage[:LEVel]:AUTO OFF | 0 | ON | 1
[AUTOLEVEL SETUP]	
[REF CHAN CH1 CH2]	SOURce:VOLTage[:LEVel]:REFerence:CHANnel INPut1 | INPut2
[REFERENCE LEVEL]	SOURce:VOLTage[:LEVel]:REFerence (-69.276 ~ 31.66 [DBVRMS | VPK | DBVPK | V | DBV | EU | DBEU | VRMS])
[REFERENCE TOLERANCE]	SOURce:VOLTage[:LEVel]:REFerence:TOLerance (0.1 ~ 20[DB])
[MAX SRC LEVEL]	SOURce:VOLTage:LIMit[:AMPLitude] (-9.9e+37 ~ 13.9794 [DBVRMS | VPK | DBVPK | V | DBV | VRMS])
[MAX INPUT LEVEL]	SOURce:VOLTage:LIMit:INPut (-69.276 ~ 31.66 [DBVRMS | VPK | DBVPK | V | DBV | EU | DBEU | VRMS])

Front Panel Key	HP-IB Command
Trigger	
[FREE RUN TRIGGER]	TRIGger:SOURce IMMediate
[EXTERNAL TRIGGER]	TRIGger:SOURce EXTernal
[CHANNEL 1 TRIGGER]	TRIGger:SOURce INTernal1
[CHANNEL 2 TRIGGER]	TRIGger:SOURce INTernal2
[SOURCE TRIGGER]	TRIGger:SOURce OUTPut
[HP-IB TRIGGER]	TRIGger:SOURce BUS
[TACHOMETR SETUP]	
[TACH PULS PER REV]	TRIGger:TACHometer:PCOunt (0.5 ~ 2048)
[TRG RANGE +/- 20 4]	TRIGger:TACHometer:RANGe HIGH | LOW
[LEVEL]	TRIGger:TACHometer:LEVel (-20 ~ 20[V])
[HOLDOFF TIME]	TRIGger:TACHometer:HOLDoff (0 ~ 0.052224[S])
[SLOPE POS NEG]	TRIGger:TACHometer:SLOPe POSitive | NEGative
[TRIGGER SETUP]	
[LEVEL]	TRIGger:LEVel (-100 ~ 100[PCT])
[SLOPE POS NEG]	TRIGger:SLOPe POSitive | NEGative
[CHANNEL 1 DELAY]	TRIGger:STARt1 (-9.9e+37 ~ 9.9e+37[S])
[CHANNEL 2 DELAY]	TRIGger:STARt2 (-9.9e+37 ~ 9.9e+37[S])
[ARM SETUP]	
[AUTOMATIC ARM]	ARM:SOURce IMMediate
[MANUAL ARM]	ARM:SOURce MANual
[RPM STEP ARM]	ARM:SOURce RPM
[TIME STEP ARM]	ARM:SOURce TIMer
[START RPM USAGE]	
[START RPM OFF]	ARM:RPM:MODE OFF
[RPM INCREASNG]	ARM:RPM:MODE UP
[RPM DECREASNG]	ARM:RPM:MODE DOWN
[START RPM]	ARM:RPM:THReshold (5 ~ 491520)
[RPM STEP SIZE]	ARM:RPM:INCRement (1 ~ 500000)
[TIME STEP SIZE]	ARM:TIMer (0 ~ 500000[S])
[WATERFALL STEPS]	CALCulate:WATerfall:COUNt (1 ~ 32760)
[ARM]	ARM[:IMMediate]
[FREE RUN TRIGGER]	TRIGger:SOURce IMMediate
[EXTERNAL TRIGGER]	TRIGger:SOURce EXTernal
[HP-IB TRIGGER]	TRIGger:SOURce BUS
[SLOPE POS NEG]	TRIGger:SLOPe POSitive | NEGative
[AUTOMATIC ARM]	ARM:SOURce IMMediate
[RPM STEP ARM]	ARM:SOURce RPM
[TIME STEP ARM]	ARM:SOURce TIMer
[DELAY TIME]	TRIGger:STARt (0 ~ 32768[S])

Front Panel Key	HP-IB Command
Start	
START Hardkey	ABORt;:INITiate[:IMMediate]
Pause Cont	
PAUSE/CONT Hardkey	INITiate:CONTinuous OFF | 0 | ON | 1
Avg	
[AVERAGE ON OFF]	[SENSe:]AVERage[:STATe] OFF | 0 | ON | 1
[NUMBER AVERAGES]	[SENSe:]AVERage:COUNt (1 ~ 99999)
[AVERAGE TYPE]	
[RMS]	[SENSe:]AVERage:TYPE RMS
[RMS EXPONENTL]	[SENSe:]AVERage:TYPE RMS;TCONtrol EXPonential
[VECTOR]	[SENSe:]AVERage:TYPE VECTor
[VECTOR EXPONENTL]	[SENSe:]AVERage:TYPE VECTor;TCONtrol EXPonential
[PEAK HOLD]	[SENSe:]AVERage:TYPE MAXimum
[FAST AVG ON OFF]	[SENSe:]AVERage:IRESult[:STATe] OFF | 0 | ON | 1
[UPDATE RATE]	[SENSe:]AVERage:IRESult:RATE (1 ~ 99999)
[REPEAT ON OFF]	[SENSe:]AVERage:TCONtrol FREeze | REPeat | EXPonential
[OVERLAP PERCENT]	[SENSe:]SWEep:OVERlap (0 ~ 99[PCT])
[OVLD REJ ON OFF]	[SENSe:]REJect:STATe OFF | 0 | ON | 1
[PREVIEW SETUP]	
[PREVIEW OFF]	[SENSe:]AVERage:PREView OFF
[MANUAL PREVIEW]	[SENSe:]AVERage:PREView MANual
[TIMED PREVIEW]	[SENSe:]AVERage:PREView TIMed
[TIME]	[SENSe:]AVERage:PREView:TIME (0.1 ~ 3600[S])
[REJECT TIME REC]	[SENSe:]AVERage:PREView:REJect
[ACCEPT TIME REC]	[SENSe:]AVERage:PREView:ACCept
[LINEAR]	[SENSe:]AVERage:TYPE RMS
[EXPONENTL]	[SENSe:]AVERage:TYPE RMS;TCONtrol EXPonential
[EQUAL CONFID]	[SENSe:]AVERage:TYPE ECONfidence
[PEAK HOLD]	[SENSe:]AVERage:TYPE MAX
[HOLD SETUP]	
[OFF]	[SENSe:]AVERage:HOLD OFF
[MAXIMUM]	[SENSe:]AVERage:HOLD MAXimum
[MINIMUM]	[SENSe:]AVERage:HOLD MINimum

Front Panel Key	HP-IB Command
[AVERAGE TIME]	[SENSe:]AVERage:TIME (0.125 ~ 8192[S])
[CONFIDNCE LEVEL]	[SENSe:]AVERage:CONFidence (0 ~ 100[DB])
[IMPULSE ON OFF]	[SENSe:]AVERage:IMPulse OFF|0|ON|1
[REPEAT ON OFF]	[SENSe:]AVERage:TCONtrol FREeze|REPeat|EXPonential
[TIME]	[SENSe:]AVERage:TYPE VECTor
[TIME EXPONENTL]	[SENSe:]AVERage:TYPE VECTor;TCONtrol EXPonential
[REPEAT ON OFF]	[SENSe:]AVERage:TCONtrol FREeze|REPeat|EXPonential
[SETTLE TIME]	[SENSe:]SWEep:STIMe (0 ~ 9.9e+37[S|CYCLE])
[INTEGRATE TIME]	[SENSe:]SWEep:DWELl (0 ~ 9.9e+37[S|CYCLE])
[FAST AVG ON OFF]	[SENSe:]AVERage:IRESult[:STATe] ON|OFF
[REPEAT ON OFF]	[SENSe:]AVERage:TCONtrol FREeze|REPeat|EXPonential

Display Group

Front Panel Key	HP-IB Command
Meas Data	
[PWR SPEC CHANNEL 1]	CALCulate#:FEED 'XFR:POW 1';MATH:STATe OFF;*WAI
[PWR SPEC CHANNEL 2]	CALCulate#:FEED 'XFR:POW 2';MATH:STATe OFF;*WAI
[LIN SPEC CHANNEL 1]	CALCulate#:FEED 'XFR:POW:LIN 1';MATH:STATe OFF;*WAI
[LIN SPEC CHANNEL 2]	CALCulate#:FEED 'XFR:POW:LIN 2';MATH:STATe OFF;*WAI
[TIME CHANNEL 1]	CALCulate#:FEED 'XTIM:VOLT 1';MATH:STATe OFF;*WAI
[TIME CHANNEL 2]	CALCulate#:FEED 'XTIM:VOLT 2';MATH:STATe OFF;*WAI
[FREQUENCY RESPONSE]	CALCulate#:FEED 'XFR:POW:RATio 2,1';MATH:STATe OFF;*WAI
[COHERENCE]	CALCulate#:FEED 'XFR:POW:COH 1,2';MATH:STATe OFF;*WAI
[CROSS SPECTRUM]	CALCulate):FEED 'XFR:POW:CROS 1,2';MATH:STATe OFF;*WAI
[MORE]	
[ORBIT]	CALCulate):FEED 'XVOL:VOLT 1,2';MATH:STATe OFF;*WAI
[WINDOWED TIME CH1]	CALCulate#:FEED 'XTIM:VOLT:WINDow 1';MATH:STATe OFF;*WAI
[WINDOWED TIME CH2]	CALCulate#:FEED 'XTIM:VOLT:WINDow 2';MATH:STATe OFF;*WAI
[CAPTURE CHANNEL 1]	CALCulate#:FEED 'TCAP 1';MATH:STATe OFF;*WAI
[CAPTURE CHANNEL 2]	CALCulate#:FEED 'TCAP 2';MATH:STATe OFF;*WAI
[MATH FUNCTION]	
[F1]	CALCulate#:MATH:SELect F1;STATe ON
[F2]	CALCulate#:MATH:SELect F2;STATe ON
[F3]	CALCulate#:MATH:SELect F3;STATe ON
[F4]	CALCulate#:MATH:SELect F4;STATe ON
[F5]	CALCulate#:MATH:SELect F5;STATe ON
[DATA REGISTER]	
[D1]	CALCulate#:FEED 'D1';MATH:STATe OFF;*WAI
[D2]	CALCulate#:FEED 'D2';MATH:STATe OFF;*WAI
[D3]	CALCulate#:FEED 'D3';MATH:STATe OFF;*WAI
[D4]	CALCulate#:FEED 'D4';MATH:STATe OFF;*WAI
[D5]	CALCulate#:FEED 'D5';MATH:STATe OFF;*WAI
[D6]	CALCulate#:FEED 'D6';MATH:STATe OFF;*WAI
[D7]	CALCulate#:FEED 'D7';MATH:STATe OFF;*WAI
[D8]	CALCulate#:FEED 'D8';MATH:STATe OFF;*WAI

Front Panel Key	HP-IB Command
[WATERFALL REGISTER]	
[W1]	CALCulate#:FEED 'W1';MATH:STATe OFF;*WAI
[W2]	CALCulate#:FEED 'W2';MATH:STATe OFF;*WAI
[W3]	CALCulate#:FEED 'W3';MATH:STATe OFF;*WAI
[W4]	CALCulate#:FEED 'W4';MATH:STATe OFF;*WAI
[W5]	CALCulate#:FEED 'W5';MATH:STATe OFF;*WAI
[W6]	CALCulate#:FEED 'W6';MATH:STATe OFF;*WAI
[W7]	CALCulate#:FEED 'W7';MATH:STATe OFF;*WAI
[W8]	CALCulate#:FEED 'W8' ;MATH:STATe OFF;*WAI
[MORE]	
[COMP PWR CHANNEL 1]	CALCulate#:FEED 'XFR:POW:COMP 1';MATH:STATe OFF;*WAI
[COMP PWR CHANNEL 2]	CALCulate#:FEED 'XFR:POW:COMP 2';MATH:STATe OFF;*WAI
[ORDER TRK CHANNEL 1]	
[TRACK 1]	CALCulate#:FEED 'XORDer:TRACk 1,1';MATH:STATe OFF;*WAI
[TRACK 2]	CALCulate#:FEED 'XORDer:TRACk 1,2';MATH:STATe OFF;*WAI
[TRACK 3]	CALCulate#:FEED 'XORDer:TRACk 1,3';MATH:STATe OFF;*WAI
[TRACK 4]	CALCulate#:FEED 'XORDer:TRACk 1,4';MATH:STATe OFF;*WAI
[TRACK 5]	CALCulate#:FEED 'XORDer:TRACk 1,5';MATH:STATe OFF;*WAI
[ORDER TRK CHANNEL 2]	
[TRACK 1]	CALCulate#:FEED 'XORDer:TRACk 2,1';MATH:STATe OFF;*WAI
[TRACK 2]	CALCulate#:FEED 'XORDer:TRACk 2,2';MATH:STATe OFF;*WAI
[TRACK 3]	CALCulate#:FEED 'XORDer:TRACk 2,3';MATH:STATe OFF;*WAI
[TRACK 4]	CALCulate#:FEED 'XORDer:TRACk 2,4';MATH:STATe OFF;*WAI
[TRACK 5]	CALCulate#:FEED 'XORDer:TRACk 2,5';MATH:STATe OFF;*WAI
[RPM PROFILE]	CALCulate#:FEED 'XRPM:PROF';MATH:STATe OFF;*WAI
[NORM VAR CHANNEL 1]	CALCulate#:FEED 'XFR:POW:VAR 1';MATH:STATe OFF;*WAI
[NORM VAR CHANNEL 2]	CALCulate#:FEED 'XFR:POW:VAR 2';MATH:STATe OFF;*WAI
[AUTO CORR CHANNEL 1]	CALCulate#:FEED 'XTIM:CORR 1';MATH:STATe OFF;*WAI
[AUTO CORR CHANNEL 2]	CALCulate#:FEED 'XTIM:CORR 2';MATH:STATe OFF;*WAI
[CROSS CORRELATN]	CALCulate#:FEED 'XTIM:CORR:CROS 1,2';MATH:STATe OFF;*WAI
[HISTOGRAM CHANNEL 1]	CALCulate#:FEED 'XTIM:VOLT:HIST 1';MATH:STATe OFF;*WAI
[HISTOGRAM CHANNEL 2]	CALCulate#:FEED 'XTIM:VOLT:HIST 2';MATH:STATe OFF;*WAI
[PDF CHANNEL 1]	CALCulate#:FEED 'XTIM:VOLT:PDF 1';MATH:STATe OFF;*WAI
[PDF CHANNEL 2]	CALCulate#:FEED 'XTIM:VOLT:PDF 2';MATH:STATe OFF;*WAI
[CDF CHANNEL 1]	CALCulate#:FEED 'XTIM:VOLT:CDF 1';MATH:STATe OFF;*WAI
[CDF CHANNEL 2]	CALCulate#:FEED 'XTIM:VOLT:CDF 2';MATH:STATe OFF;*WAI
[UNFILTERD TIME CH 1]	CALCulate#:FEED 'XTIM:VOLT 1';MATH:STATe OFF;*WAI
[UNFILTERD TIME CH 2]	CALCulate#:FEED 'XTIM:VOLT 2'

Front Panel Key	HP-IB Command
Trace Coords	
[LINEAR MAGNITUDE]	CALCulate#:FORMat MLINear
[LOG MAGNITUDE]	CALCulate#:FORMat MLINear;:DISPlay:WINDow#:TRACe:Y:SPACing LOGarithmic
[dB MAGNITUDE]	CALCulate#:FORMat MLOGarithmic
[PHASE]	CALCulate#:FORMat PHASe
[UNWRAPPED PHASE]	CALCulate#:FORMat UPHase
[REAL PART]	CALCulate#:FORMat REAL
[IMAGINARY PART]	CALCulate#:FORMat IMAGinary
[NYQUIST DIAGRAM]	CALCulate#:FORMat NYQuist
[AMPLITUDE PEAK RMS]	CALCulate#:UNIT:AMPLitude PEAK | RMS
[PHASE DEG RAD]	CALCulate#:UNIT:ANGLe DEGRee | RADian
[VOLTS]	CALCulate#:UNIT:VOLT 'V'
[VOLTS ^ 2]	CALCulate#:UNIT:VOLT 'V2'
[V/rtHz]	CALCulate#:UNIT:VOLT 'V/RTHZ'
[V ^ 2/Hz (PSD)]	CALCulate#:UNIT:VOLT 'V2/HZ'
[V ^ 2s/Hz (ESD)]	CALCulate#:UNIT:VOLT 'V2S/HZ'
[X-AXIS LIN LOG]	DISPlay:WINDow#:TRACe:X:SPACing LINear | LOGarithmic

Front Panel Key	HP-IB Command
Scale	
[AUTOSCALE ON OFF]	DISPlay:WINDow#:TRACe:Y[:SCALe]AUTO OFF|0|ON|1|ONCE
[TOP REFERENCE]	DISPlay:WINDow#:TRACe:Y[:SCALe]:REFerence:TOP (-9.9e+37 ~ 9.9e+37[DBVRMS|VRMS|VPK|DBVPK|V|DBV|EU|DBEU])
[CENTER REFERENCE]	DISPlay:WINDow#:TRACe:Y[:SCALe]:REFerence:CENTer (-9.9e+37 ~ 9.9e+37[DBVRMS|VRSM|VPK|DBVPK|V|DBV|EU|DBEU])
[BOTTOM REFERENCE]	DISPlay:WINDow#:TRACe:Y[:SCALe]:REFerence:BOTTom (-9.9e+37 ~ 9.9e+37[DBVRMS|VRMS|VPK|DBVPK|V|DBV|EU|DBEU])
[INP RANGE TRACKING]	DISPlay:WINDow#:TRACe:Y[:SCALe]:REFerence RANGe
[Y PER DIV (DECADES)]	DISPlay:WINDow#:TRACe:Y[:SCALe]PDIV (1e-06 ~ 9.9e+37[DB|VRMS|VPK|V])
[MATCH X SCALE]	DISPlay:WINDow#:TRACe:X:MATCh)
[MATCH Y SCALE]	DISPlay:WINDow#:TRACe:Y:MATCh)
[FULL SCALE]	DISPlay:WINDow#:TRACe:X[:SCALe]:AUTO ONCE DISPlay:WINDow):TRACe:Y[:SCALe]:AUTO ONCE
Active Trace	
ACTIVE TRACE Hardkey	CALCulate):ACTive [OFF|0|ON|1]
Analys	
[DEFINE FUNCTION]	
[DEFINE F1]	CALCulate#:MATH[:EXPRession1
[MEAS DATA]	
[PWR SPEC CHANNEL 1]	PSPEC1
[+]	+
[-]	-
[*]	*
[/]	/
[ )]	)
[PWR SPEC CHANNEL 2]	PSPEC2
[LIN SPEC CHANNEL 1]	LSPEC1
[LIN SPEC CHANNEL 2]	LSPEC2
[FREQUENCY RESPONSE]	FRES
[COHERENCE]	COH
[CROSS SPECTRUM]	CSPEC

Front Panel Key	HP-IB Command
[TIME CHANNEL 1]	TIME1
[TIME CHANNEL 2]	TIME2
[WNDO TIME CHANNEL 1]	WTIME1
[WNDO TIME CHANNEL 2]	WTIME2
[RESAMPLED TIME CH 1]	TIME1
[RESAMPLED TIME CH 2]	TIME2
[COMP PWR CHANNEL 1]	CPOW1
[COMP PWR CHANNEL 2]	CPOW1
[ORDER TRK CHANNEL 1]	TRACK1
[ORDER TRK CHANNEL 2]	TRACK2
[RPM PROFILE]	RPM
[AUTO CORR CHANNEL 1]	ACORR1
[AUTO CORR CHANNEL 2]	ACORR2
[CROSS CORRELATN]	XCORR
[HISTOGRAM CHANNEL 1]	HIST1
[HISTOGRAM CHANNEL 2]	HIST2
[PDF CHANNEL 1]	PDF1
[PDF CHANNEL 2]	PDF2
[CDF CHANNEL 1]	CDF1
[CDF CHANNEL 2]	CDF2
[DATA REG (D1-D8)]	
[DATA REG D1]	D1
[DATA REG D2]	D2
[DATA REG D3]	D3
[DATA REG D4]	D4
[DATA REG D5]	D5
[DATA REG D6]	D6
[DATA REG D7]	D7
[DATA REG D8]	D8
[CONSTANT (K1-K5)]	
[CONSTANT K1]	K1
[CONSTANT K2]	K2
[CONSTANT K3]	K3
[CONSTANT K4]	K4
[CONSTANT K5]	K5
[FUNCTION (F1-F5)]	
[FUNCTION F1]	F1
[FUNCTION F2]	F2
[FUNCTION F3]	F3
[FUNCTION F4]	F4
[FUNCTION F5]	F5

<table>
<tr><th>Front Panel Key</th><th>HP-IB Command</th></tr>
<tr><td>[OPERATION]</td><td></td></tr>
<tr><td>[CONJ( ]</td><td>CONJ(</td></tr>
<tr><td>[MAG( ]</td><td>MAG(</td></tr>
<tr><td>[REAL( ]</td><td>REAL(</td></tr>
<tr><td>[IMAG( ]</td><td>IMAG(</td></tr>
<tr><td>[SQRT( ]</td><td>SQRT(</td></tr>
<tr><td>[FFT( ]</td><td>FFT(</td></tr>
<tr><td>[INVERSE FFT( ]</td><td>IFFT(</td></tr>
<tr><td>[PSD( ]</td><td>PSD(</td></tr>
<tr><td>[MORE]</td><td></td></tr>
<tr><td>[LN( ]</td><td>LN(</td></tr>
<tr><td>[EXP( ]</td><td>EXP(</td></tr>
<tr><td>[*jOMEGA( ]</td><td>XJOM(</td></tr>
<tr><td>[/jOMEGA( ]</td><td>DJOM(</td></tr>
<tr><td>[AWEIGHT( ]</td><td>AWEIGHT(</td></tr>
<tr><td>[BWEIGHT( ]</td><td>BWEIGHT(</td></tr>
<tr><td>[CWEIGHT( ]</td><td>CWEIGHT(</td></tr>
<tr><td>[DIFF( ]</td><td>DIFF(</td></tr>
<tr><td>[INTEG( ]</td><td>INTEG(</td></tr>
<tr><td>[( ]</td><td>(</td></tr>
<tr><td>[DEFINE F2]</td><td>CALCulate#:MATH:EXPRession2</td></tr>
<tr><td>[DEFINE F3]</td><td>CALCulate#:MATH:EXPRession3</td></tr>
<tr><td>[DEFINE F4]</td><td>CALCulate#:MATH:EXPRession4</td></tr>
<tr><td>[DEFINE F5]</td><td>CALCulate#:MATH:EXPRession5</td></tr>
<tr><td>[DEFINE CONSTANT]</td><td></td></tr>
<tr><td>[DEFINE K1]</td><td>CALCulate:MATH:CONStant1 (-9.9e+37 ~ 9.9e+37)[,(-9.9e+37 ~ 9.9e+37)]</td></tr>
<tr><td>[DEFINE K2]</td><td>CALCulate:MATH:CONStant2 (-9.9e+37 ~ 9.9e+37)[,(-9.9e+37 ~ 9.9e+37)]</td></tr>
<tr><td>[DEFINE K3]</td><td>CALCulate:MATH:CONStant3 (-9.9e+37 ~ 9.9e+37)[,(-9.9e+37 ~ 9.9e+37)]</td></tr>
<tr><td>[DEFINE K4]</td><td>CALCulate:MATH:CONStant4 (-9.9e+37 ~ 9.9e+37)[,(-9.9e+37 ~ 9.9e+37)]</td></tr>
<tr><td>[DEFINE K5]</td><td>CALCulate:MATH:CONStant5 (-9.9e+37 ~ 9.9e+37)[,(-9.9e+37 ~ 9.9e+37)]</td></tr>
<tr><td>[LIMIT TEST]</td><td></td></tr>
<tr><td>[LINES ON OFF]</td><td>DISPlay[:WINDow)]:LIMit:STATe OFF|0|ON|1</td></tr>
<tr><td>[TEST EVAL ON OFF]</td><td>CALCulate#:LIMit:STATe OFF|0|ON|1</td></tr>
<tr><td>[FAIL BEEP ON OFF]</td><td>CALCulate:LIMit:BEEP[:STATe] OFF|0|ON|1</td></tr>
<tr><td>[DEFINE UPPER LIM]</td><td></td></tr>
<tr><td>[START SEGMENT]</td><td></td></tr>
<tr><td>[FINISH SEGMENT]</td><td>CALCulate#:LIMit:UPPer:SEGMENT (-9.9e+37 ~ 9.9e+37)</td></tr>
<tr><td>[MOVE ALL VERTICAL]</td><td>CALCulate#:LIMit:UPPer:MOVE:Y (-9.9e+37 ~ 9.9e+37)</td></tr>
<tr><td>[DELETE SEGMENT]</td><td>CALCulate#:LIMit:UPPer:SEGMENT:CLEar</td></tr>
<tr><td>[DELETE ALL]</td><td></td></tr>
<tr><td>[CONFIRM/ DELETE]</td><td>CALCulate#:LIMit:UPPer:CLEar</td></tr>
<tr><td>[TRACE TO LIMIT]</td><td>CALCulate#:LIMit:UPPer:TRACe[:IMMediate]</td></tr>
</table>

Front Panel Key	HP-IB Command
[DEFINE LOWER LIM]	
[START SEGMENT]	
[FINISH SEGMENT]	CALCulate#:LIMit:LOWer:SEGMENT (-9.9e+37 ~ 9.9e+37)
[MOVE ALL VERTICAL]	CALCulate#:LIMit:LOWer:MOVE:Y (-9.9e+37 ~ 9.9e+37)
[ ]	
[DELETE SEGMENT]	CALCulate#:LIMit:LOWer:SEGMENT:CLEar
[DELETE ALL]	
[CONFIRM/ DELETE]	CALCulate#:LIMit:LOWer:CLEar
[TRACE TO LIMIT]	CALCulate#:LIMit:LOWer:TRACe[:IMMediate]
[CURVE FIT]	
[START FIT]	CALCulate2:CFIT[:IMMediate]
[ABORT FIT]	CALCulate2:CFIT:ABORt
[CURVE FIT REGISTER]	
[D1]	CALCulate#:CFIT:DESTination D1
[D2]	CALCulate:CFIT:DESTination D2
[D3]	CALCulate:CFIT:DESTination D3
[D4]	CALCulate:CFIT:DESTination D4
[D5]	CALCulate:CFIT:DESTination D5
[D6]	CALCulate:CFIT:DESTination D6
[D7]	CALCulate:CFIT:DESTination D7
[D8]	CALCulate:CFIT:DESTination D8
[COPY FROM SYNTHESIS]	CALCulate:CFIT:COPY SYNThesis
[FIT REGION]	
[FULL SPAN]	CALCulate#:CFIT:FREQuency[:AUTO] ON
[USER SPAN]	CALCulate#:CFIT:FREQuency[:AUTO] OFF
[START]	CALCulate#:CFIT:FREQuency:STARt (0 ~ 115000[HZ])
[STOP]	CALCulate#:CFIT:FREQuency:STOP (0.390625 ~ 115000[HZ])
[CURVE FIT SETUP]	
[ORDER MAX FIXED]	CALCulate:CFIT:ORDer:AUTO OFF | 0 | ON | 1
[NUMBER OF POLES]	CALCulate:CFIT:ORDer:POLes (0 ~ 20)
[NUMBER OF ZEROS]	CALCulate:CFIT:ORDer:ZERos (0 ~ 20)
[WEIGHT AUTO USER]	CALCulate:CFIT:WEIGht:AUTO OFF | 0 | ON | 1
[WEIGHT REGISTER]	
[D1]	CALCulate):CFIT:WEIGht:REGister D1
[D2]	CALCulate:CFIT:WEIGht:REGister D2
[D3]	CALCulate:CFIT:WEIGht:REGister D3
[D4]	CALCulate:CFIT:WEIGht:REGister D4
[D5]	CALCulate:CFIT:WEIGht:REGister D5
[D6]	CALCulate:CFIT:WEIGht:REGister D6
[D7]	CALCulate:CFIT:WEIGht:REGister D7
[D8]	CALCulate:CFIT:WEIGht:REGister D8
[TIME DELAY]	CALCulate:CFIT:TDELay (-100 ~ 100[S])
[FREQUENCY SCALE]	CALCulate:CFIT:FSCale (1e-06 ~ 1000000)
[TABLE ON OFF]	DISPlay:CONTents CTAB

Front Panel Key	HP-IB Command
[SYNTHESIS]	
[START SYNTHESIS]	CALCulate:SYNThesis[:IMMediate]
[SYNTHESIS REGISTER]	
[D1]	CALCulate#:SYNThesis:DESTination D1
[D2]	CALCulate:SYNThesis:DESTination D2
[D3]	CALCulate:SYNThesis:DESTination D3
[D4]	CALCulate:SYNThesis:DESTination D4
[D5]	CALCulate:SYNThesis:DESTination D5
[D6]	CALCulate:SYNThesis:DESTination D6
[D7]	CALCulate:SYNThesis:DESTination D7
[D8]	CALCulate:SYNThesis:DESTination D8
[COPY FROM CURVE FIT]	CALCulate:SYNThesis:COPY CFIT
[CONVERT TABLE]	
[CONVRT TO POLE ZERO]	CALCulate):SYNThesis:TTYP# PZERo
[CONVRT TO POLE RESD]	CALCulate:SYNThesis:TTYPe PFRaction
[CONVRT TO POLYNMIAL]	CALCulate:SYNThesis:TTYPe POLYnomial
[SYNTHESIS SETUP]	
[GAIN FACTOR]	CALCulate2:SYNThesis:GAIN (-1000000 ~ 1000000)
[TIME DELAY]	CALCulate2:SYNThesis:TDELay (-100 ~ 100[S])
[FREQUENCY SCALE]	CALCulate2:SYNThesis:FSCale (1e-06 ~ 1000000)
[X-AXIS LIN LOG]	CALCulate2:SYNThesis:SPACing LINear | LOGarithmic
[TABLE ON OFF]	DISPlay:CONTents STAB
Format	
[SINGLE]	DISPlay:FORMat SINGle
[UPPER/ LOWER]	DISPlay:FORMat ULOWer
[FRONT/ BACK]	DISPlay:FORMat FBACk
[WATERFALL]	DISPlay:FORMat WATerfall
[MEASURMNT STATE]	DISPlay:CONTents MSTate
[INPUT STATE]	DISPlay:CONTents lSTate
[WATERFALL ACT TRACE]	CALCulate2:FEED (CALCulate2:FEED?);ACT ON;:DISPlay:FORMat WATerfall;*WAI
[WATERFALL SETUP]	
[MAX TRCES DISPLAYED]	DISPlay:WATerfall:COUNt (1 ~ 50000)
[TRACE HEIGHT]	DISPlay:WATerfall:HEIGht (1 ~ 100[PCT])
[HIDN LINE ON OFF]	DISPlay:WATerfall:HIDDen OFF | 0 | ON | 1
[BASELINE SUPPRESS]	DISPlay:WATerfall:BASeline (0 ~ 100[PCT])
[BODE DIAGRAM]	DISPlay:BODE
[MORE]	
[GRID ON OFF]	DISPlay[:WINDow)]:TRACe:GRATicule:GRID [:STATe] OFF | 0 | ON | 1
[BLANK ANNOTATN]	DISPlay:ANNotation OFF
[BLANK DISPLAY]	DISPlay:ENABle OFF
[TRACE TITLE]	DISPlay:[:WINDow)]TRACe:LABel <STRING>
[DFLT TITL ON OFF]	DISPlay:[:WINDow)]TRACe:LABel:DEFault[:STATe] OFF | 0 | ON | 1

Marker Group

Front Panel Key	HP-IB Command
Marker	
[MARKER ON OFF]	CALCulate#:MARKer[:STATe] OFF|0|ON|1
[COUPLED ON OFF]	CALCulate#:MARKer:COUPled[:STATe] OFF|0|ON|1
[MARKER X ENTRY]	CALCulate#:MARKer:X[:ABSolute] (-9.9e+37 ~ 9.9e+37 [HZ|S|VRMS|VPK|V|ORD|RPM])
[MKR VALUE ABS REL]	CALCulate#:MARKer:MODE ABSolute|RELative
[REFERENCE TO MARKER]	CALCulate#:MARKer:MODE RELative;REFerence:X (CALC#:MARK:X?);Y (CALC#:MARK:Y?)
[REFERENCE SETUP]	
[REFERENCE TO MARKER]	CALCulate#:MARKer:REFerence:MODE RELative;X (CALCulate#:MARKer:X[:ABSolute]);Y (CALCulate#:MARKer:Y[:ABSolute])
[REFERENCE X ENTRY]	CALCulate#:MARKer:REFerence:X (-9.9e+37 ~ 9.9e+37 [HZ|S|VRMS|VPK|V|ORD|RPM])
[REFERENCE Y ENTRY]	CALCulate#:MARKer:REFerence:Y (-9.9e+37 ~ 9.9e+37 [DBVRMS|VPK|DBVPK|V|DBV|EU|DBEU|VRMS])
[PEAK TRK ON OFF]	CALCulate#:MARKer:MAXimum:[GLOBal]TRACk OFF|0|ON|1
[NEXT PEAK RIGHT]	CALCulate#:MARKer:MAXimum:RIGHt
[NEXT PEAK LEFT]	CALCulate#:MARKer:MAXimum:LEFT
[MARKER TO PEAK]	CALCulate#:MARKer:MAXimum[:GLOBal]
Marker Fctn	
[MARKER FCTN OFF]	CALCulate#:MARKer:FUNCtion OFF
[HARMONIC MARKER]	
[FUNDAMNTL FREQUENCY]	CALCulate#:MARKer:HARMonic:FUNDamental (0 ~ 102400[HZ])
[NUMBER OF HARMONICS]	CALCulate#:MARKer:HARMonic:COUNt (0 ~ 400)
[COMPUTE OFF]	CALCulate#:MARKer:FUNCtion OFF
[THD]	CALCulate#:MARKer:FUNCtion THD
[HARMONIC POWER]	CALCulate#:MARKer:FUNCtion HPOWer
[BAND MARKER]	
[BAND SPAN]	CALCulate#:MARKer:BAND:SPAN (0 ~ 102400[HZ])
[BAND CENTER]	CALCulate#:MARKer:BAND:CENTer (0 ~ 102400[HZ])
[BAND START]	CALCulate#:MARKer:BAND:STARt (0 ~ 102400[HZ])
[BAND STOP]	CALCulate#:MARKer:BAND:STOP (0 ~ 102400[HZ])
[COMPUTE OFF]	CALCulate#:MARKer:FUNCtion OFF
[BAND POWER]	CALCulate#:MARKer:FUNCtion BPOWer
[RMS SQRT(PWR)]	CALCulate#:MARKer:FUNCtion BRMS

Front Panel Key	HP-IB Command
[SIDEBAND MARKER]	
[CARRIER FREQ]	CALCulate#:MARKer:SIDeband:CARRier (0 ~ 115000[HZ])
[SIDEBAND INCREMENT]	CALCulate#:MARKer:SIDeband:INCRement (0 ~ 102400[HZ])
[NUMBER OF SIDEBANDS]	CALCulate#:MARKer:SIDeband:COUNt (0 ~ 200)
[COMPUTE OFF]	CALCulate#:MARKer:FUNCtion OFF
[SIDEBAND POWER]	CALCulate#:MARKer:FUNCtion SPOWer
[WATERFALL MARKERS]	
[TRACE SELECT]	CALCulate2:WATerfall:TRACe:SELect (-9.9e+37 ~ 9.9e+37 [S | RPM | COUNT])
[SLICE SELECT]	CALCulate2:WATerfall:SLICe:SELect (-9.9e+37 ~ 9.9e+37 [HZ | S | VRMS | VPK | V | ORD | RPM])
[SAVE AND DISP DATA]	CALCulate2:WATerfall:TRACe:SELect#; :CALCulate2:WATerfall:TRACe:COPY D1; :CALCulate1:FEED 'D1'
[SAVE TO DATA REG]	
[SELECT SAVE REG]	
[D1]	CALCulate2:WATerfall:TRACe:SELect#; :CALCulate2:WATerfall:TRACe:COPY D1
[D2]	CALCulate2:WATerfall:TRACe:SELect#; :CALCulate2:WATerfall:TRACe:COPY D2
[D3]	CALCulate2:WATerfall:TRACe:SELect#; :CALCulate2:WATerfall:TRACe:COPY D3
[D4]	CALCulate2:WATerfall:TRACe:SELect#; :CALCulate2:WATerfall:TRACe:COPY D4
[D5]	CALCulate2:WATerfall:TRACe:SELect#; :CALCulate2:WATerfall:TRACe:COPY D5
[D6]	CALCulate2:WATerfall:TRACe:SELect#; :CALCulate2:WATerfall:TRACe:COPY D6
[D7]	CALCulate2:WATerfall:TRACe:SELect#; :CALCulate2:WATerfall:TRACe:COPY D7
[D8]	CALCulate2:WATerfall:TRACe:SELect#; :CALCulate2:WATerfall:TRACe:COPY D8
[WATERFALL SETUP]	
[MAX TRCES DISPLAYED]	DISPlay:WATerfall:COUNt (1 ~ 50000)
[TRACE HEIGHT]	DISPlay:WATerfall:HEIGht (1 ~ 100[PCT])
[HIDN LINE ON OFF]	DISPlay:WATerfall:HIDDen OFF | 0 | ON | 1
[BASELINE SUPPRESS]	DISPlay:WATerfall:BASeline (0 ~ 100[PCT])
[TIME PARAMTERS]	
[START TIME]	CALCulate#:MARKer:BAND:STARt (-9.9e+37 ~ 9.9e+37[S | REC | PNT])
[STOP TIME]	CALCulate#:MARKer:BAND:STOP (-9.9e+37 ~ 9.9e+37[S | REC | PNT])
[COMPUTE OFF]	CALCulate#:MARKer:FUNCtion OFF
[OVERSHOOT]	CALCulate#:MARKer:FUNCtion OVERshoot
[RISE TIME]	CALCulate#:MARKer:FUNCtion RTIMe
[SETTLING TIME]	CALCulate#:MARKer:FUNCtion STIMe
[DELAY TIME]	CALCulate#:MARKer:FUNCtion DTIMe

Front Panel Key	HP-IB Command
[GAIN/PHAS MARGINS]	
[START FREQUENCY]	CALCulate#:MARKer:BAND:STARt (0~102400[HZ])
[STOP FREQUENCY]	CALCulate#:MARKer:BAND:STOP (0~102400[HZ])
[COMPUTE OFF]	CALCulate#:MARKer:FUNCtion OFF
[COMPUTE MARGINS]	CALCulate#:MARKer:FUNCtion GMARgin
[FREQ & DAMPING]	
[START FREQUENCY]	CALCulate#:MARKer:BAND:STARt (0~102400[HZ])
[STOP FREQUENCY]	CALCulate#:MARKer:BAND:STOP (0~102400[HZ])
[COMPUTE OFF]	CALCulate#:MARKer:FUNCtion OFF
[COMPUTE COEFFICNT]	CALCulate#:MARKer:FUNCtion FREQuency

System Group

Front Panel Key	HP-IB Command
Preset	
[DO PRESET]	SYSTem:PRESet
[RECALL AUTOSTATE]	MMEMory:LOAD:STATe #,'<FILENAME>'
BASIC	
	PROGram:[SELected]STATe PAUSe
[Run Program 1]	PROGram:NAME PROGram1 ;STATe RUN
[Run Program 2]	PROGram:NAME PROGram2;STATe RUN
[Run Program 3]	PROGram:NAME PROGram3;STATe RUN
[Run Program 4]	PROGram:NAME PROGram4;STATe RUN
[Run Program 5]	PROGram:NAME PROGram5;STATe RUN
[DISPLAY SETUP]	
[OFF]	DISPlay:PROGram[:MODE] OFF
[FULL]	DISPlay:PROGram[:MODE] FULL
[UPPER]	DISPlay:PROGram[:MODE] UPPer
[LOWER]	DISPlay:PROGram[:MODE] LOWer
[CLEAR SCREEN]	DISPlay:ENABle OFF
[CONTINUE]	PROGram[:SELected]:STATe CONTinue
[INSTRUMNT BASIC]	
[RUN PROGRAM]	PROGram[:SELected]:STATe RUN
[SELECT PROGRAM]	
[Program 1]	PROGram[:SELected]:NAME PROG1
[Program 2]	PROGram[:SELected]:NAME PROG2
[Program 3]	PROGram[:SELected]:NAME PROG3
[Program 4]	PROGram[:SELected]:NAME PROG4
[Program 5]	PROGram[:SELected]:NAME PROG5
[LABEL PROGRAM]	PROGram[:SELected]:LABel <STRING>
[PRINT PROGRAM]	
[UTILITIES]	
[MEMORY SIZE]	PROGram[:SELected]:MALLocate (1200~ 7340030)
[AUTO MEMORY]	PROGram[:SELected]:MALLocate DEFault
[SCRATCH A]	
[PERFORM SCRATCH]	PROGram[:SELected]:DELete[:SELected]
[DEBUG]	
[RUN]	PROGram[:SELected]:STATe RUN
[CONTINUE]	PROGram[:SELected]:STATe CONTinue
[SINGLE STEP]	
[LAST ERROR]	
[EXAMINE VARIABLE]	PROGram[:SELected]:NUMBER
	PROGram[:SELected]:STRING
[RESET]	PROGram[:SELected]:STATe STOP

Front Panel Key	HP-IB Command
Save/ Recall	
[SAVE DATA]	
[SAVE TRACE]	
[INTO D1]	TRACe:DATA D1,TRACe#
[INTO D2]	TRACe:DATA D2,TRACe#
[INTO D3]	TRACe:DATA D3,TRACe#
[INTO D4]	TRACe:DATA D4,TRACe#
[INTO D5]	TRACe:DATA D5,TRACe#
[INTO D6]	TRACe:DATA D6,TRACe#
[INTO D7]	TRACe:DATA D7,TRACe#
[INTO D8]	TRACe:DATA D8,TRACe#
[INTO FILE]	MMEMory:STORe:TRACe#, '<FILENAME>'
[SAVE CAPTURE]	MMEMory:STORe:TCAPture '<FILENAME>'
[SAVE WATERFALL]	
[INTO W1]	TRACe:WATerfall:DATA W1 TRACe#
[INTO W2]	TRACe:WATerfall:DATA W2 TRACe#
[INTO W3]	TRACe:WATerfall:DATA W3 TRACe#
[INTO W4]	TRACe:WATerfall:DATA W4 TRACe#
[INTO W5]	TRACe:WATerfall:DATA W5 TRACe#
[INTO W6]	TRACe:WATerfall:DATA W6 TRACe#
[INTO W7]	TRACe:WATerfall:DATA W7 TRACe#
[INTO W8]	TRACe:WATerfall:DATA W8 TRACe#
[INTO FILE]	MMEMory:STORe::WATerfall TRACe#,, '<FILENAME>'
[CONTINUE SAVE]	MMEMory:STORe:CONTinue
[CATALOG ON OFF]	DISPlay:CONTents MMEMory
[SAVE STATE]	MMEMory:STORe:STATe #,'<FILENAME>'
[SAVE MORE]	
[SAVE UPPER LIM]	MMEMory:STORe:LIMit:UPPer:TRACe# '<FILENAME>'
[SAVE LOWER LIM]	MMEMory:STORe:LIMit:LOWer:TRACe# '<FILENAME>'
[SAVE MATH]	MMEMory:STORe:MATH '<FILENAME>'
[SAVE PROGRAM]	MMEMory:STORe:PROGram '<FILENAME>'
[SAVE FIT TABLE]	MMEMory:STORe:CFIT '<FILENAME>'
[SAVE SNTH TABLE]	MMEMory:STORe:SYNThesis '<FILENAME>'
[SAVE AUTOSTATE]	MMEMory:STORe:STATe (, 'NVRAM:AUTO_ST'
[RECALL DATA]	
[RECALL TRACE]	
[FROM FILE INTO D1]	MMEMory:LOAD:TRACe D1, '<FILENAME>'
[FROM FILE INTO D2]	MMEMory:LOAD:TRACe D2, '<FILENAME>'
[FROM FILE INTO D3]	MMEMory:LOAD:TRACe D3, '<FILENAME>'
[FROM FILE INTO D4]	MMEMory:LOAD:TRACe D4, '<FILENAME>'
[FROM FILE INTO D5]	MMEMory:LOAD:TRACe D5, '<FILENAME>'
[FROM FILE INTO D6]	MMEMory:LOAD:TRACe D6, '<FILENAME>'
[FROM FILE INTO D7]	MMEMory:LOAD:TRACe D7, '<FILENAME>'
[FROM FILE INTO D8]	MMEMory:LOAD:TRACe D8, '<FILENAME>'
[RECALL CAPTURE]	MMEMory:LOAD:TCAPture '<FILENAME>'

Front Panel Key	HP-IB Command
[RECALL WATERFALL]	
[FROM FILE INTO W1]	MMEMory:LOAD:WATerfall W1, '<FILENAME>'
[FROM FILE INTO W2]	MMEMory:LOAD:WATerfall W2, '<FILENAME>'
[FROM FILE INTO W3]	MMEMory:LOAD:WATerfall W3, '<FILENAME>'
[FROM FILE INTO W4]	MMEMory:LOAD:WATerfall W4, '<FILENAME>'
[FROM FILE INTO W5]	MMEMory:LOAD:WATerfall W5, '<FILENAME>'
[FROM FILE INTO W6]	MMEMory:LOAD:WATerfall W6, '<FILENAME>'
[FROM FILE INTO W7]	MMEMory:LOAD:WATerfall W7, '<FILENAME>'
[FROM FILE INTO W8]	MMEMory:LOAD:WATerfall W8, '<FILENAME>'
[CONTINUE RECALL]	MMEMory:LOAD:CONTinue
[RECALL STATE]	MMEMory:LOAD:STATe #, '<FILENAME>'
[RECALL MORE]	
[RECALL UPPER LIM]	MMEMory:LOAD:LIMit:UPPer:TRACe# '<FILENAME>'
[RECALL LOWER LIM]	MMEMory:LOAD:LIMit:LOWer:TRACe# '<FILENAME>'
[RECALL MATH]	MMEMory:LOAD:MATH '<FILENAME>'
[RECALL PROGRAM]	MMEMory:LOAD:PROGram '<FILENAME>'
[RCL FIT TABLE]	MMEMory:LOAD:CFIT '<FILENAME>'
[RCL SYNTH TABLE]	MMEMory:LOAD:SYNThesis '<FILENAME>'
[RECALL AUTOSTATE]	MMEMory:LOAD:STATe (, 'NVRAM:AUTO_ST'
[DEFAULT DISK]	MMEMory:MSIS 'NVRAM:'
[NON-VOL RAM DISK]	MMEMory:MSIS 'RAM:'
[VOLATILE RAM DISK]	MMEMory:MSIS 'INT:'
[INTERNAL DISK]	MMEMory:MSIS 'EXT:'
[EXTERNAL DISK]	

Front Panel Key	HP-IB Command
Disk Utilities	
[RENAME FILE]	
[ORIGINAL FILENAME]	<FILENAME>
[NEW FILENAME]	<FILE>
[PERFORM RENAME]	MMEMory:REName '<FILENAME>', '<FILE>'
[DELETE FILE]	MMEMory:DELete '<FILENAME>'
[DELETE ALL FILES]	MMEMory:DELete '<DISK>'
[COPY FILE]	
[SOURCE FILENAME]	<FILENAME>
[DESTIN FILENAME]	<FILENAME>
[PERFORM FILE COPY]	MMEMory:COPY '<FILENAME>' , '<FILENAME>'
[COPY ALL FILES]	
[SOURCE DISK]	<DISK>
[DESTIN DISK]	<DISK>
[PERFORM COPY ALL]	MMEMory:COPY '<DISK>', '<DISK>
[FORMAT DISK]	
[DISK TYPE LIF DOS]	
[RAM DISK SIZE]	<FORMAT OPTION> (65536 ~ 2097150)
[INTRLEAVE FACTOR]	<INTERLEAVE FACTOR>)0 ~ 255)
[PERFORM FORMAT]	MMEMory:INITialize '<DISK>', [LIF | DOS], <FORMAT OPTION>, <INTERLEAVE FACTOR>
[DEFAULT DISK]	MMEMory:MSIS '<DISK>'
Local/ HP-IB	
[ABORT HP-IB]	ABORt
[ANALYZER ADDRESS]	SYSTem:COMMunicate:GPIB:ADDRess (0 ~ 30)
[GPIB ECHO ON OFF]	DISPlay:GPIB:ECHO OFF | 0 | ON | 1
[PLOTTER ADDRESS]	HCOPy:PLOT:ADDRess (0 ~ 30)
[PRINTER ADDRESS]	HCOPy:PRINt:ADDRess (0 ~ 30)
[DISK ADDRESS]	MMEMory:DISK:ADDRess (0 ~ 30)
[DISK UNIT]	MMEMory:DISK:UNIT (0 ~ 10)

Front Panel Key	HP-IB Command
Plot/ Print	
[START PLOT/PRNT]	HCOPy[:IMMediate]
[PLOT DATA SELECT]	
[ALL]	HCOPy:SOURce ALL
[TRACE]	HCOPy:SOURce TRACe
[TRACE MARKER]	HCOPy:SOURce MARKer
[MARKER REFERENCE]	HCOPy:SOURce REFerence
[GRID]	HCOPy:SOURce GRID
[OUTPUT TO HPIB FILE]	HCOPy:DESTination HPIB|FILE
[DEVICE IS PLOT PRNT]	HCOPy:DEVice PLOT|PRINt
[OUTPUT FILENAME]	HCOPy:FILename '<FILENAME>'
[PLOT PEN SETUP]	
[DEFAULT PENS]	HCOPy:PLOT:PEN:DEFault
[TRACE A PEN]	HCOPy:PLOT:PEN:TRACe1 (1 ~ 16)
[TRACE B PEN]	HCOPy:PLOT:PEN:TRACe2 (1 ~ 16)
[TRACE A MKR PEN]	HCOPy:PLOT:PEN:MARKer1 (1 ~ 16)
[TRACE B MKR PEN]	HCOPy:PLOT:PEN:MARKer2 (1 ~ 16)
[ALPHA PEN]	HCOPy:PLOT:PEN:ALPHa (1 ~ 16)
[GRID PEN]	HCOPy:PLOT:PEN:GRID (1 ~ 16)
[TRACE A LINE TYPE]	
[SOLID]	HCOPy:PLOT:LTYPe:TRACe1 0
[DOTTED]	HCOPy:PLOT:LTYPe:TRACe1 1
[DASHED]	HCOPy:PLOT:LTYPe:TRACe1 2
[USER LINE TYPE]	HCOPy:PLOT:LTYPe:TRACe1 (-6 ~ 6)
[TRACE B LINE TYPE]	
[SOLID]	HCOPy:PLOT:LTYPe:TRACe2 0
[DOTTED]	HCOPy:PLOT:LTYPe:TRACe2 1
[DASHED]	HCOPy:PLOT:LTYPe:TRACe2 2
[USER LINE TYPE]	HCOPy:PLOT:LTYPe:TRACe2 (-6 ~ 6)
[MORE PLOT SETUP]	
[FAST (50 cm/s)]	HCOPy:PLOT:SPEed 50
[SLOW (10 cm/s)]	HCOPy:PLOT:SPEed 10
[DEFINE (? cm/s)]	HCOPy:PLOT:SPEed (1 ~ 100)
[PAGE EJCT ON OFF]	HCOPy:EJECt OFF|0|ON|1
[TIME STMP ON OFF]	HCOPy:TSTamp[:STATe] OFF|0|ON|1

Front Panel Key	HP-IB Command
System Utility	
[CALIBRATN]	
[SINGLE CAL]	CALibration:AUTO ONCE
[AUTO CAL ON OFF]	CALibration:AUTO OFF|0|ON|1 |ONCE
[SAVE CH1 CAL TRACE]	
[INTO D1]	TRACe:DATA D1,CAL1
[INTO D2]	TRACe:DATA D2,CAL1
[INTO D3]	TRACe:DATA D3,CAL1
[INTO D4]	TRACe:DATA D4,CAL1
[INTO D5]	TRACe:DATA D5,CAL1
[INTO D6]	TRACe:DATA D6,CAL1
[INTO D7]	TRACe:DATA D7,CAL1
[INTO D8]	TRACe:DATA D8,CAL1
[SAVE CH2 CAL TRACE]	
[INTO D1]	TRACe:DATA D1,CAL2
[INTO D2]	TRACe:DATA D2,CAL2
[INTO D3]	TRACe:DATA D3,CAL2
[INTO D4]	TRACe:DATA D4,CAL2
[INTO D5]	TRACe:DATA D5,CAL2
[INTO D6]	TRACe:DATA D6,CAL2
[INTO D7]	TRACe:DATA D7,CAL2
[INTO D8]	TRACe:DATA D8,CAL2
[BEEPER ON OFF]	SYSTem:BEEPer:STATe OFF|0|ON|1
[CLOCK SETUP]	
[TIME HHMM]	SYSTem:TIME (0 ~ 2359)
[DATE MMDDYY]	SYSTem:DATE (10100 ~ 123199)
[TIMESTAMP SETUP]	
[24 HR DD/MM/YY]	HCOPy:TSTamp:MODE CHOice1
[24 HR DD.MM.YY]	HCOPy:TSTamp:MODE CHOice2
[24 HR YY MM DD]	HCOPy:TSTamp:MODE CHOice3
[12 HR DD/MM/YY]	HCOPy:TSTamp:MODE CHOice4
[12 HR MM-DD-YY]	HCOPy:TSTamp:MODE CHOice5
[MEMORY USAGE]	DISPlay:CONTents MEM
[REMOVE CAPTURE]	
[CONFIRM REMOVE]	MEMory:DELete:NAME TCAPture
[REMOVE WATERFALL]	
[CONFIRM REMOVE]	MEMory:DELete:NAME WATerfall
[REMOVE WTFL REGS]	
[CONFIRM REMOVE]	MEMory:DELete:NAME WREGister
[REMOVE PROGRAMS]	
[CONFIRM REMOVE]	MEMory:DELete:NAME PROGram

Cross-Reference from Front-Panel Keys to HP-IB Commands
System Group

Front Panel Key	HP-IB Command
[FAULT LOG]	DISPlay:CONTents FTABle
[CLEAR FAULT LOG]	SYSTem:FLOG:CLEar
[S/N VERSION]	*IDN?
[SELF TEST]	
[QUICK CONF TEST]	*TST?
[LONG CONF TEST]	TEST:LONG
[TEST LOG]	DISPlay:CONTents TTAB
[CLEAR TEST LOG]	TEST:LOG:CLEar

Error Messages

Introduction

This appendix contains a listing of all the error messages that can be generated by the HP 35665A in response to HP-IB commands. Each message consists of an error number (always negative) followed by a string. The string contains a general description of the error followed by additional information about the cause of the error.

In this appendix, error numbers and their general descriptions are shown using a bold font. Phrases that complete the descriptions with additional information are grouped under the associated error number.

Up to five error messages are temporarily stored in the analyzer's error queue. They are returned to the controller, one message at a time, when you send the SYST:ERR query.

Command Errors

-100: Command error.

Command is query only.
Max Order x Max RPM is too high.
Min RPM can't be = Max RPM.
Min RPM too low.
Order [number] out of range.
Required Span is too low.
Time Trigger Level out of range.
Too large blocksize required.
delta Order/maxOrder/blocksize problem.

-104: Data type error.

This marker function is only valid for complex data.
This marker function is only valid for frequency domain data.
This marker function is only valid for frequency-response data.
This marker function is only valid for time domain data.

-108: Parameter not allowed.

-109: Missing parameter.

Missing parameter.
Parameter not allowed.

-113: Undefined header.

-131: Invalid suffix.

-141: Invalid character data.

-151: Invalid string data.

-161: Invalid block data.

Execution Errors

-200: Execution error.

Blank lines or Laurent terms cannot be changed, deleted or undeleted.

Bode diagram available in 2 channel FFT ANALYSIS and SWEPT SINE Instrument Modes only.

CAPTURE ABORTED !!

A measurement parameter was changed.

Pressed [ABORT CAPTURE] softkey.

Specified maximum RPM for tachometer < actual RPM; maximum RPM is too small.

Can't move the marker reference to the power band.

Capture cannot be used with Inst Mode ORDER ANALYSIS.

Capture cannot be used with Inst Mode SWEPT SINE.

Channel 1 capture data not available.

Channel 1 input cannot be disabled.

Channel 2 capture data not available.

Channel 2 trigger not permitted in Inst Mode 1 Channel.

Constant trace. No frequency and damping.

Conversion failure. Double precision overflow.

Curve fit may be poor. Coherence around peaks in weighting function is too low.

Delete program Not Allowed while RECORDING ENABLED.

Does not match marker units.

Download program Not Allowed while RECORDING ENABLED.

Exponent overflow. E+99 upper limit exceeded.

Exponent underflow. E-38 lower limit exceeded.

[Function register] definition is not valid for execution.

[Function register] execution requires invalid operator: [name of operator].

[Function register] execution requires recursion.

[Function register] execution requires unavailable operand: [measurement data selection].

[Function register] not defined.

FREQUENCY RESPONSE, COHERENCE, and CROSS SPECTRUM data are not valid with PEAK HOLD average.

Fell out of real time data acquisition.

File operation aborted, Capture changed.

File operation aborted, Capture in progress.

File operation aborted. Waterfall changed.

File operation aborted. Waterfall register changed.

File operation not completed.

Fit aborted. Curve fit algorithm failed. Must change input to curve fit.

Fit aborted. Trace A data cannot be Octave.

Fit aborted. Trace A data invalid.

Fit aborted. Trace A data must be 32-bit floating point.

Fit aborted. Trace A data must be LINEAR or LOG spaced.

Fit aborted. Trace A data must be complex.

Fit aborted. Trace A data must be in frequency domain.

Fit aborted. Trace A is a constant. Does not have finite poles or zeros.

Fit aborted. User span completely outside trace A boundaries.

Fit aborted. Weight data cannot be negative.

Fit aborted. Weight data spacing must be LIN or LOG.

Fit aborted. Weight register data invalid.
Fit or Synth table data invalid.
Fit or Synth table data invalid. Found an invalid floating point value.
Fit or Synth table data invalid. System order greater than 20.
Fit or Synth table data invalid. Table value outside valid range.
Fit table format invalid. Only pole zero format allowed.
Function definition is not valid.
Function definition is too long.
Function definition may not reference higher numbered functions.
HP-IB control not received.
Initial RPM [RPM value] < Min RPM.
Initial RPM [RPM value] > Max RPM.
Input Range tracking not valid on this data.
Instrument BASIC not installed.
Instrument State Controller Missing.
Instrument must be in either FFT ANALYSIS or SWEPT SINE mode.
Invalid Function Code.
Invalid Instrument State Request.
Invalid Instrument State Value.
Invalid Instrument State.
Invalid TRACE COORD selection.
LOG X AXIS invalid with NYQUIST or ORBIT.
Limit table invalid.
Limit testing is turned off.
Limits are undefined.
MEAS DATA selection not available with Inst Mode 1 Channel.
MEAS DATA selection not available with Inst Mode CORRELATN ANALYSIS.
MEAS DATA selection not available with Inst Mode FFT ANALYSIS.
MEAS DATA selection not available with Inst Mode HISTOGRAM.
MEAS DATA selection not available with Inst Mode OCTAVE ANALYSIS.
MEAS DATA selection not available with Inst Mode ORDER ANALYSIS.
MEAS DATA selection not available with Inst Mode SWEPT SINE.
MEAS DATA selection requires TRACK OFF.
MEAS DATA selection requires TRACK ON.
Marker Function invalid for Waterfall display Trace B.
Marker Function invalid for non frequency domain MEASurement DATA.
Marker Function invalid for non time domain MEASurement DATA.
Math not valid. [Specified math] operation requires [specified measurement] data.
Math not valid: Incompatible X-axis resolutions in '[specified math]' operation.
Math not valid: Incompatible data domains in '[specified math]' operation.
Math not valid: Incompatible number of elements in '[specified math]' operation.
NYQUIST not available on waterfall.
No CAPTURE data.
No CAPTURE data for channel 2.
No Coupled Markers allowed in with WATERFALL displays.
No Main Marker allowed with WATERFALL displays.
No Marker Functions allowed with ORBIT Meas Data or NYQUIST Trace Coord.
No REFERENCE indicator allowed with WATERFALL displays.
No REFERENCE indicator allowed with ORBIT Meas Data or NYQUIST Trace Coord.

Not enough CAPTURE data for any Measurement Result.
OPTION 1D0, Computed Order Tracking not installed.
OPTION 1D1, Realtime Octave Measurements not installed.
OPTION 1D2, Swept Sine Measurements not installed.
OPTION 1D3 Curve Fit/Synthesis not installed.
OPTION 1D4 Arbitrary Source not installed.
ORBIT not available in zoom mode.
ORBIT not available on waterfall.
Online measurement not possible. Decrease MaxOrder/delta Order ratio or use Time Capture Playback.
Only AUTOMATIC ARM can be used with EXTERNAL TRIGGER in ORDER ANALYSIS.
Only AUTOMATIC ARM can be used with averaging in ORDER ANALYSIS.
Pause the measurement before plotting a waterfall.
Plot/Print already in progress.
Plotter/printer not responding.
Program memory re-size Not Allowed while RECORDING ENABLED.
Program variable access Not Allowed while RECORDING ENABLED.
RPM or ramp rate too high.
Received HP-IB control without requesting it.
Recording mode canceled because: HP Instrument BASIC execution error. Refer to *HP Instrument BASIC Users Handbook*, Appendix A, "Error Messages."
SAVE/RECALL PROGRAM Not Allowed during power-on calibration.
SAVE/RECALL PROGRAM Not Allowed while RECORDING ENABLED.
Select AVERAGE ON and AVERAGE TYPE RMS or RMS EXPONENTIAL to view COHERENCE data.
Serial number must be 10 characters.
Synth table data invalid. Complex polynomial coefficients not allowed.
Synth table data invalid. Complex residue over real pole not allowed.
Synth table data invalid. Need a residue for every pole.
Synth table data invalid. Residue of 0 +/- j0 not allowed.
Table invalid. Complex coefficients not allowed.
Table invalid. Complex residue over real pole not allowed.
Table invalid. Need a residue for every pole.
Table order too large.
The imaginary term cannot be negative.
The maximum order for this column has been reached.
The total number of fixed poles or zeros in the table cannot be greater than the number entered under the curve fit setup key.
The value entered for number of poles cannot be less than the total fixed poles in the table.
The value entered for number of zeros cannot be less than the total fixed zeros in the table.
This data register contains data with non-uniform x axis spacing and therefore cannot be edited.
Too many RPM steps for order tracking.
WATERFALL STEPS must be 1.
Waterfall markers invalid for Trace A.
Waterfall markers require WATERFALL DISPLAY FORMAT.
Waterfall operation not available with measurement running.
Y-axis scale matching not possible with current Trace Coord selections for trace A and B.

-220: Parameter error.

Invalid Instrument State Parameter.

-221: Settings conflict.

Capture does not contain tach data.
Capture frequency is not compatible with this measurement.
Incompatible band limit mode.
Invalid program state change requested.
Log grid available only for linear magnitude.
LogX not valid for negative X axis.
Marker function result not available.
Marker is not On.
Meas Data selection invalid for Display Format WATERFALL.
Measurement mode incompatible with command.
No TRIGGER With SWEPT SINE Instrument Mode.
No WATERFALL display with ORBIT Meas Data or NYQUIST Trace Coord.
No WATERFALL display with ORDER ANALYSIS Instrument Mode and TRACK ON.
No WATERFALL display with SWEPT SINE Instrument Mode.
No WINDOW With HISTOGRAM Instrument Mode.
No WINDOW With OCTAVE Instrument Mode.
No WINDOW With SWEPT SINE Instrument Mode.
Offset Marker is not On.
SOURCE LEVEL is 0 Volts, measurement paused.
Synthesis table must be pole-zero.
Zoom capture data cannot be used with this measurement.

-222: Data out of range.

-224: Illegal parameter value.

REAL format length is only 32 and 64.

-230: Data corrupt or stale.

Trace contains invalid data.
Data does not contain SDF header information.

-240: Hardware error (see Fault Log)

-250: Mass storage error.

Bad or unformatted disk.
Disk file/unit possibly corrupt.
Disk operation aborted.
External disk not responding.
FORMAT aborted: file(s) are open.
File does not contain MATH definitions.
File does not contain LIMIT definitions.
File does not contain a CAPTURE.
File does not contain a STATE.
File does not contain a TRACE.
File does not contain a WATERFALL.
File system error.
HP-IB system controller needed.
INSTALL aborted - invalid option.
Illegal format parameter(s).
Improper file name.
Improper file type.
Improper mass storage unit specifier.
Invalid SDF file format.
Mass storage units must be the same when renaming.
No data in the limit file to save.
No memory available.
Operation failed on one (or more) files.
Operation not allowed while file(s) open.
Permission denied.
RAM disk re-size failed.
SDF feature NOT supported.
Source and destination units are same.
Too many disk units active.
Unexpected end of file.
Wildcard expands to more than one file.
Wildcard not allowed.

-251: Missing mass storage.

Mass storage unit not present.

-252: Missing media.

Disk not in drive

-253: Corrupt media.

Not a valid directory.

-254: Media full.

File too large, Press CONTINUE SAVE to split file.
Insert next disk with file '[filename]', press CONTINUE RECALL.
Insert next disk, press CONTINUE SAVE.
Insufficient disk space.

-255: Directory full.

Full directory.

-256: File name not found.

File name is undefined.

-257: File name error.

Duplicate file name.

-258: Media protected.

Write protected disk.

-280: Program error.

HP Instrument BASIC execution error. Refer to *HP Instrument BASIC Users Handbook*, Appendix A, "Error Messages."

-283: Illegal variable name.

-284: Program currently running.

-285: Program syntax error.

Downloaded program line must have a line number.
ERROR 949 Syntax error at cursor.

Device-Specific Errors

-310: System error.

Calibration DMA timeout.
Calibration Failure.
Calibration overloads.
Serial number already set.

-311: Memory error.

EEPROM not initialized correctly.
Out Of Memory.
Out of Memory. Need [specified number of] bytes. See MEMORY USAGE.

-315: Configuration memory lost.

Save system configuration to EEPROM failed.

-330: Self-test failed.

No apparent source amplitude control at Gilbert Cell output.
No apparent source amplitude control at output.

Query Errors

-400: Query error.
[HP-IB command which generated error].

-410: Query interrupted.

-420: Query unterminated.

-430: Query deadlocked.

-450: Query not allowed.

D

Instrument Modes

This table indicates HP-IB commands that are valid in each instrument mode. HP-IB commands that *cannot* be executed in a particular instrument mode are listed as "not valid."

Restrictions for use of these commands within the specific instrument mode may apply.

Table D-1. Valid HP-IB Commands For Each Instrument Mode

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
*CAL?						
*CLS						
*ESE						
*ESR?						
*IDN?						
*OPC						
*OPC?						
*OPT?						
*PCB						
*PSC						
*RST						
*SRE						
*STB?						
*TRG				not valid		
*TST?						
*WAI						
ABORt						
ARM[:IMMediate]				not valid		
ARM:RPM:INCRement				not valid		
ARM:RPM:MODE				not valid		
ARM:RPM:THReshold	not valid			not valid	not valid	
ARM:SOURce				not valid		
ARM:TIMer				not valid		
CALCulate:ACTive						
CALCulate:CFIT:ABORt						
CALCulate:CFIT:COPY						

Table D-1. Valid HP-IB Commands For Each Instrument Mode (continued)

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
CALCulate:CFIT:DATA						
CALCulate:CFIT:DESTination						
CALCulate:CFIT:FREQuency[:AUTO]						
CALCulate:CFIT:FREQuency:STARt						
CALCulate:CFIT:FREQuency:STOP						
CALCulate:CFIT:FSCale						
CALCulate:CFIT[:IMMediate]						
CALCulate:CFIT:ORDer:AUTO						
CALCulate:CFIT:ORDer:POLes						
CALCulate:CFIT:ORDer:ZERos						
CALCulate:CFIT:TDELay						
CALCulate:CFIT:WEIGht:AUTO						
CALCulate:CFIT:WEIGht:REGister						
CALCulate:DATA?						
CALCulate:DATA:HEADer:POINts?						
CALCulate:FEED						
CALCulate:FORMat						
CALCulate:LIMit:BEEP[:STATe]						
CALCulate:LIMit:FAIL?						
CALCulate:LIMit:LOWer:CLEar[:IMMediate]						
CALCulate:LIMit:LOWer:MOVE:Y						
CALCulate:LIMit:LOWer:REPort[:DATA]?						
CALCulate:LIMit:LOWer:REPort:YDATa?						
CALCulate:LIMit:LOWer:SEGMent						
CALCulate:LIMit:LOWer:SEGMent:CLEar						
CALCulate:LIMit:LOWer:TRACe[:IMMediate]						
CALCulate:LIMit:STATe						
CALCulate:LIMit:UPPer:CLEar[:IMMediate]						
CALCulate:LIMit:UPPer:MOVE:Y						
CALCulate:LIMit:UPPer:REPort[:DATA]?						
CALCulate:LIMit:UPPer:REPort:YDATa?						
CALCulate:LIMit:UPPer:SEGMent						
CALCulate:LIMit:UPPer:SEGMent:CLEar						
CALCulate:LIMit:UPPer:TRACe[:IMMediate]						
CALCulate:MARKer:BAND:STARt						
CALCulate:MARKer:BAND:STOP						
CALCulate:MARKer:COUPled[:STATe]						

Table D-1. Valid HP-IB Commands For Each Instrument Mode (continued)

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
CALCulate:MARKer:FUNCtion						
CALCulate:MARKer:FUNCtion:RESult?						
CALCulate:MARKer:HARMonic:COUNt						
CALCulate:MARKer:HARMonic:FUNDamental						
CALCulate:MARKer:MAXimum[:GLOBal]						
CALCulate:MARKer:MAXimum[:GLOB:TRACk						
CALCulate:MARKer:MAXimum:LEFT						
CALCulate:MARKer:MAXimum:RIGHt						
CALCulate:MARKer:MODE						
CALCulate:MARKer:POSition						
CALCulate:MARKer:POSition:POINt						
CALCulate:MARKer:REFerence:X						
CALCulate:MARKer:REFerence:Y						
CALCulate:MARKer:SIDeband:CARRier						
CALCulate:MARKer:SIDeband:COUNt						
CALCulate:MARKer:SIDeband:INCRement						
CALCulate:MARKer[:STATe]						
CALCulate:MARKer:X[:ABSolute]						
CALCulate:MARKer:X:RELative						
CALCulate:MARKer:Y[:ABSolute]?						
CALCulate:MARKer:Y:RELative						
CALCulate:MATH:CONStant						
CALCulate:MATH:DATA						
CALCulate:MATH[:EXPRession						
CALCulate:MATH:SELect						
CALCulate:MATH:STATe						
CALCulate:SYNThesis:COPY						
CALCulate:SYNThesis:DATA						
CALCulate:SYNThesis:DESTination						
CALCulate:SYNThesis:FSCale						
CALCulate:SYNThesis:GAIN						
CALCulate:SYNThesis[:IMMediate]						
CALCulate:SYNThesis:SPACing						
CALCulate:SYNThesis:TDELay						
CALCulate:SYNThesis:TTYPe						
CALCulate:UNIT:AMPLitude						
CALCulate:UNIT:ANGLe						

Table D-1. Valid HP-IB Commands For Each Instrument Mode (continued)

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
CALCulate:UNIT:VOLTage						
CALCulate:WATerfall:COUNt				not valid		
CALCulate:WATerfall[:DATA]						
CALCulate:WATerfall:SLICe:COPY				not valid		
CALCulate:WATerfall:SLICe:SELect				not valid		
CALCulate:WATerfall:SLICe:SELect:POINt				not valid		
CALCulate:WATerfall:TRACe:COPY				not valid		
CALCulate:WATerfall:TRACe:SELect				not valid		
CALCulate:WATerfall:TRACe:SELect:POINt				not valid		
CALibration[:ALL]?						
CALibration:AUTO						
DISPlay:ANNotation						
DISPlay:BODE		*not valid*	*not valid*		*not valid*	*not valid*
DISPlay:CONTents						
DISPlay:ENABle						
DISPlay:FORMat						
DISPlay:GPIB:ECHO						
DISPlay:PROGram[:MODE]						
DISPlay:WATerfall:BASeline				not valid		
DISPlay:WATerfall:COUNt				not valid		
DISPlay:WATerfall:HEIGht				not valid		
DISPlay:WATerfall:HIDDen				not valid		
DISPlay[:WINDow]:LIMit:STATe						
DISPlay[:WINDow]:TRAC:GRAT:GRID[:STAT]						
DISPlay[:WINDow]:TRACe:LABel						
DISPlay[:WINDow]:TRAC:LABel:DEF[:STATe]						
DISPlay[:WINDow]:TRACe:X:MATch						
DISPlay[:WINDow]:TRACe:X[:SCALe]:AUTO						
DISPlay[:WINDow]:TRACe:X[:SCALe]:LEFT						
DISPlay[:WINDow]:TRACe:X[:SCALe]:RIGHt						
DISPlay[:WINDow]:TRACe:X:SPACing						
DISPlay[:WINDow]:TRACe:Y:MATch						
DISPlay[:WINDow]:TRACe:Y[:SCALe]:AUTO						
DISPlay[:WINDow]:TRAC:Y[:SCAL]:BOTTom						
DISPlay[:WINDow]:TRACe:Y[:SCALe]:CENTer						
DISPlay[:WINDow]:TRAC:Y[:SCAL]:PDIVision						
DISPlay[:WIND]:TRAC:Y[:SCAL]:REFerence						

Table D-1. Valid HP-IB Commands For Each Instrument Mode (continued)

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
DISPlay[:WINDow]:TRACe:Y[:SCALe]:TOP						
DISPlay[:WINDow]:TRACe:Y:SPACing						
FORMat[:DATA]						
HCOPy:DESTination						
HCOPy:DEVice						
HCOPy:EJECt						
HCOPy:FILename						
HCOPy[:IMMediate]						
HCOPy:PLOT:ADDRess						
HCOPy:PLOT:LTYPe:TRACe						
HCOPy:PLOT:PEN:ALPHa						
HCOPy:PLOT:PEN:DEFault						
HCOPy:PLOT:PEN:GRID						
HCOPy:PLOT:PEN:MARKer						
HCOPy:PLOT:PEN:TRACe						
HCOPy:PLOT:SPEed						
HCOPy:PRINt:ADDRess						
HCOPy:SOURce						
HCOPy:TSTamp:MODE						
HCOPy:TSTamp[:STATe]						
INITiate:CONTinuous						
INITiate[:IMMediate]						
INPut:BIAS[:STATe]						
INPut:COUPling						
INPut:FILTer:AWEighting[:STATe]						not valid
INPut:FILTer[:LPASs][:STATe]						not valid
INPut:LOW						
INPut:REFerence:DIRection						
INPut:REFerence:POINt						
INPut[:STATe]						
INSTrument:NSELect						
INSTrument:SELect						
MEMory:CATalog[:ALL]						
MEMory:CATalog:NAMe						
MEMory:DELete:ALL						
MEMory:DELete[:NAMe]						
MEMory:FREE[:ALL]						

Table D-1. Valid HP-IB Commands For Each Instrument Mode (continued)

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
MMEMory:COPY						
MMEMory:DELete						
MMEMory:DISK:ADDRess						
MMEMory:DISK:UNIT						
MMEMory:FSYStem						
MMEMory:INITialize						
MMEMory:LOAD:CFIT						
MMEMory:LOAD:CONTinue						
MMEMory:LOAD:LIMit:LOWer:TRACe						
MMEMory:LOAD:LIMit:UPPer:TRACe-						
MMEMory:LOAD:MATH						
MMEMory:LOAD:PROGram						
MMEMory:LOAD:STATe						
MMEMory:LOAD:SYNThesis						
MMEMory:LOAD:TCAPture						
MMEMory:LOAD:TRACe						
MMEMory:LOAD:WATerfall						
MMEMory:MSIS						
MMEMory:REName						
MMEMory:STORe:CFIT						
MMEMory:STORe:CONTinue						
MMEMory:STORe:LIMit:LOWer:TRACe						
MMEMory:STORe:LIMit:UPPer:TRACe						
MMEMory:STORe:MATH						
MMEMory:STORe:PROGram						
MMEMory:STORe:STATe						
MMEMory:STORe:SYNThesis						
MMEMory:STORe:TCAPture						
MMEMory:STORe:TRACe						
MMEMory:STORe:WATerfall						
OUTPut[:STATe]						
PROGram:EXPLicit:DEFine						
PROGram[:SELected]:DEFine						
PROGram[:SELected]:DELete:ALL						
PROGram[:SELected]:DELete[:SELected]						
PROGram[:SELected]:LABel						
PROGram[:SELected]:MALLocate						

Table D-1. Valid HP-IB Commands For Each Instrument Mode (continued)

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
PROGram[:SELected]:NAME						
PROGram[:SELected]:NUMBer						
PROGram[:SELected]:STATe						
PROGram[:SELected]:STRing						
[SENSe:]AVERage:CONFidence	not valid			not valid	not valid	not valid
[SENSe:]AVERage:COUNt				not valid		not valid
[SENSe:]AVERage:HOLD	not valid			not valid	not valid	not valid
[SENSe:]AVERage:IMPulse	not valid			not valid	not valid	not valid
[SENSe:]AVERage:IRESult:RATE				not valid		not valid
[SENSe:]AVERage:IRESult[:STATe]						
[SENSe:]AVERage:PREView		not valid	not valid	not valid	not valid	not valid
[SENSe:]AVERage:PREView:ACCept		not valid	not valid	not valid	not valid	not valid
[SENSe:]AVERage:PREView:REJect		not valid	not valid	not valid	not valid	not valid
[SENSe:]AVERage:PREView:TIME		not valid	not valid	not valid	not valid	not valid
[SENSe:]AVERage[:STATe]				not valid		not valid
[SENSe:]AVERage:TCONtrol				not valid		
[SENSe:]AVERage:TIME	not valid			not valid	not valid	
[SENSe:]AVERage:TYPE				not valid		not valid
[SENSe:]FEED				not valid		
[SENSe:]FREQuency:CENTer		not valid	not valid		not valid	not valid
[SENSe:]FREQuency:MANual	not valid	not valid	not valid		not valid	not valid
[SENSe:]FREQuency:RESolution	not valid	not valid	not valid		not valid	not valid
[SENSe:]FREQuency:RESolution:AUTO	not valid	not valid	not valid		not valid	not valid
[SENSe:]FREQuency:RES:AUTO:MCHange	not valid	not valid	not valid		not valid	not valid
[SENSe:]FREQuency:RES:AUTO:MINimum	not valid	not valid	not valid		not valid	not valid
[SENSe:]FREQuency:RESolution:OCTave	not valid		not valid	not valid	not valid	not valid
[SENSe:]FREQuency:SPAN		not valid	not valid		not valid	
[SENSe:]FREQuency:SPAN:FULL		not valid	not valid	not valid		not valid
[SENSe:]FREQuency:SPAN:LINK		not valid	not valid		not valid	not valid
[SENSe:]FREQuency:STARt			not valid		not valid	not valid
[SENSe:]FREQuency:STEP[:INCRement]		not valid	not valid		not valid	not valid
[SENSe:]FREQuency:STOP			not valid		not valid	not valid
[SENSe:]HISTogram:BINS	not valid	not valid	not valid	not valid	not valid	
[SENSe:]ORDer:MAXimum	not valid	not valid		not valid	not valid	not valid
[SENSe:]ORDer:RESolution	not valid	not valid		not valid	not valid	not valid
[SENSe:]ORDer:RPM:MAXimum	not valid	not valid		not valid	not valid	not valid
[SENSe:]ORDer:RPM:MINimum	not valid	not valid		not valid	not valid	not valid

Table D-1. Valid HP-IB Commands For Each Instrument Mode (continued)

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
[SENSe:]ORDer:TRACk	not valid	not valid		not valid	not valid	
[SENSe:]REJect:STATe		not valid	not valid	not valid		not valid
[SENSe:]SWEep:DIRection	not valid	not valid	not valid		not valid	not valid
[SENSe:]SWEep:DWELl	not valid	not valid	not valid		not valid	not valid
[SENSe:]SWEep:MODE	not valid	not valid	not valid		not valid	not valid
[SENSe:]SWEep:OVERlap		not valid	not valid	not valid		not valid
[SENSe:]SWEep:SPACing	not valid	not valid	not valid		not valid	not valid
[SENSe:]SWEep:STIMe	not valid	not valid	not valid		not valid	not valid
[SENSe:]SWEep:TIME	not valid	not valid	not valid	not valid		
[SENSe:]TCAPture:ABORt				not valid		
[SENSe:]TCAPture:DELete						
[SENSe:]TCAPture[:IMMediate]				not valid		
[SENSe:]TCAPture:LENGth						
[SENSe:]TCAPture:MALLocate						
[SENSe:]TCAPture:STARt						
[SENSe:]TCAPture:STOP						
[SENSe:]TCAPture:TACHometer:RPM:MAX						
[SENSe:]TCAPture:TACHometer[:STATe]						
[SENSe:]VOLTage:RANGe:AUTO						
[SENSe:]VOLTage:RANGe:UNIT:MODE						
[SENSe:]VOLTage:RANGe:UNIT:USER:LABel						
[SENSe:]VOLT:RANG:UNIT:USER:SFACtor						
[SENSe:]VOLT:RANG:UNIT:USER[:STATe]						
[SENSe:]VOLTage:RANGe[:UPPer]						
[SENSe:]WINDow:EXPonential		not valid	not valid	not valid	not valid	not valid
[SENSe:]WINDow:FORCe		not valid	not valid	not valid	not valid	not valid
[SENSe:]WINDow:ORDer:DC	not valid	not valid		not valid	not valid	not valid
[SENSe:]WINDow[:TYPE]		not valid		not valid		not valid
SOURce:BURSt		not valid	not valid	not valid		
SOURce:FREQuency[:CW]				not valid		
SOURce:FREQuency:FIXed				not valid		
SOURce:FUNCtion[:SHAPe]				not valid		
SOURce:USER[:REGister]		not valid	not valid	not valid		
SOURce:USER:REPeat		not valid	not valid	not valid		
SOURce:VOLTage[:LEVel]:AUTO	not valid	not valid	not valid		not valid	not valid
SOUR:VOLT[:LEV][:IMMediate][:AMPLitude]						
SOURce:VOLTage[:LEVel]:REFerence	not valid	not valid	not valid		not valid	not valid

Table D-1. Valid HP-IB Commands For Each Instrument Mode (continued)

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
SOUR:VOLTage[:LEVel]:REFerence:CHANnel	not valid	not valid	not valid		not valid	not valid
SOUR:VOLT[:LEVel]:REFerence:TOLerance	not valid	not valid	not valid		not valid	not valid
SOURce:VOLTage:LIMit[:AMPLitude]	not valid	not valid	not valid		not valid	not valid
SOURce:VOLTage:LIMit:INPut	not valid	not valid	not valid		not valid	not valid
SOURce:VOLTage:SLEW	not valid	not valid	not valid		not valid	not valid
STATus:DEVice:CONDition?						
STATus:DEVice:ENABle						
STATus:DEVice[:EVENt]?						
STATus:DEVice:NTRansition						
STATus:DEVice:PTRansition						
STATus:OPERation:CONDition?						
STATus:OPERation:ENABle						
STATus:OPERation[:EVENt]?						
STATus:OPERation:NTRansition						
STATus:OPERation:PTRansition						
STATus:PRESet						
STATus:QUEStionable:CONDition?						
STATus:QUEStionable:ENABle						
STATus:QUEStionable[:EVENt]?						
STATus:QUEStionable:LIMit:CONDition?						
STATus:QUEStionable:LIMit:ENABle						
STATus:QUEStionable:LIMit[:EVENt]?						
STATus:QUEStionable:LIMit:NTRansition						
STATus:QUEStionable:LIMit:PTRansition						
STATus:QUEStionable:NTRansition						
STATus:QUEStionable:PTRansition						
STATus:QUEStionable:VOLTage:CONDition?						
STATus:QUEStionable:VOLTage:ENABle						
STATus:QUEStionable:VOLTage[:EVENt]?						
STATus:QUEStionable:VOLTage:NTRansition						
STATus:QUEStionable:VOLTage:PTRansition						
STATus:USER:ENABle						
STATus:USER[:EVENt]?						
STATus:USER:PULSe-?						
SYSTem:BEEPer[:IMMediate]						
SYSTem:BEEPer:STATe						
SYSTem:COMMunicate:GPIB:ADDRess						

Table D-1. Valid HP-IB Commands For Each Instrument Mode (continued)

HP-IB Command	Instrument Mode (INST:SEL)					
	FFT	OCT	ORD	SINE	CORR	HIST
SYSTem:DATE						
SYSTem:ERRor						
SYSTem:FLOG:CLEar						
SYSTem:KEY						
SYSTem:KLOCk						
SYSTem:PRESet						
SYSTem:SET						
SYSTem:TIME						
TEST:LOG:CLEar						
TEST:LONG						
TEST:LONG:RESult?						
TRACe[:DATA]						
TRACe:WATerfall[:DATA]						
TRACe:X[:DATA]						
TRACe:X:UNIT						
TRACe:Z[:DATA]						
TRACe:Z:UNIT						
TRIGger[:IMMediate]				not valid		
TRIGger:LEVel		not valid	not valid	not valid		
TRIGger:LEVel:TTL	not valid			not valid	not valid	not valid
TRIGger:SLOPe		not valid	not valid	not valid		
TRIGger:SOURce				not valid		
TRIGger:STARt			not valid	not valid		
TRIGger:TACHometer:HOLDoff				not valid		
TRIGger:TACHometer:LEVel				not valid		
TRIGger:TACHometer:PCOunt				not valid		
TRIGger:TACHometer:RANGe				not valid		
TRIGger:TACHometer[:RPM]?				not valid		
TRIGger:TACHometer:SLOPe				not valid		

Determining Units

The following tables (table E-1 - table E-6) show the units available for the Y-axis. The tables indicate which Y-axis units are available for each measurement data selection and which HP-IB commands result in setting the Y-axis units. See the "Command Reference" in this guide for a description of these commands.

Table E-1 specifies the default Y-axis unit for each measurement data type (specified with the CALC:FEED command) and each trace coordinate system (specified with the CALC:FORMat command).

You can change the Y-axis unit for some types of measurement data with the CALC:UNIT:VOLT and CALC:UNIT:AMPLitude commands. See table E-2 for vertical unit selection and tables E-3 — E-6 for amplitude selection.

Note

If a measurement data type is *not* listed in table E-2, the default unit listed in table E-1 is *always* used. You cannot select the Y-axis (vertical) units in these cases. Tables E-3 - E-6 *only* apply to the measurement data listed in table E-2.

The analyzer determines the default Y-axis unit based upon the specified measurement data and trace coordinate system. Table E-1 lists these default Y-axis units.

Units for data registers (D[1|2... |8]) and waterfall registers (W[1|2| ... |8]) are dependent upon the type of measurement data stored in the register. See the appropriate measurement data row for valid unit selections.

Table E-1. Default Y-axis Units for Measurement Data and Trace Coordinates.

Measurement Data	Specified Trace Coordinates			
Type of Measurement CALC:FEED?	Linear or Logarithmic Magnitude CALC:FORM MLIN	dB Magnitude CALC:FORM MLOG	Phase/Unwrapped Phase CALC:FORM [PHAS|UPH] *	Real/Imaginary CALC:FORM [REAL|IMAG]
auto correlation CALC:FEED 'XTIM:CORR'	VPK2	DBV	DEG	VPK2
capture buffer CALC:FEED 'TCAP'	VPK	DBV	DEG	VPK
coherence CALC:FEED 'XFR:POW:COH'	none	none	DEG	none
composite power CALC:FEED 'XFR:POW:COMP'	VRMS	DBVRMS	DEG	VRMS
cross correlation CALC:FEED 'XTIM:CORR:CROS'	VPK2	DBV	DEG	VPK2
cross spectrum ** CALC:FEED 'XFR:POW:CROS'	VRMS2	DBVRMS	DEG	VRMS2
cumulative density function CALC:FEED ' XTIM:VOLT:CDF'	none	DB	DEG	none
frequency response CALC:FEED 'XFR:POW:RAT'	none	DB	DEG	none
histogram CALC:FEED 'XTIM:VOLT:HIST'	COUNT	DBCOUNT	DEG	COUNT
linear spectrum CALC:FEED ' XFR:POW:LIN'	VRMS	DBVRMS	DEG	VRMS
normalized variance CALC:FEED 'XFR:POW:VAR'	none	DB	DEG	none
orbits CALC:FEED 'XVOL:VOLT'	not valid	not valid	not valid	VPK
order track CALC:FEED 'XORD:TRACK'	VRMS	DBVRMS	DEG	VRMS
power spectrum CALC:FEED 'XFR:POW'	VRMS2	DBVRMS	DEG	VRMS2
probability density function CALC:FEED 'XTIM:VOLT:PDF'	1/V	DB1/V	DEG	1/V

Table E-1. Default Y-axis Units for Measurement Data and Trace Coordinates (continued)

Measurement Data	Specified Trace Coordinates			
Type of Measurement CALC:FEED?	Linear or Logarithmic Magnitude CALC:FORM MLIN	dB Magnitude CALC:FORM MLOG	Phase/Unwrapped Phase CALC:FORM [PHAS|UPH] *	Real/Imaginary CALC:FORM [REAL|IMAG]
rpm profile CALC:FEED 'XRPM:PROF'	COUNT***	DBCOUNT	DEG	COUNT
time CALC:FEED 'XTIM:VOLT'	VPK	DBV	DEG	VPK
windowed time CALC:FEED 'XTIM:VOLT:WIND'	VPK	DBV	DEG	VPK

* This is the default unit. You can select radians (RAD).

** This is the default unit. You can specify VPK using the CALC:UNIT AMPL command.

*** This is the default unit. The unit is S (seconds) if order track is on (SENSe:ORDer:TRACk:STATe ON).

For linear spectrum and power spectrum measurement data, the Y-axis vertical unit is set with the CALCulate:UNIT:VOLTage command. See table E-2 for valid Y-axis vertical units. An "X" indicates a valid selection.

Table E-2. Valid Unit Selections for CALC:UNIT:VOLT

Measurement Data CALC:FEED command (INST:SEL command)	CALC:UNIT:VOLT				
	V	V2	V/RTHZ	V2/HZ	V2S/HZ
Composite Power CALC:FEED 'XFR:POW:COMP' (INST:SEL ORD)	X	X			
Linear Spectrum CALC:FEED 'XFR:POW:LIN' (INST:SEL FFT)	X	X	X	X	X
Linear Spectrum CALC:FEED 'XFR:POW:LIN' (INST:SEL SINE)	X	X			
Order Track CALC:FEED 'XORD:TRACK' (INST:SEL ORD)	X	X			
Power Spectrum CALC:FEED 'XFR:POW' (INST:SEL FFT)	X	X	X	X	X
Power Spectrum CALC:FEED 'XFR:POW' (INST:SEL OCT)	X	X	X	X	X
Power Spectrum CALC:FEED 'XFR:POW' (INST:SEL ORD)	X	X			
Power Spectrum CALC:FEED 'XFR:POW' (INST:SEL ORD	X	X			

Units for data registers (D[1|2... |8]) and waterfall registers (W[1|2| ... |8]) are dependent upon the type of measurement data stored in the register. See the appropriate measurement data row for valid unit selections.

The CALCulate:UNIT:VOLTage command is only valid for the measurement data listed in the table. It is not valid for any other types of measurement data. If you do not know the current measurement data selection, send the query, CALCulate[1|2]:FEED?.

For linear spectrum and power spectrum measurement data, the unit of amplitude for the Y-axis scale is set with the CALCulate:UNIT:AMPLitude command. In addition, tables E-5 and E-6 show the unit of amplitude with engineering units (enabled with the [SENSe:]VOLT:RANG:UNIT:USER command).

Table E-3. Y-axis Units when CALC:UNIT:AMPL PEAK and VOLT:RANG:UNIT:USER OFF.

CALC:UNIT:VOLT	Trace Coordinate Setting	
	CALC:FORM [MLIN|REAL|IMAG]	CALC:FORM MLOG
V	V VPK	DBV
V2	V2	DBV
V/RTHZ	V/RTHZ	DBV/RTHZ
V2/HZ	V2/HZ	DBV/RTHZ
V2S/HZ	V2S/HZ	DBV2S/HZ

Table E-4. Y-axis Units when CALC:UNIT:AMPL RMS and VOLT:RANG:UNIT:USER OFF.

CALC:UNIT:VOLT	Trace Coordinate Setting	
	CALC:FORM [MLIN|REAL|IMAG]	CALC:FORM MLOG
V	VRMS	DBVRMS
V2	VRMS2	DBVRMS
V/RTHZ	VRMS/RTHZ	DBVRMS/RTHZ
V2/HZ	VRMS2/HZ	DBVRMS/RTHZ
V2S/HZ	VRMS2S/HZ	DBVRMS2S/HZ

Table E-5. Y-axis Units when CALC:UNIT:AMPL PEAK and VOLT:RANG:UNIT:USER ON.

CALC:UNIT:VOLT	Trace Coordinate Setting	
	CALC:FORM [MLIN|REAL|IMAG]	CALC:FORM MLOG
V	EU	DBEU
V2	EU2	DBEU
V/RTHZ	EU/RTHZ	DBEU/RTHZ
V2/HZ	EU2/HZ	DBEU/RTHZ
V2S/HZ	EU2S/HZ	DBEU2S/HZ

Table E-6. Y-axis Units when CALC:UNIT:AMPL RMS and VOLT:RANG:UNIT:USER ON.

CALC:UNIT:VOLT	Trace Coordinate Setting	
	CALC:FORM [MLIN|REAL|IMAG]	CALC:FORM MLOG
V	EURMS	DBEURMS
V2	EURMS2	DBEURMS
V/RTHZ	EURMS/RTHZ	DBEURMS/RTHZ
V2/HZ	EURMS2/HZ	DBEURMS/RTHZ
V2S/HZ	EURMS2S/HZ	DBEURMS2S/HZ

Index