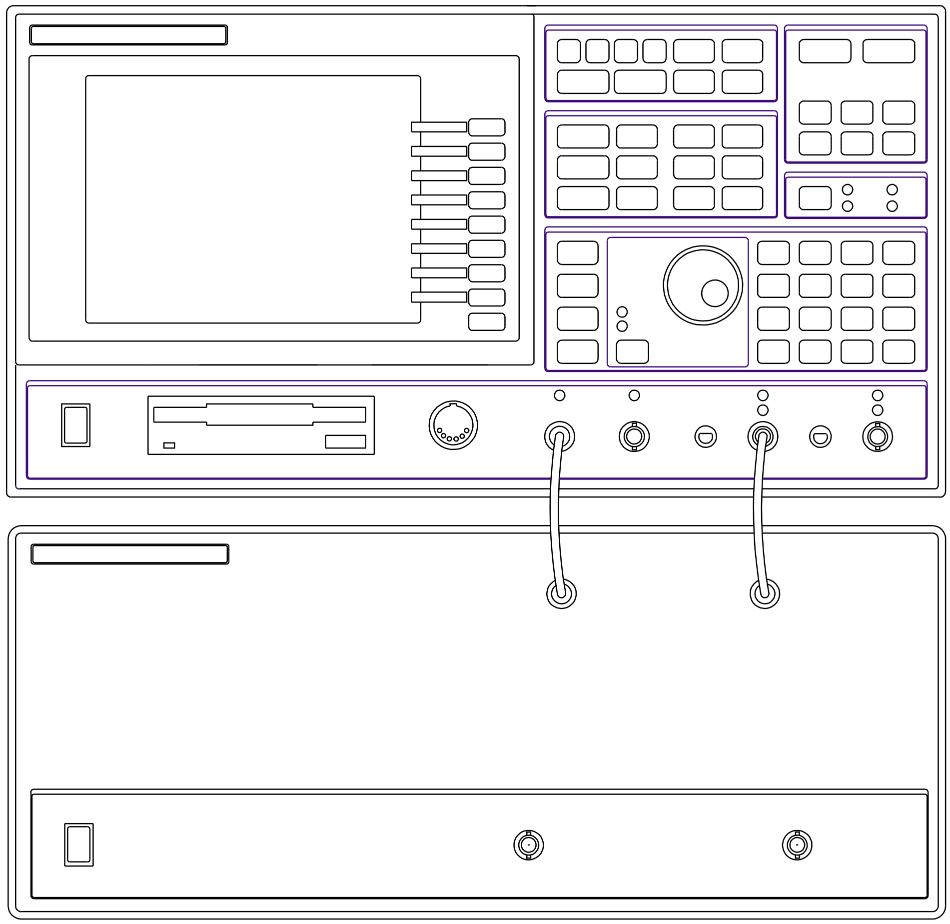
Agilent Technologies 89400-Series GPIB Command Reference



Agilent Technologies

Agilent Technologies Part Number 89400-90039
For instruments with firmware version A.08.00
Printed in U.S.A.

Print Date: May, 2000

8600 Soper Hill Road Everett, Washington 98205-1209 U.S.A.

TABLE OF CONTENTS

COMMAND LISTING

1

GPIB Programming with Agilent 89400-Series Analyzers

Introduction to GPIB Programming

For an introduction to GPIB programming, read the *GPIB Programmer's Guide*. It is provided to help those not familiar with GPIB programming or remote control of an instrument. The book introduces the basic concepts of GPIB programming and describes the Standard Commands for Programmable Instruments (SCPI). It also describes how to operate an instrument in an GPIB system and how to transfer data between an external controller and an instrument.

In This Book

This is the *GPIB Command Reference* for the Agilent 89400-Series Vector Signal Analyzer. It contains the command syntax, structure, and a detailed description of each GPIB command available for the Agilent 89400-Series analyzers.

Chapter 1 presents GPIB programming information specific to the analyzer:

- How to connect the analyzer to an external controller and verify that it works.
- A description of the analyzer's status registers.

Chapter 2 describes the SCPI instrument model.

Chapter 3 is the command reference. It contains a detailed description of each GPIB command. The commands are organized alphabetically.

Appendix A discusses SCPI error messaging and lists error messages reported by this instrument.

Appendix B is an example program listing which demonstrates the sockets feature of option UFG.

Also, a card containing a command quick-reference is included.

Note Most front-panel key presses that change the configuration or initiate an action correspond to one SCPI command but some require more than one (see [SENSe:]AVERage:TYPE). To display the SCPI command strings that correspond to front-panel keys, activate the SCPI command echo by pressing [**Local/Setup**], [SCPI cmd echo **on**] or sending "`SYST:GPIB:ECHO 1`". Subsequent SCPI command strings are displayed in the upper-lefthand corner of the display.

GPIB Setup and Verification

This section contains a procedure for configuring the Agilent 89400-Series Vector Signal Analyzer and an external controller in a simple GPIB system. Although an HP 9000 Series 340 computer is used in this procedure example, other computers that support an GPIB interface can be used. If you are using a computer other than the Series 340, the configuration procedure should be used as a general guide. Consult your computer's documentation for more complete information.

Equipment and Software Required

- Agilent 89400-Series Vector Signal Analyzer
- HP 9000 Series 340 computer
- GPIB cable (Agilent 10833A, B, C or D)
- Agilent Technologies BASIC programming language

Procedure

1. Turn off the analyzer and the HP 9000 Series 340

2. Connect them with the GPIB cable.

3. Turn on the computer. If necessary, load BASIC. 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 may require more language extensions. For a complete list of loaded language extensions, execute the BASIC command:

   ```
   LIST BIN
   ```

4. Turn on the analyzer. When the startup processes are complete, press the [**Local/Setup**] hardkey.

5. Verify that the analyzer's address is set to 19. The current address setting is displayed in the [analyzer adrs] softkey. You can change the address by pressing the [analyzer adrs] softkey, entering the number using the numeric keypad, and then pressing the [enter] softkey. Instructions in the verification procedure assume that the analyzer address is set to 19.

6. Verify that the analyzer is set to the addressable-only mode. The softkey labels that appear when you press the [**Local /Setup**] hardkey include [system controller] and [addressable only]. Only one of these two softkeys can be selected at a time, and the one that is selected is highlighted with yellow text. If [addressable only] is not selected, then press its softkey.

7. Type `REMOTE 719` on the computer and press Enter. The Remote indicator (near the [**Local/Setup**] hardkey) should light up. This indicates that the analyzer is in remote control.

GPIB Interface Capabilities

The Agilent 89400A-Series analyzers have 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**	no Parallel Poll capability
PP1*	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 Instrument BASIC program is running
** only when an Instrument BASIC program is *not* running

General Status Register Model

Overview

The general status register model, shown in figure 1, is the building block of the analyzer'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 transfer within a register set starts at the condition register and ends at the register summary bit (see figure 2). The information is controlled by altering bits in the transition and enable registers.

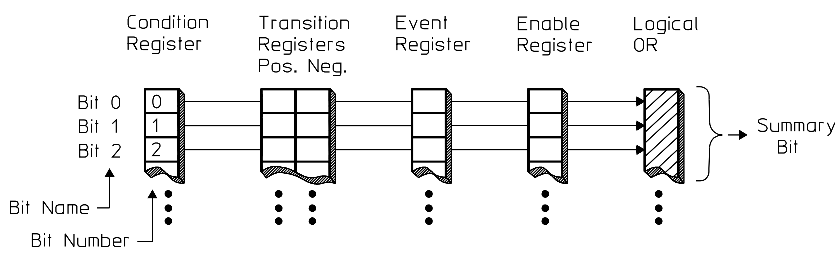


Figure 1. General Model of a Status Register Group

When a status group is implemented in a SCPI instrument, it always contains all of the component registers. There is *not* always a corresponding command to read or write every register, however. The flow within a status group starts at the condition register and ends at the register summary bit as shown in figure 2. You control the flow by setting the bit patterns in the transition and enable registers.

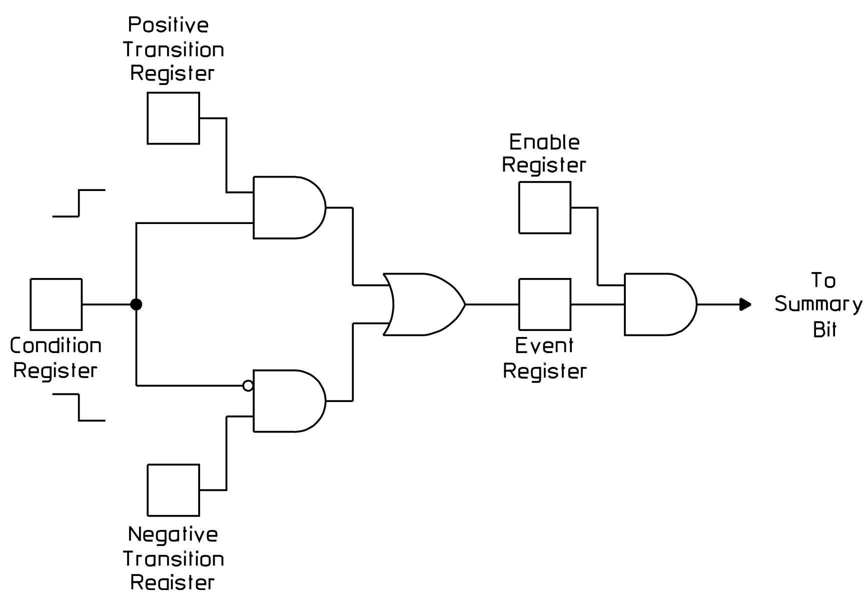


Figure 2. Flow of Information Within a Register Set

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. They specify which types of changes in the condition register set corresponding bits in the event register. Transition registers are read-write.

Transition register bits may be configured to signal positive changes, negative changes, or both. Positive changes in the state of a condition bit (0 to 1) are 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 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.

Transition registers are not affected by *CLS (clear status) or queries. They are set to default values when power is turned on and after receiving *RST. Some registers have a fixed setting if there are no commands to access a particular transition register. This fixed setting, along with dependent values, are specified in the *GPIB Command Reference*.

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.

Notes

Reading the Event Register, clears the Event Register.
All event registers are cleared by the *CLS command.

Enable Register

Enable registers control the reporting of events (which are latched condition register information) 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.

Enable registers are cleared by the *CLS (clear status) command. Querying enable registers does not affect them. All enable registers have commands to read and write them.

An Example Sequence

The example illustrated in figure 3 presents four different cases, as defined by the settings in the leftmost block; each row contains a slightly different variation of the sequence as defined by the settings of the transition and enable registers. The changing state of the condition in question is shown at the bottom of the figure. A small binary table shows the state of the chosen bit in each status register at selected times (T1–T5). There are two "events" or changes in the condition being monitored that cause register values to change. There are also three reads of the event register (indicated with asterisks), which clear it.

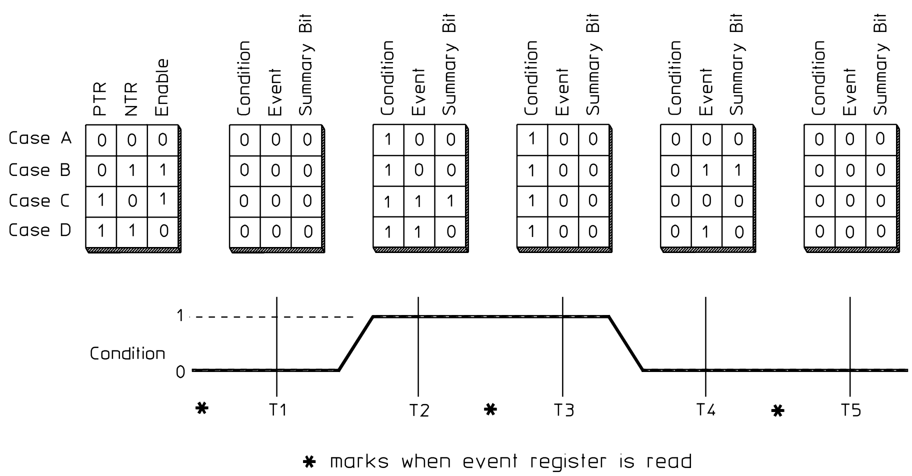


Figure 3. Example Sequence

In cases A and D, when the enable bit is zero, the summary bit cannot be set, regardless of the transition register settings, and no service request is made. When the enable bit is one, the summary bit is set when the transition occurs for which the corresponding transition register is configured to report.

How to Use Registers

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

- The polling method.
- The service request (SRQ) method.

In the polling 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.

The polling method works well if you do not need to know about changes the moment they occur. The SRQ method should be used if you must know immediately when a condition changes. To detect a change using the polling method, the program must repeatedly read the registers.

Use the SRQ method when:

- you need time-critical notification of changes
- you are monitoring more than one device which supports SRQs
- you need to do have the controller do something else while waiting
- you can’t afford the performance penalty inherent to polling

Use polling when:

- your language/development environment does not support SRQ interrupts
- you want to write a simple, single-purpose program and don’t want the added complexity of setting up an SRQ handler.

To monitor a condition:

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

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 GPIB commands to enable the bit that monitors the condition and to enable the summary bits that report the condition to the RQS bit.
4. Enable the controller to respond to service requests.

When the condition changes, the analyzer sets its RQS bit and the GPIB's SRQ line. The controller is informed of the change as soon as it occurs. The time the controller would otherwise have used to monitor the condition can now be used to perform other tasks. 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 4, other register sets in the analyzer report to the Status Byte. Most of them report directly, but three report indirectly—via the Questionable Status register set (see figure 5).

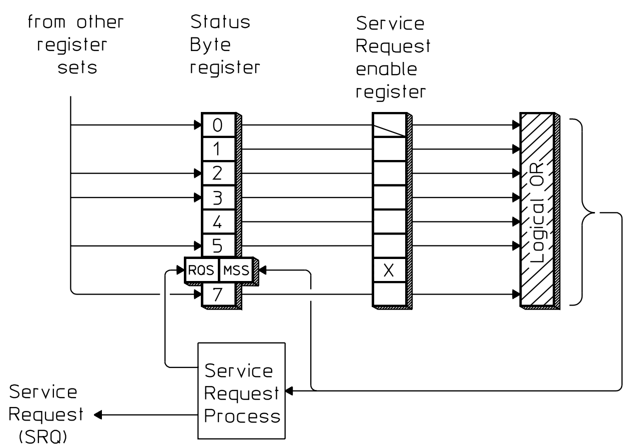


Figure 4. Generating a Service Request

Bit 6 of the Status Byte serves two functions; the request service function (RQS) and the master summary status function (MSS). The RQS bit is set whenever something (that it is configured to report) changes and is cleared when the status byte is read (with a serial poll). The MSS bit is set in the same way as the RQS bit, but it clears only when the condition that set it is cleared. The MSS bit is read with *STB?.

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 GPIB'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 analyzer 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 Agilent 89400-series analyzer, 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 GPIB's SRQ line is set true. Each device on the bus returns the contents of its Status Byte register in response to this poll. The device whose RQS bit is set to 1 is the device that requested service.

Notes

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.

Restarting a measurement (INIT) can cause the measuring bit to pulse low, which causes an SRQ if the status register is configured to SRQ on end-of-measurement. To avoid this:

1. INIT:CONT OFF
2. Set/enable status registers
3. Restart measurement (send INIT)

The Agilent 89400A's Register Sets

Register Summary

The Agilent 89400-Series analyzers use eight register sets to keep track of instrument status:

- Status Byte
- Device State
- Questionable Status
- Questionable Voltage
- Questionable Frequency
- Questionable Modulation
- Standard Event
- Operational Status
- User Status

The reporting structure is summarized in figure 5. They are described in detail in the following sections.

Note Register bits not explicitly presented in the following sections are not used by the Agilent 89400A Series. A query to one of these bits returns a value of 0.

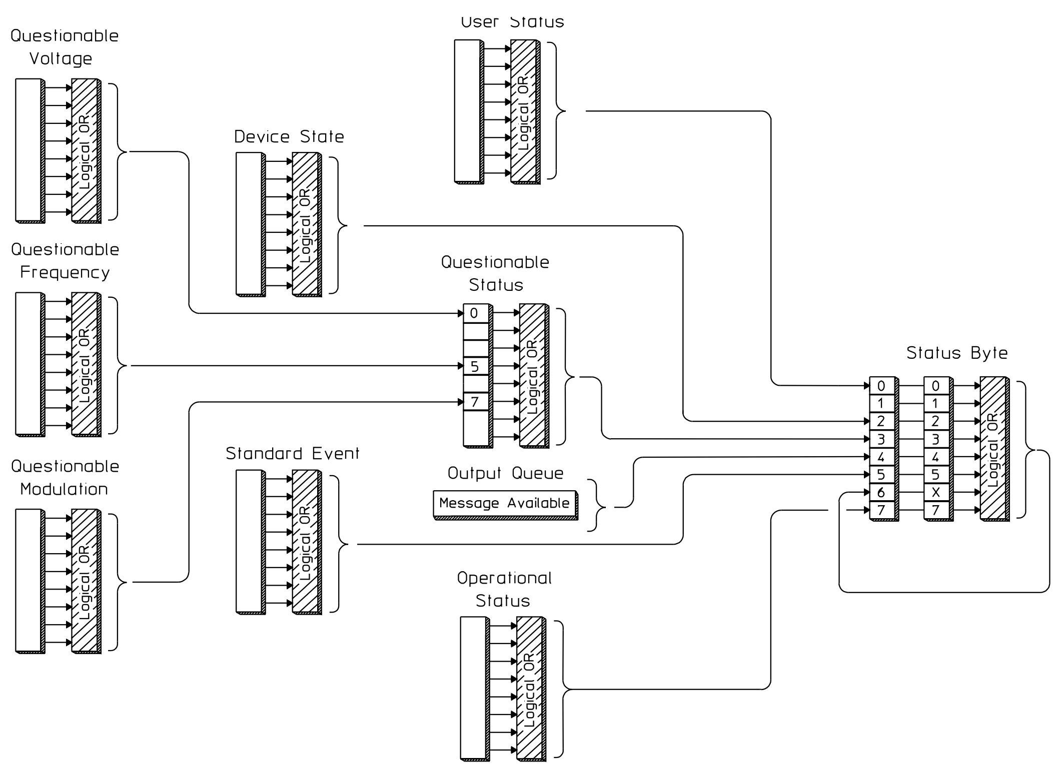


Figure 5. Agilent 89400 Series Register Set Overview

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.

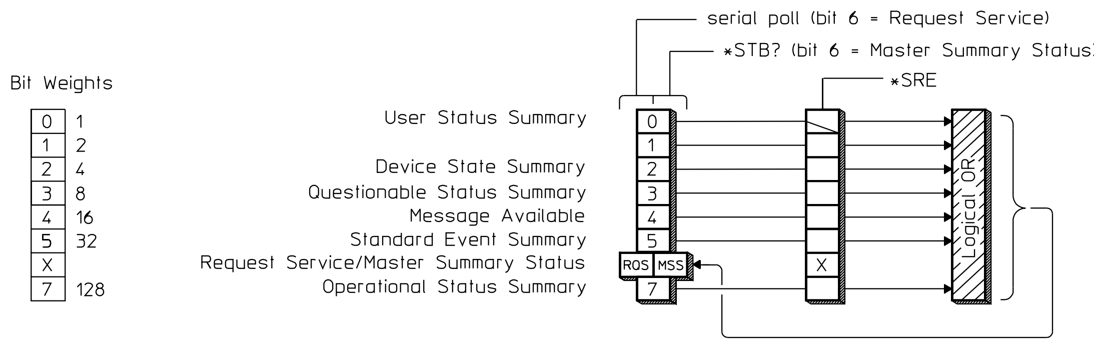


Figure 6. The Status Byte Register Set

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 6 also shows the commands you use to read and write the Status Byte registers. See the STATUS commands for more information about these commands.

Device State Register Set

The Device State register set monitors the states of eight device-specific parameters. See figure 7.

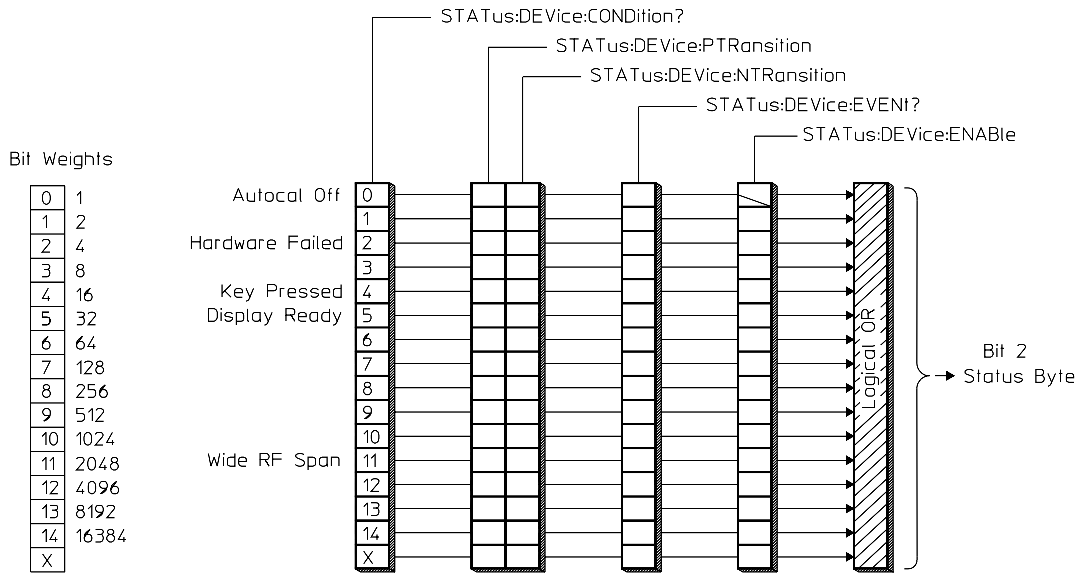


Figure 7. The Device State Register Set

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 always returns 0 for this bit.
- **Display Ready** (bit 5) is set to 1 when measurement results are available. This is an event. The condition register always returns 0 for this bit.

Figure 7 also shows the commands you use to read and write the Device State registers. See the STATUS commands 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 8.

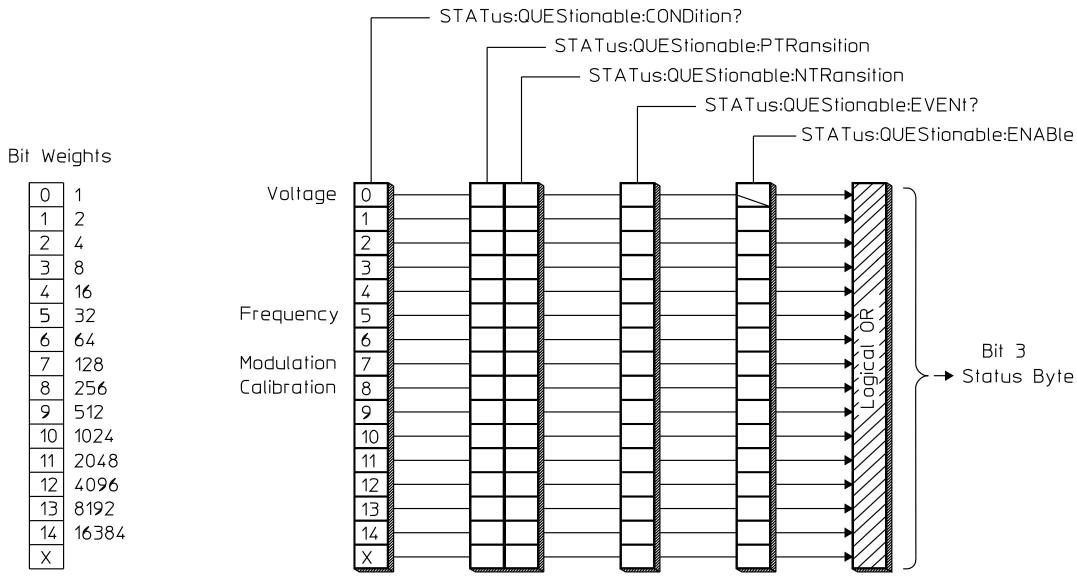


Figure 8. The Questionable Status Register Set

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.
- **Frequency** (bit 5) is set to 1 when one or more enabled bits in the Questionable Frequency Event register are set to 1.
- **Modulation** (bit 7) is set to 1 when one or more enabled bits in the Questionable Modulation Event Register are set to 1.
- **Calibration** (bit 8) is set to 1 when the self-calibration fails.

Figure 8 also shows the commands you use to read and write the Questionable Status registers. See the STATUS commands 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 9.

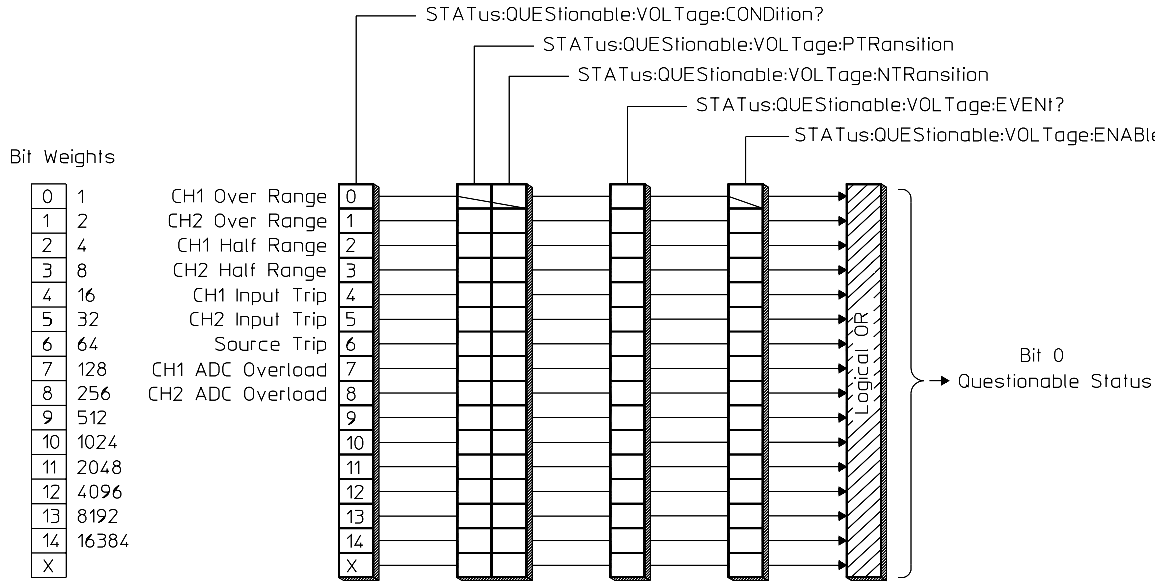


Figure 9. The Questionable Voltage Register Set

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

- **Channel 1 Over Range** (bit 0) is set to 1 when the channel-1 input signal exceeds the channel-1 input range setting.
- **Channel 2 Over Range** (bit 1) is set to 1 when the channel-2 input signal exceeds the channel-2 input range setting.
- **Channel 1 Half Range** (bit 2) is set to 1 when the channel-1 input signal is within approximately 6 dB of the channel-1 range setting.
- **Channel 2 Half Range** (bit 3) is set to 1 when the channel-2 input signal is within approximately 6 dB of the channel-2 range setting.
- **Channel 1 Input Trip** (bit 4) is set to 1 when the channel-1 input-protection relay trips (opens).
- **Channel 2 Input Trip** (bit 5) is set to 1 when the channel-2 input-protection relay trips (opens).
- **Source Trip** (bit 6) is set to 1 when the source-protection relay trips (opens).
- **CH1 ADC Overload** (bit 7) is set to 1 when the channel-1 input signal exceeds the ADC range.
- **CH2 ADC Overload** (bit 8) is set to 1 when the channel-2 input signal exceeds the ADC range.

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

Questionable Frequency Register Set

The Questionable Frequency register set monitors conditions that affect the frequency accuracy of measurement data. See figure 10.

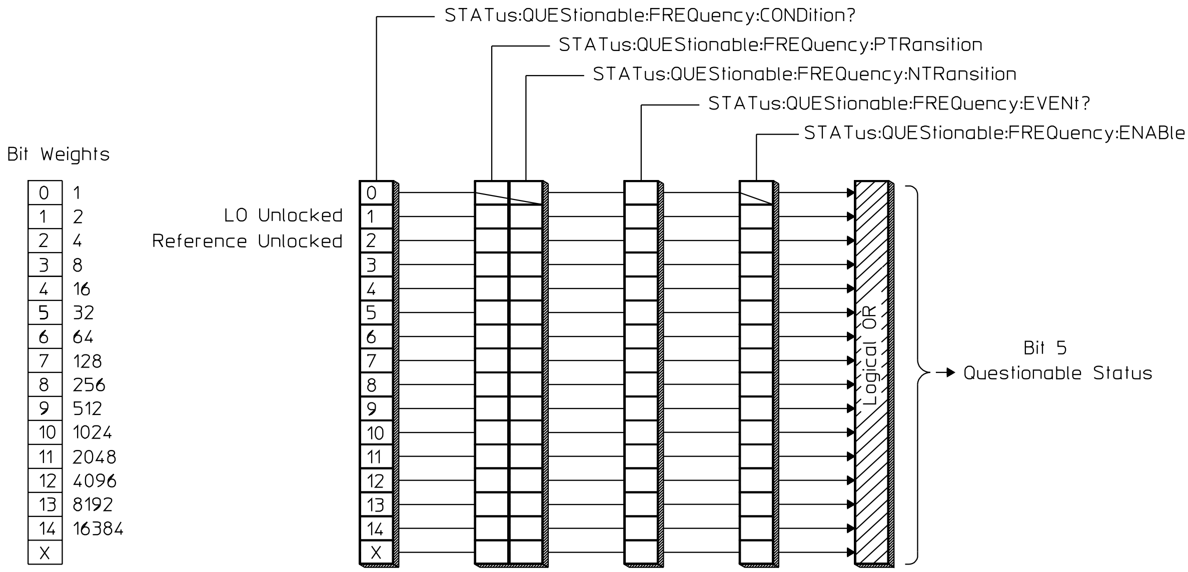


Figure 10. The Questionable Frequency Register Set

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

- **LO Unlocked** (bit 1) is set to 1 when the analyzer's local oscillator is not locked to its internal reference signal(s).
- **Reference Unlocked** (bit 2) is set to 1 when the analyzer's internal reference signal is not locked to the external reference signal being applied to the analyzer's rear panel.

Figure 10 also shows the commands you use to read and write the Questionable Frequency registers. See the STATUS commands for more information about these commands.

Questionable Modulation Register Set

The Questionable Modulation register set monitors conditions that affect the modulation accuracy of measurement data. These conditions occur only when option AYA is present. See figure 11.

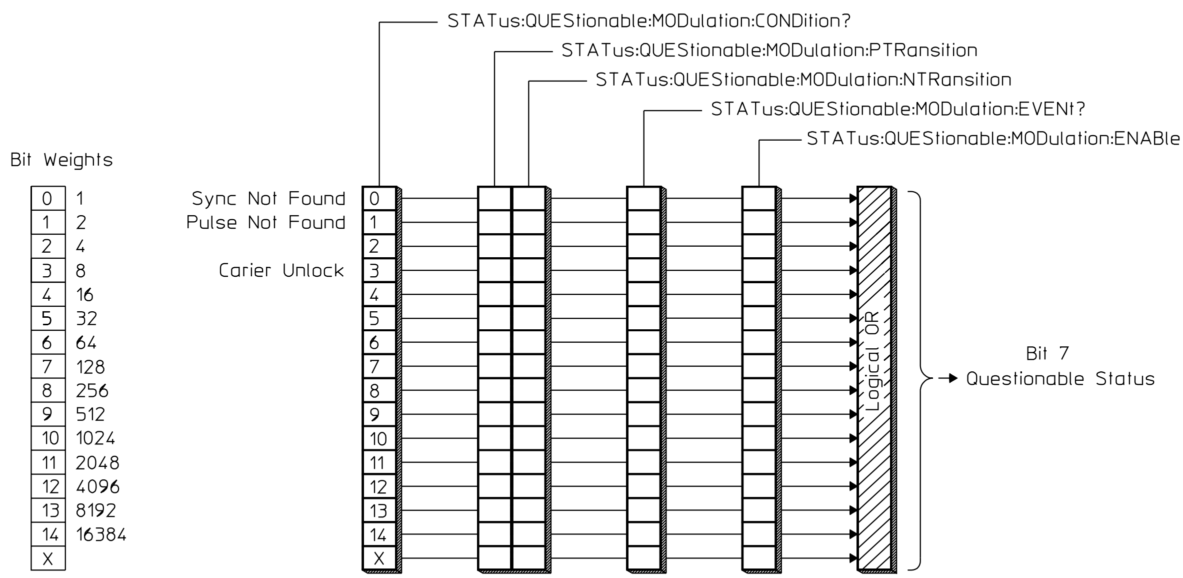


Figure 11. The Questionable Modulation Register Set

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

- **Sync Not Found** (bit 0) is set to 1 when the sync pattern cannot be found during a digital demodulation measurement in which sync search is enabled.
- **Pulse Not Found** (bit 1) is set to 1 when a pulse (an off-on-off transition) cannot be found within the specified *search length* during a digital demodulation measurement in which pulse search is enabled.
- **Carrier Unlock** (bit 3) is set to 1 if you use the VSB demodulation format and the analyzer cannot obtain carrier lock. VSB demodulation is only available with option AYH (Digital Video Modulation Analysis). For details about carrier lock with VSB measurements, see the ''Video Demodulation Concepts'' chapter in the *Operator's Guide*.

Figure 11 also shows the commands you use to read and write the Questionable Modulation registers. See the STATUS commands for more information about these commands.

Standard Event Register Set

The Standard Event register set monitors GPIB errors and synchronization conditions. See figure 12.

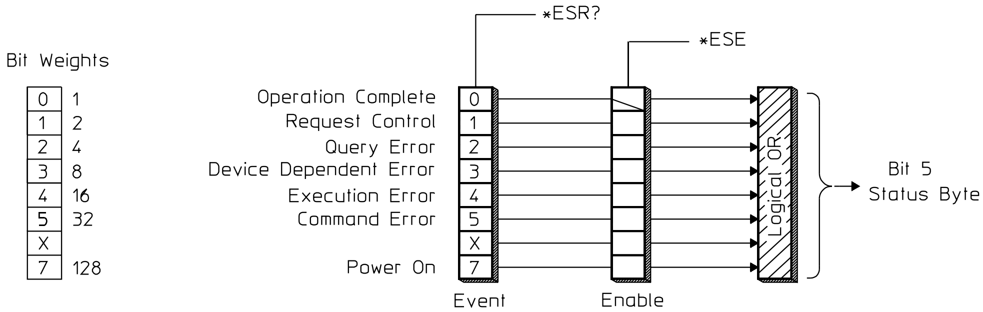


Figure 12. The Standard Event Register Set

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 events occur (in order listed):
  - You send the *OPC command to the analyzer.
  - The analyzer completes all pending overlapped commands (see "Synchronization" in the *GPIB Programming Guide*).
- **Request Control** (bit 1) is set to 1 when both of the following conditions are true:
  - The analyzer is configured as an addressable-only device (see "Controller Capabilities").
  - 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 12 also shows the commands you use to read and write the Standard Event registers. See the STATUS commands 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 Agilent Technologies Instrument BASIC program. See figure 13.

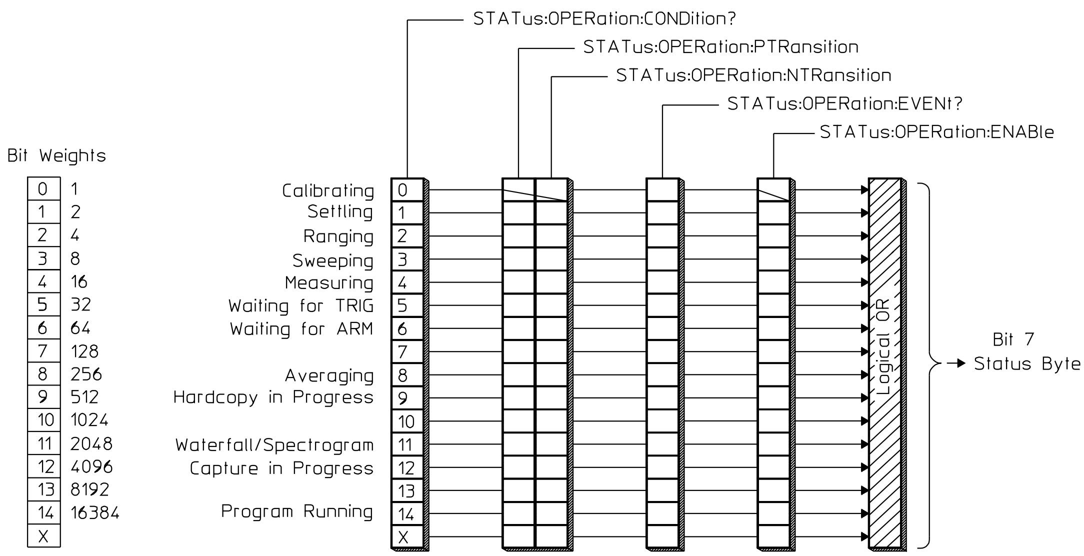


Figure 13. The Operational Status Register Set

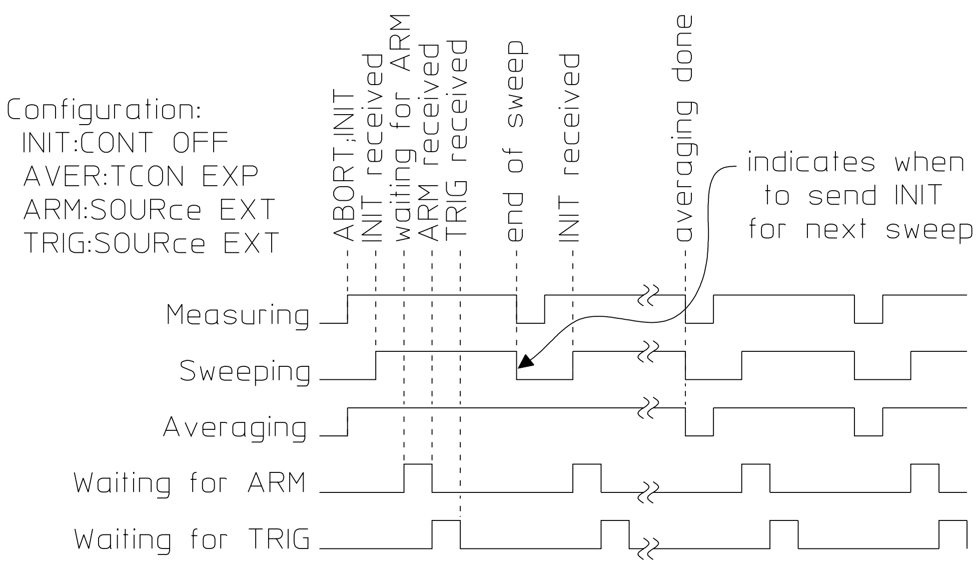
Bits in the Operational Status condition register are set to 1 under the following conditions:

- **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 an autorange is in progress.
- **Sweeping** (bit 3) has the same state as the Measuring bit when the sweep is Continuous. When the sweep is Single, the Sweeping bit is 0 until INIT is received and 1 while sweeping. The negative transition of this bit can be used to determine when INIT:IMM should be sent to initiate the next sweep. See following table and illustration.
- **Measuring** (bit 4) is set to 1 while the analyzer is actively measuring. The instrument is considered to be "measuring" following an ABOR command or another command which forces an internal abort such as a configuration change in time data or zoom/baseband status. So the measuring bit is 1 while waiting for an arm and while waiting for trigger. In single-sweep it is also 1 while waiting for INT:IMM. It goes low briefly at the end of each scan.
- **Waiting for TRIG** (bit 5) is set to 1 when the analyzer is ready for 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:
  - External arm is on (ARM:SOURce EXTernal).
  - The analyzer is ready to be armed.
- **Averaging** (bit 8) is set to 1 while the analyzer is averaging measurement data. For exponential and continuous averaging, it stays set to 1 until the specified number of averages is taken, and then it has the same state as the Measuring bit. See following illustration.
- **Hardcopy In Progress** (bit 9) is set to 1 while the analyzer is printing or plotting.

- **Waterfall/Spectrogram** (bit 11) is set to 1 when a waterfall or spectrogram is in progress on any trace measurement. It is set to 0 when all waterfalls and spectrograms are complete and then it has the same state as the Measuring bit. ''Complete'' means that the specified buffer depth number of scans have been acquired after the settling is done. ''Settling'' means that if averaging is exponential, the specified number of averages have been acquired and if the analyzer is operating real-time, the first time record is full.
- **Capture in Progress** (bit 12) is set to 1 when a time capture begins. It is set to 0 when the capture either comes to completion or is aborted.
- **Program Running** (bit 14) is set to 1 while the current Agilent Technologies Instrument BASIC program runs.

Bit States versus Instrument Condition

condition	Measuring bit	Sweeping bit
paused	0	0
avg done	0	0
waiting for INIT	1	0
settling	1	1
waiting for ARM	1	1
waiting for TRIG	1	1
meas running	1	1



Timing diagram for single-sweep, exp averaged, externally armed, triggered measurement. Notice that the averaging bit follows the measurement bit after the specified number of averages have been acquired.

Figure 12 also shows the commands you use to read and write the Operational Status registers. See the STATUS commands for more information about these commands.

User Status Register Set

The User Status register set detects STATus:USER:PULSe commands. See figure 14.

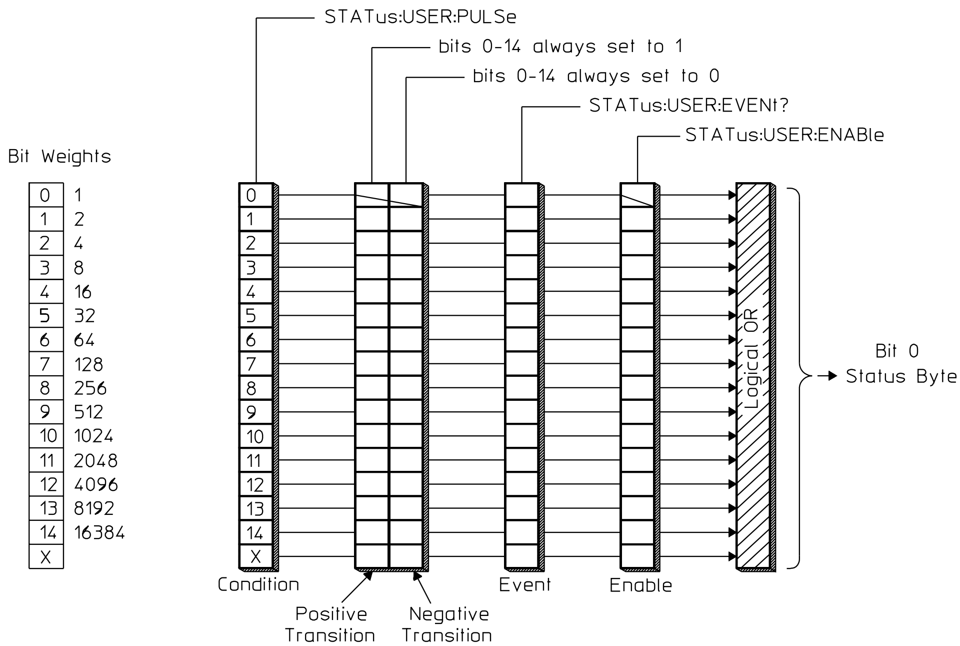


Figure 14. The User Status Register Set

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 14 also shows the commands you use to read or write the User Status registers. See the STATUS commands for more information about these commands.

Agilent 89400 Series Register Set Summary

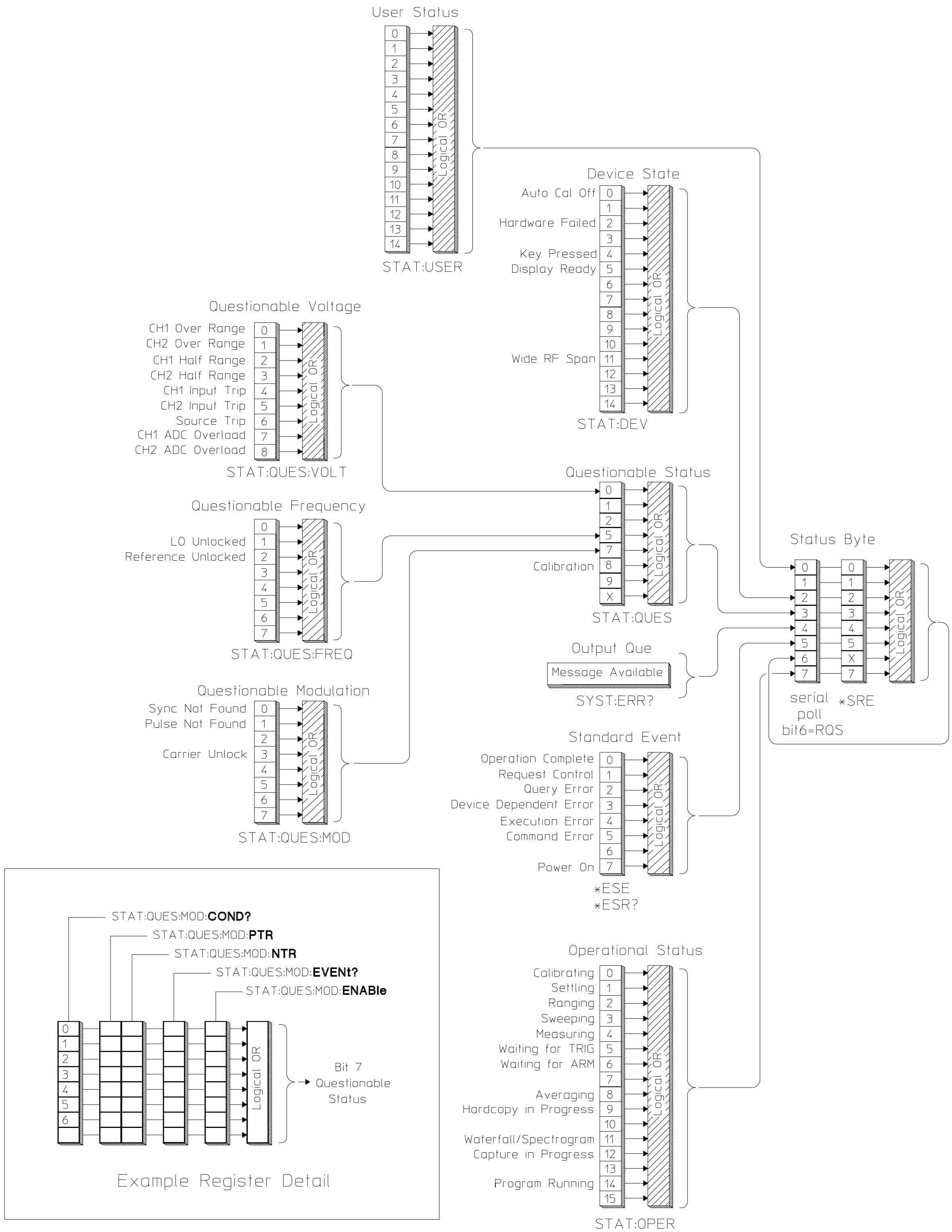


Figure 15. Status Register Quick Reference

2

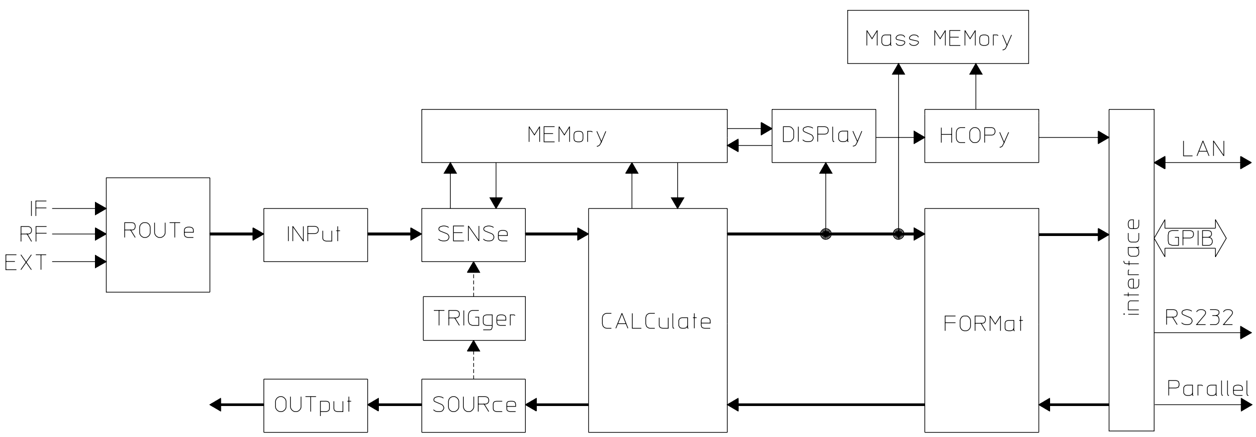
The SCPI Instrument Model

Introduction

A model is used within SCPI as a means of achieving compatibility. SCPI concerns itself with three types of compatibility. The first form of compatibility is called vertical compatibility. Vertical compatibility is where two instruments of the same type have identical controls.

The second form of compatibility is called horizontal compatibility. Horizontal compatibility is where two instruments can make the same measurement, regardless of the actual measurement techniques used. To be horizontally compatibile, both instruments would use the same commands to make this measurement. For example, both an oscilloscope and a counter can perform a risetime measurement on a pulse. The two instruments are said to be horizontally compatible if the same command is used in both instruments.

The third form of compatibility is called functional compatibility. Functional compatibility is where two instruments which perform the same function do so with the same commands. For example, a spectrum analyzer and an rf source may both sweep in frequency. If the same frequency and sweep commands are used in both instruments, they would be functionally compatible in this area.



The figure above represents the way in which instrument functionality is viewed and categorized by SCPI. The purpose of this categorization is to provide organization and consistancy between the various commands available in SCPI for all the different types of instrumentation. The model defines where elements of the language must be assigned in the SCPI hierarchy. Major areas of signal functionality are shown broken into blocks; each of these blocks are major command subtrees in SCPI. For simplicity, all connections between blocks are not shown. In this chapter each subtree is discussed.

ROUTe

The purpose of the signal routing block is to control the routing of signals between an instrument's signal ports and its internal signal functionality. Signal routing also controls the connection from signal port to signal port, where such capability exists. The commands which control this block are described in the SCPI tree under the ROUTe subsystem.

INPut

The purpose of the INPut block is to condition the incoming signal before it is converted into data by the SENSe block. INPut block functions include filtering, biasing, frequency conversion (such as a mixer or prescaler function), and attenuation.

SENSe

The purpose of the SENSe block is to convert signal(s) into internal data that can be manipulated by normal computer techniques. The commands associated with the SENSe block control the various characteristics of the conversion process. Examples are range, resolution, bandwidth, and gate time. This block does not include any mathematical manipulation of the data after it has been converted.

CALCulate

In the measurement function path, the purpose of the CALCulate block is to convert sensed data into a form more useful to the application. Typical calculations include converting units and postprocessing calculations.

OUTPut

The purpose of the OUTPut block is to condition the outgoing signal after it has been generated. The OUTPut block functions include filtering, biasing, frequency conversion (such as a mixer function), and attenuation.

SOURce

The purpose of the SOURce block is to generate a signal based on specified characteristics and/or supplied data. The commands associated with this block describe the characteristics of the generated signal.

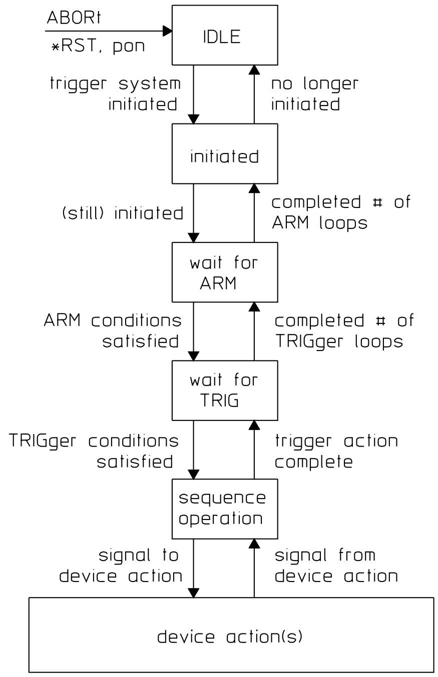
TRIGger

The trigger subsystem is used to synchronize device action(s) with events. A device action might be the acquisition of a measurement or the application of a stimulus. The trigger subsystem consists of the expanded capability model which is capable of describing very complex device trigger systems. It also makes provision, through the ARM-TRIGger model, for simple descriptions of less complicated trigger systems. These two models are consistent and compatible with each other. The ARM-TRIGger model represents a subset of the capability available in the expanded capability model.

Special terms in the following discussion are defined as follows. A box in a flow chart diagram identifies a state of a transition diagram and is referred to as a layer. A sequence is a set of vertically connected layers. A solid line defines flow of control between states and a dashed line defines signals used as semaphores.

ARM-TRIGger Model

The ARM-TRIGger model represents a level of capability that is often found in a device and is shown in the following figure. This model has two independent levels of event detection, one in each of the ARM and TRIGger layers.

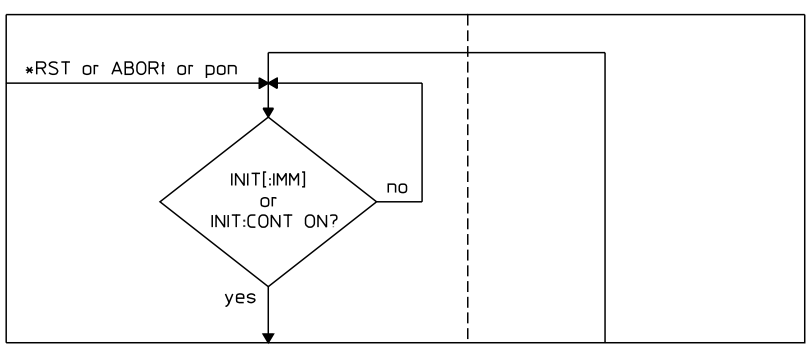


Model Layers

The following figures detail each of the various types of layers that may exist in a sequence. Each layer is shown with a dotted line, used to divide the upward and downward traverses through the layer. The different traverses are reference in describing the operation of each layer.

IDLE State

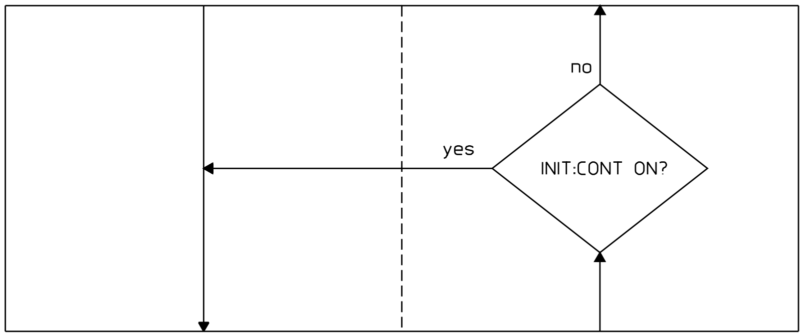
On receipt of either *RST or ABORt, the trigger subsystem shall enter the IDLE state. Receiving the IEEE 488.2 dcas message may also cause a transition to IDLE. Devices which cannot process commands when not in IDLE must enter IDLE when dcas is received to meet the IEEE 488.2 requirements for device clear. The downward traverse from IDLE state, ''trigger system initiated,'' is affected by either the INITiate[:IMMediate] command or by setting INITiate:CONTinuous to ON.



Exiting the IDLE stat causes the pending operation flag associated with the initiated action to be set true. Entering the IDLE state causes the pending operation flag to be set false.

Initiated

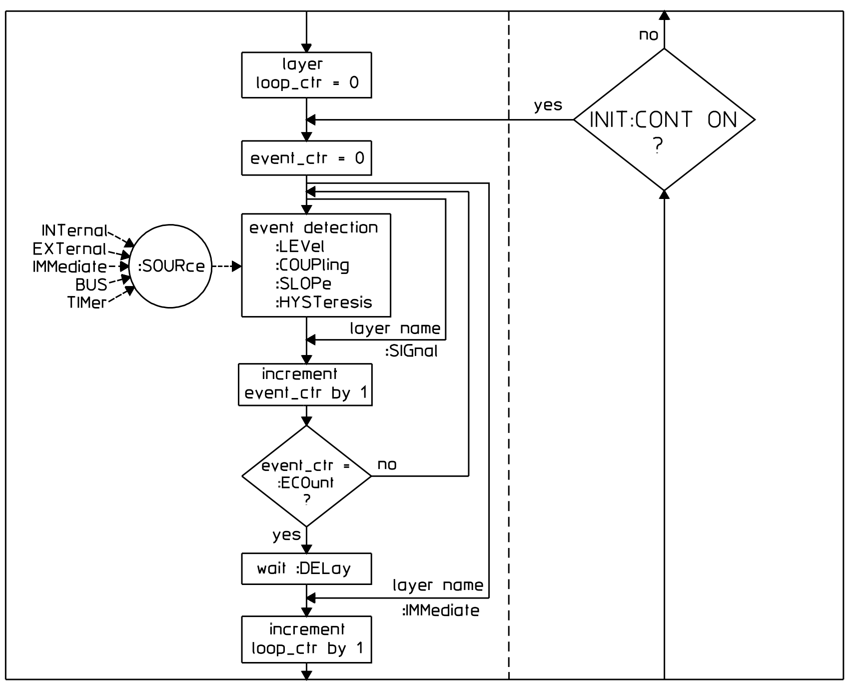
Once the trigger system is initiated from the IDLE state, it passes through the initiated state, immediately making the downward traverse to the ''wait for ARM'' state. The upward traverse is dependent on the setting of CONTinuous. If CONTinuous is set to OFF, the upward trasition to IDLE is made. Otherwise, the downward traverse to the ''wait for ARM'' state is made, and the trigger sequence remains initiated.



The initiated state avoids having the pending operation flag changes states every time the entire trigger sequence is completed when CONTinuous is ON.

Event Detection Layer

The ARM and TRIGger layers are both event detection layers with the TRIGger layer being subservient to the ARM layer. Each layer provides one level of event detection.



The downward traverse through an event detection layer depends on the sourced event being detected the specified number of times. Typically, the first event detected after entering the event detection layer is all that is required to proceed. However, a particular number of occurances of the same event may be specified with ECOunt; for example, wait for the tenth positive edge of a signal. the downward traverse is also subject to a time delay if one is specified.

Two commands override a normal downward traverse. The layer name followed by :IMMediate causes the event detection and the subsequent delay to be bypassed. The layer name followed by :SIGNal causes the device to proceed as though a single event had occured.

The upward traverse is dependent on the value of COUNt for the given layer. When COUNt is greater than one, all of the subservient layers are cycled repeatedly COUNt times, For example, to make five measurements, each qualified by the same combination of ARM and TRIGger events, the COUNt in the ARM layer should be set to 5. Each time the ARM layer is entered from below, the flow is redirected to follow the downward traverse. After the fifth cycle, when COUNt is satisfied, the upward traverse to the initated state is made.

MEMory

The purpose of the MEMory block is to hold data inside the instrument. The memory may be implicit and inaccessible to the user (as internal calibration data, for example), may be fixed (as current measurement data, for example), or may be allocated and user-addressed.

FORMat

The purpose of the FORMat block is to convert between data representations, especially on the data that is transferred over the external interface. An example is conversion of internal data formats to ASCII.

Hard COPy

The purpose of the HCOPy block is to control the setup of plotting and printing to an external device. The Hard COPy subsystem does not perform any data formatting such as converting data from one representation to another (e.g. COMPlex to POLar). Instead, HCOPy adds the necessary page formatting (dependent upon the hard copy device language) to turn the data into an acceptable form for the hard copy device.

DISPlay

The purpose of the DISPlay block is to format data (usually generated by the CALC block) for presentation on a visual device such as a CRT. The commands associated with this block control aspects such as scaling, color, layout, and user annotation.

Mass MEMory

The purpose of the MMEM block is to provide access to mass storage devices such as internal RAM disks or internal or external hard disks or floppy disks. The commands in this block allow file operations such as saving traces, recalling states, copying files, and formatting disk volumes.

3

Command Reference

*CAL? query

Calibrates the analyzer and returns the (pass/fail) result.

Query Syntax: `*CAL?`

Example Statements:
OUTPUT 719;"*CAL?"
OUTPUT 719;"*cal?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

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

See also, the calibration commands.

*CLS command

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

Command Syntax: `*CLS`

Example Statements:

```
OUTPUT 719;"*Cls"
OUTPUT 719;"*CLS"
```

Attribute Summary:

```
Synchronization Required:  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.
- Questionable Frequency.
- Limit Fail.
- Questionable Status.
- Operation Status.

In addition, *CLS cancels any preceding *OPC command or query. This ensures that bit 0 of the Standard Event register is not set to 1 and that a response is not 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.

*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 719;"*ese 0"
OUTPUT 719;"*Ese 30"

Query Syntax: `*ESE?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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:

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

Acceptable values for bit_number are 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.

*ESR? query

Reads and clears the Standard Event Enable register.

Query Syntax: `*ESR?`

Example Statements:
```
OUTPUT 719;"*esr?"
OUTPUT 719;"*Esr?"
```

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  undefined value
SCPI Compliance:  confirmed
```

Description:

This 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. The decimal weight for each bit is assigned according to the following formula:

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

Acceptable values for bit_number are 0 through 7.

The register is cleared after being read by this query.

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.

*IDN? query

Returns a comma-separated list of arbitrary ascii response data items that uniquely identifies the analyzer.

Query Syntax: `*IDN?`

Example Statements:
OUTPUT 719;"*IDN?"
OUTPUT 719;"*idn?"

Return Format: `ARB_ASCII`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  instrument-specific
SCPI Compliance:  confirmed
```

Description:

The response to this query is in the form:

<manufacturer name>,<model number>,<serial number>,<firmware version>

Here is an example: HEWLETT-PACKARD,89410A,3046A00132,A.00.01

The response to this query uniquely identifies your analyzer.

*OPC **command/query**

Specifies or queries completion of all pending overlapped commands.

Command Syntax: `*OPC`

Example Statements:
```
OUTPUT 719;"*Opc"
OUTPUT 719;"*OPC"
```

Query Syntax: `*OPC?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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. See ''Synchronization'' in the *Programmer's Guide*.

When *OPC? returns true, given measurement configuration and measurement-initiating command.

<table>
<tr><th rowspan="2">Sweep</th><th rowspan="2">Waterfall/
Spectrogram</th><th rowspan="2">Command</th><th rowspan="2">AVG off</th><th colspan="2">Averaging is RMS or TIME</th><th colspan="2">Averaging is CONT or EXP</th></tr>
<tr><th>FAST AVG off</th><th>FAST AVG on</th><th>FAST AVG off</th><th>FAST AVG on</th></tr>
<tr><td rowspan="6">Continuous</td><td rowspan="3">Off</td><td>Start (ABOR)</td><td>first scan is complete</td><td colspan="2" rowspan="3">Current AVG = Start AVG + NUM AVGs</td><td colspan="2">Current AVG = NUM AVGS</td></tr>
<tr><td>CONT
(from paused)</td><td>first scan is complete</td><td colspan="2">Current AVG = NUM AVGS
or
first displayed scan after
Current AVG = NUM AVGS</td></tr>
<tr><td>CONT (from AVG complete)</td><td>*</td><td>*</td><td>*</td></tr>
<tr><td rowspan="3">On</td><td>Start (ABOR)</td><td>Waterfall/
Spectrogram full</td><td colspan="2" rowspan="3">Current AVG = Start AVG + NUM AVGs</td><td colspan="2">Waterfall/Spectrogram full</td></tr>
<tr><td>CONT
(from paused)</td><td>Waterfall/
Spectrogram full</td><td colspan="2">Waterfall/Spectrogram full</td></tr>
<tr><td>CONT(from AVG complete)</td><td>*</td><td>*</td><td>*</td></tr>
<tr><td>Single</td><td>(either)</td><td>INIT
(INIT:IMM)</td><td>first scan is complete</td><td>first scan is complete</td><td>first scan is complete</td><td>first scan is complete</td><td>first scan is complete</td></tr>
</table>

* impossible combination

- *Waterfall/Spectrogram full:* A waterfall/spectrogram (buffer) is full when buffer-depth number of scans have been added to it, which begins when the first scan is complete.
- The *first scan is complete* when the following conditions are met:
  - If the operation is in real time, the first scan is added when a complete time record has been acquired. (Traces displayed before this are of partial time records.)
  - If exponential averaging is on, the first scan is added when the specified number of averages have been taken.

*OPT? query

Returns a comma-separated list as a string that identifies the analyzer's option configuration.

Query Syntax: `*OPT?`

Example Statements:
OUTPUT 719;"*opt?"
OUTPUT 719;"*Opt?"

Return Format: `STRING`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The response to this query identifies the analyzer's option configuration. For example, if your analyzer has the second input (receiver) channel installed, it returns AY7 to this query. Options are identified by the following:

- AY7 Second (10 MHz) input channel
- 1C2 IBASIC
- AY9 1 Msample memory for time capture
- AYA Vector modulation analysis (IQ demod)
- AYB Waterfall and spectrogram displays
- AYH Digital Video Modulation Analysis. AYA required.
- UTH Add 20 MBytes Memory and basic LAN
- UG7 Advanced LAN: X11 and FTP.
- B73 Digital Wideband CDMA Analysis
- B79 Digital ARIB 1.0-1.2 Wideband CDMA Analysis
- B7A Enhanced Data rates for GSM Evolution (EDGE)

For the 89410A:

- AY5 Precision frequency reference (oven)

For the 89441A:

- AY8 Internal RF source
- AY4 Reported if crystal oven is absent from down converter section (lower cabinet)

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 <number>[, <number>]`

<number> ::= a real number (NRf data)
limits: 0:30

Example Statements:

```
OUTPUT 719;"*PCB 19"
OUTPUT 719;"*pcb 19, 1"
```

Attribute Summary:

```
Synchronization Required:  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 GPIB 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 interpreted as the secondary address of the controller.

The address last specified with this command is saved in nonvolatile memory, so it is unaffected when you turn the analyzer off and on. It is also unaffected by the *RST command.

*PSC **command/query**

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

Command Syntax: `*PSC <number>`

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

Example Statements:
OUTPUT 719;"*psc 0"
OUTPUT 719;"*PSC 1"

Query Syntax: `*PSC?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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.

Sending *PSC 0 sets the power-on-status-clear flag false which allows instruments to assert SRQ after power-on. Use of any value other than 0 sets the power-on-status-clear flag to true which Enables the power-on clear status and disallows any SRQ assertion after power-on.

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 restored when you turn the analyzer on, but only if the Power-on Status Clear (PSC) flag is set to 0. When the PSC flag is set to 1, the two enable registers are cleared at power-up.

The number last specified with *PSC is saved in nonvolatile memory, so it is unaffected when you turn the analyzer off and back on. It is also unaffected by the *RST command.

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

The query returns the current state of the PSC flag (0|1).

*RST command

Executes a device reset.

Command Syntax: `*RST`

Example Statements:
OUTPUT 719;"*RST"
OUTPUT 719;"*rst"

Attribute Summary:
```
Synchronization Required:  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 commands or query.

If a file named AUTO_ST containing an instrument state exists on any drive, that state is loaded and becomes active at power-on.

The reset state is similar to the preset state. The preset state of each command is listed in the Attribute Summary for that command. In some cases, a command's reset state differs from its preset state (SYST:PRES). These commands (and their reset states) are:

- CALibration:AUTO is set to 0 (off).
- INIT:CONT is set to OFF.

Note This command is *not* equivalent to a front-panel preset (SYST:PRES).

The following are *not* affected by the *RST command (or SYST:PRES):

- The state of the Power-on Status Clear flag.
- The state of all enabled and transition registers.
- The GPIB input and output queues.
- The time and date (SYST:TIME and SYST:DATE).
- The GPIB address settings.
  - SYST:COMM:GPIB:ADDR
  - HCOP:PLOT:ADDR
  - HCOP:PRIN:ADDR
- The memory configuration.
- The GPIB controller capability setting.
- The default disk selection (MMEM:MSIS).
- External disk address.
- External receiver address.
- Contents of data registers.
- Contents of math function and constant registers.
- Contents of the RAM and NVRAM disks.
- Calibration constants.
- LAN configuration.

*SRE command/query

Sets bits in the Service Request Enable register (masks status byte).

Command Syntax: `*SRE <number>`

<number> ::= a real number (NRf data)
limits: 0:255

Example Statements:

```
OUTPUT 719;"*sre 0"
OUTPUT 719;"*SRE 5"
```

Query Syntax: `*SRE?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  dependent on setting of *PSC
SCPI Compliance:  confirmed
```

Description:

Use this command to set the Service Request Enable register to mask, or select, the events which cause a service request (SRQ). See the section Status Byte Register Set in "Using Status Registers."

Assign a decimal weight to each bit you want set (to 1) according to the following formula:

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

Acceptable values for bit_number are 0 through 7. Add the weights and then send the sum with this command. Sending *SRE 0 disables SRQ; *SRE 5 enables bits 0 and 2, User Status Summary and Device State Summary.

Note The analyzer ignores the setting specified for bit 6 of the Service Request Enable register. 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 719;"*Stb?"
OUTPUT 719;"*STB?"
```

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  IEEE 488.2
```

Description:

This command allows you to read 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:

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

Acceptable values for bit_number are 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 1.

*TRG **command**

Triggers the analyzer when TRIG:SOUR is BUS.

Command Syntax: `*TRG`

Example Statements:
OUTPUT 719;"*trg"
OUTPUT 719;"*Trg"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

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

- The GPIB is designated as the trigger source (send the TRIG:SOUR BUS command.)
- The analyzer is waiting to trigger. (Bit 5 of the Operation Status register, waiting for trigger, must be set).

*TST? query

Tests the analyzer hardware and returns the result.

Query Syntax: `*TST?`

Example Statements:

```
OUTPUT 719;"*TST?"
OUTPUT 719;"*tst?"
```

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  IEEE 488.2
```

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 719;"*Wai"
OUTPUT 719;"*WAI"
```

Attribute Summary:

```
Synchronization Required:  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. See also, *OPC.

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.

ABORt **command**

Abort the current measurement in progress.

Command Syntax: `ABORt`

Example Statements:

```
OUTPUT 719;"abor"
OUTPUT 719;"Abort"
```

Attribute Summary:

```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command aborts any measurement in progress and resets the trigger system. Any actions related to the trigger system that are in progress, such as a sweep or taking a measurement are aborted and the NPO (no pending operations) flag is set false.

- Sets the Measuring and Averaging bits in the Operational Status register to 0.
- ABOR;INIT:IMM serves a special synchronization function.
- If INIT:CONT is OFF, then INIT:IMM starts a new measurement.
- If INIT CONT is ON, a new measurement begins immediately.

The ABORt command is an important part of synchronizing measurements. To assure that the measured data is the result of the current configuration, use one of the methods described in Chapter 3 of the *GPIB Programmer's Guide*.

ARM:DELay command/query

Specifies the delay after arming occurs before triggering may occur.

Command Syntax: `ARM:DELay <number>[<unit>]|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:41.94 S in 2.5 us steps

<unit> ::= S

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;":ARM:DEL 2"
OUTPUT 719;"arm:delay 220E-3"
```

Query Syntax: `ARM:DELay?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0 seconds
SCPI Compliance:  confirmed
```

Description:

If ARM:SOUR is EXT, the analyzer must be armed before triggering can occur. ARM is either EXTernal or IMMediate (external arm on/off). External arm is activated with the ARM:SOUR EXT command. See TRIG:SOUR for more information.

This command has no effect if ARM:SOUR is IMM (external arm off).

ARM:LEVel **command/query**

Specifies the threshold value for external arm.

Command Syntax: `ARM:LEVel <number>[<unit>]|<step>|<bound>`

<number>	::=	a real number (NRf data) limits: -11:11
<unit>	::=	V
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"Arm:Level -3"
OUTPUT 719;"ARM:LEV 4V"
```

Query Syntax: `ARM:LEVel?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0V
SCPI Compliance:  confirmed
```

Description:

If ARM:SOUR is EXT, the analyzer must be armed before triggering can occur. ARM is either EXTernal or IMMediate (external arm on/off). External arm is activated with the ARM:SOUR EXT command. See TRIG:SOUR for more information.

This command has no effect if ARM:SOUR is IMM (external arm off).

ARM:REGion command/query

When ARM:SOUR is EXT, this command specifies whether arming occurs when the external arm signal is positive-with-respect-to or negative-with-respect-to the specified trigger level.

Command Syntax: `ARM:REGion ABOVe|BELow`

Example Statements:
```
OUTPUT 719;":ARM:REGION ABOVE"
OUTPUT 719;"arm:reg BELOW"
```

Query Syntax: `ARM:REGion?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  ABOV
SCPI Compliance:  instrument-specific
```

Description:

If ARM:SOUR is EXT, the analyzer must be armed before triggering can occur. ARM is either EXTernal or IMMediate (external arm on/off). External arm is activated with the ARM:SOUR EXT command. See TRIG:SOUR for more information.

This command has no effect if ARM:SOUR is IMM (external arm off).

ARM:SOURce — command/query

Selects either external arming or immediate (continuous) arming.

Command Syntax: `ARM:SOURce IMMediate|EXTernal`

Example Statements:
```
OUTPUT 719;"Arm:Sour IMMEDIATE"
OUTPUT 719;"ARM:SOUR IMMEDIATE"
```

Query Syntax: `ARM:SOURce?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  IMM
SCPI Compliance:  confirmed
```

Description:

Immediate triggering corresponds to turning external arm off. In this state the analyzer operates as usual–it makes a measurement when the trigger conditions are met.

External triggering corresponds to turning external arm on. In this state the analyzer makes a measurement when these conditions are met, in this order:

1. The analyzer receives a valid ''external arm'' signal.
2. Trigger conditions are met.

The external-arm signal is valid when it meets the conditions set with the ARM's LEVEL, REGION, and DELAY commands. The external-arm signal is input via the EXT ARM (BNC) connector on the analyzer's rear panel.

When the external-arm signal is valid (as defined by the level and region), it arms the analyzer. Once armed, the analyzer remains armed regardless of the value of the EXT ARM signal. If the analyzer receives multiple valid external-arm signals, the first valid signal arms the analyzer; the others do nothing.

CALCulate[1|2|3|4]:CCDF:COUNt? query

Returns the current number of data samples in the CCDF measurement.

Query Syntax: `CALCulate[1|2|3|4]:CCDF:COUNt?`

Example Statements:

OUTPUT 719;":calc4:ccdf:coun?"
OUTPUT 719;"Calculate4:Ccdf:Coun?"

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

CALCulate[1|2|3|4]:CCDF:POWer? query

Returns the signal average power used to compute the CCDF measurement.

Query Syntax: `CALCulate[1|2|3|4]:CCDF:POWer?`

Example Statements:

```
OUTPUT 719;"CALCULATE3:CCDF:POWER?"
OUTPUT 719;"calc:ccdf:power?"
```

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The analyzer plots the CCDF (Complementary Cumulative Density Function) using units of percent (%) for the y-axis (using a logarithmic scale) and power (dB) for the x-axis. Power on the x-axis is relative to the signal average power in units of dBm.

This command returns the signal average power when CCDF measurement data is in the active trace.

For details about CCDF measurements, see the *Operator's Guide* and see online help for the [CCDF] softkey.

CALCulate[1|2|3|4]:DATA? query

Returns trace data that's transformed to current coordinate system (CALC:FORM).

Query Syntax: `CALCulate[1|2|3|4]:DATA?`

Example Statements:
OUTPUT 719;":Calc:Data?"
OUTPUT 719;"CALC:DATA?"

Return Format: `DEF_USER`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This query returns a block of coordinate transformed trace data for the trace specified with a number 1 through 4. The block is returned as a series of Y-axis values. The units for these values is determined with the CALC:UNIT:<unit> queries. Units are the same as the reference level units. The X-axis value for a given point is implied by the order of the points.

For more information on definite-length blocks (DEF_BLOCK), see "Block Parameters" in chapter 4 of the *Programmer's Guide* or refer to the example in the documentation for TRAC:DATA.

Note CALC:DATA cannot be used to transfer trace data *to* the analyzer. (see TRAC:DATA)

Related Commands:

- To load a trace into the analyzer from a mass-storage device, use MMEM:LOAD:TRAC.
- To store trace data in a data register, use TRACE[:DATA].
- To specify whether the data transferred is in ASCII or REAL format, use FORMAT[:DATA].
- To determine how many points are in the trace, use CALC:DATA:HEAD:POIN?.

CALCulate[1|2|3|4]:DATA:HEADer:POINts? query

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

Query Syntax: `CALCulate[1|2|3|4]:DATA:HEADer:POINts?`

Example Statements:
OUTPUT 719;"calculate:data:header:poin?"
OUTPUT 719;"Calc:Data:Head:Points?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The display’s X axis is divided into discrete points. The CALC:DATA:HEAD:POIN query returns the number of points on the X axis.

CALCulate[1|2|3|4]:FEED **command/query**

Selects the measurement data to be displayed.

Command Syntax: CALCulate[1|2|3|4]:FEED <string>

<string> ::= (see table)

Example Statements: OUTPUT 719;":CALCULATE4:FEED ‘XFR:POW 1’"
OUTPUT 719;"calc2:feed ‘XTIM:VOLT 2’"

Query Syntax: CALCulate[1|2|3|4]:FEED?

Return Format: STRING

Attribute Summary: Synchronization Required: no
Preset State: dependent on options installed
SCPI Compliance: confirmed

Description:

Selects measurement results for a trace. The numbers 1, 2, 3, 4 in the CALCulate node specify the trace (A, B, C, or D). “CALC1" (trace A) is assumed if the number is omitted.

Special Considerations:

- The measurement data depends on the instrument mode, as shown in the following table.
- The number at the end of the string specifies the input channel (except for data registers); ‘‘1” is used in the table but ‘‘2” may be used if second channel is installed.
- Math functions are defined and selected with the CALC:MATH commands.

Strings Used to Specify Measurement Types Given Instrument Mode

Standard Meas Types	Scalar Mode	Vector Mode	Analog Demod Mode[4]
spectrum	XFR:POW 1	XFR:POW 1	XFR:AM 1
PSD	XFR:POW:PSD 1	XFR:POW:PSD 1	XFR:AM:PSD 1
main time		XTIM:VOLT 1	XTIM:AM 1
gate time[1]		XTIM:VOLT:GATE 1	XTIM:AM:GATE 1
freq response[2]		XFR:POW:RAT 2,1	XFR:AM:RAT 2,1[5]
coherence[2]		XFR:POW:COH:2,1	XFR:AM:COH 2,1[5]
cross spectrum[2]		XFR:POW:CROS 2,1	XFR:AM:CROS 2,1[5]
auto correlation		XTIM:VOLT:CORR 1	XTIM:AM:CORR 1
cross correlation[2]		XTIM:VOLT:CORR:CROS 2,1	XTIM:AM:CORR:CROS 2,1[5]
capture buffer	TCAP 1	TCAP 1	TCAP 1
data register	D1to D6	D1 to D6	D1 to D6
instant spectrum	XFR:POW:INST 1	XFR:POW:INST 1	XFR:AM:INST 1
CCDF		XPOW:CCDF 1	

Digital/Video/W-CDMA Meas Types	
measured time	XTIM:DDEM:MEAS
measured spectrum	XFR:DDEM:MEAS
reference time	XTIM:DDEM:REF
reference spectrum	XFR:DDEM:REF
error magnitude[6]	XTIM:DDEM:ERR:MAGN
error phase[6]	XTIM:DDEM:ERR:PHAS
error vector time[7]	XTIM:DDEM:ERR:VECT
error vector spectrum[7]	XFR:DDEM:ERR:VECT
symbol table/err sum	XTIM:DDEM:SYMB
capture buffer	TCAP 1
data register	D1 to D6
chan freq response	XFR:DDEM:RAT
equalizer imp. response	XTIM:DDEM:IMP
code domain composite	XCOD:DDEM:COMP
code domain single	XCOD:DDEM:CLAYER <layer>
time domain composite	XTIM:DDEM:COMP

1 If SWE:TIME:GATE:STAT is OFF, data is same as main time
2 Available only with second input channel installed and both channels ON
4 For demod formats other than AM, replace AM with either FM or PM in the string. Use POW or VOLT if demod “type” is OFF.
5 In a 2-channel measurement, the measurement type used in the string is the same as that of channel 2.

Use POW or VOLT instead of AM, FM, or PM if demod is OFF.

[6] Called “carrier error” magnitude and phase for FSK demodulation.

[7] Called “FSK error” for FSK demodulation.

CALCulate[1|2|3|4]:FORMat **command/query**

Specifies the coordinates of the indicated trace.

Command Syntax: CALCulate[1|2|3|4]:FORMat <param>

<param> ::= MLIN|MLOG|PHASe|UPH|REAL|IMAG|GDELay|COMP|CONS|IEYE|QEYE|TEYE

Example Statements:

```
OUTPUT 719;":CALC4:FORM UPHASE"
OUTPUT 719;"calculate:form QEYE"
```

Query Syntax: CALCulate[1|2|3|4]:FORMat?

Return Format: CHAR

Attribute Summary:

Synchronization Required: no
Preset State: MLOG
SCPI Compliance: instrument-specific

Description:

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

MLIN specifies linear magnitude trace coordinates. When this coordinate type is active, the trace data is displayed with time or frequency data on the X axis and SQRT($RE^2 + IM^2$) on the Y axis.

MLOG specifies log-magnitude trace coordinates. When this coordinate type is active and the display units are dB, the trace data is displayed with time or frequency on the X axis and 10 log($RE^2 + IM^2$)on the Y axis if the data is in power units, or 20 log($RE^2 + IM^2$) if the data is linear (volts). If the Y-axis units are dBm, the trace data is displayed with time or frequency on the X axis and 10 log($Mag_{rms}^2 / .001 \times Z$) on the Y axis where Mag is the input magnitude and Z is the receiver input impedance (50Ω). The impedance value may be changed with the INP:IMP command.

PHAS specifies wrapped phase trace coordinates. When this coordinate type is active, the Y axis is phase in degrees. The default scale is ±225°.

UPH specifies unwrapped phase trace coordinates. When this coordinate type is active, the zero-phase reference and unwrapped-phase offset are values specified by the CALC:UPH:CREF command (in Hz) and CALC:UPH:OFFS command (in degrees), respectively.

REAL specifies real y-axis trace coordinates versus time or frequency. The real part of time waveforms is correctly scaled for either baseband or zoom.

IMAG specifies imaginary y-axis coordinates versus time or frequency and displays the imaginary part of complex data. If the data is purely real, zero values are displayed. Complex time waveforms have an imaginary part corresponding to the Hilbert transform of the real part (due to zooming). So, a 2Vpk sine wave input in zoom mode results in a Real Part display of a (frequency shifted) 2Vpk sine wave, and the Imag Part display will also be a (frequency shifted) 2Vpk sine wave shifted in phase.

GDEL specifies group delay trace coordinates. Group delay is the derivative of phase with respect to frequency, $-{}^{d\varphi}/_{df}$. In this analyzer it is approximated by the function $-{}^{\Delta\varphi}/_{\Delta f}$. The aperture ($\Delta f$) is specified with the CALC:GDAP:APER command.

Note Digital demodulation features are implemented only if option AYA (vector modulation analysis) is installed in the analyzer.

The following display formats are part of option AYA.

COMP specifies a *complex* polar vector diagram, with the real part on the x axis and the imaginary part on the y axis. When the instrument mode is digital demod, this is an IQ diagram, with the in-phase signal on the x axis and the quadrature-phase signal on the y axis.

The complex diagram differs from the constellation diagram in that the constellation diagram shows information only at specified time intervals; the symbol clock times.

CONS specifies a *constellation* polar vector diagram. This is a time-domain diagram. It displays information only at symbol clock times (detection decision-points) and the displayed points are called symbols.

IEYE specifies the *in-phase eye diagram*.

QEYE specifies the *quadrature-phase eye diagram*.

TEYE specifies the *trellis eye diagram*.

For more information on these display types, see online help and the concepts discussion in the *Operator's Guide*.

CALCulate[1|2|3|4]:GDAPerture:APERture command/query

Specifies the group delay aperture.

Command Syntax: `CALCulate[1|2|3|4]:GDAPerture:APERture <param>`

```
<param>   ::= <number>[<unit>]|<step>|<bound>
<number>  ::= a real number (NRf data)
              limits: 0.0625:16
<unit>    ::= PCT
<step>    ::= UP|DOWN
<bound>   ::= MAX|MIN
```

Example Statements:

```
OUTPUT 719;"Calc3:Gdapterture:Aper 16PCT"
OUTPUT 719;"CALC:GDAP:APER .0625"
```

Query Syntax: `CALCulate[1|2|3|4]:GDAPerture:APERture?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  +5E-1
SCPI Compliance:  instrument-specific
```

Description:

Values are rounded to the nearest acceptable values which are: .0625%,.125%,.25%,.5%,1%, 2%, 4%, 8%, or 16% of span.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1 | 2 | 3 | 4]:MARKer:BAND:STARt command/query

Specifies the start (left) band marker position.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:BAND:STARt <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -3.40282347E+38:3.40282347E+38
<unit>	::=	HZ|S|CODE
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"Calculate3:Mark:Band:Start 1e6;Stop 2e6"
OUTPUT 719;"CALC:MARKER:BAND:START 10 us;STOP 15 us"
```

Query Syntax: `CALCulate[1|2|3|4]:MARKer:BAND:STARt?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  4.5 MHz, 10.09375 us
SCPI Compliance:  instrument-specific
```

Description:

This command is used with CALC:MARK:BAND:STOP to define the band in which power or signal-to-noise measurements are made.

Related Commands:

- To set the band function, use CALC:MARK:FUNC.
- To read the band marker data, query with CALC:MARK:FUNC:RES?.

CALCulate[1 | 2 | 3 | 4]:MARKer:BAND:STOP **command/query**

Specifies the stop (right) band marker position.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:BAND:STOP <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -3.40282347E+38:3.40282347E+38
<unit>	::=	HZ|S|CODE
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;"Calculate3:Mark:Band:Start 1e6;Stop 2e6"
OUTPUT 719;"CALC:MARKER:BAND:START 10 us;STOP 15 us"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:BAND:STOP?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  5.5 MHz, 12.09375 us
SCPI Compliance:  instrument-specific
```

Description:

This command is used with CALC:MARK:BAND:STARt to define the band in which power or signal-to-noise measurements are made.

Related Commands:

- To set the band function, use CALC:MARK:FUNC.
- To read the band marker data, query with CALC:MARK:FUNC:RES?.

CALCulate[1|2|3|4]:MARKer:COUPled[:STATe] command/query

Turns marker coupling on and off.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:COUPled[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Calculate2:Mark:Coup:State OFF"
OUTPUT 719;"CALC:MARKER:COUP ON"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:COUPled[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Description:

Marker coupling ties the X values of the inactive traces to that of the active trace. This command couples the markers no matter which trace is specified. No trace number is needed, as in CALC:MARK:COUP 1.

Marker coupling is applicable only between traces that have the same x-axis coordinates. For example, marker coupling is meaningless if one trace has x-axis units of Hertz and another trace has x-axis units of seconds. When coupling is on, changing the position of one marker causes the marker(s) on other traces *of the same type* (frequency or time) to move to the same position.

Related Commands:

- To place a marker at a particular position on the X axis, use CALC:MARK:X.
- To display four traces, use DISP:FORM FOUR.

CALCulate[1|2|3|4]:MARKer:FCOunt command/query

Turns the marker frequency counter on/off.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:FCOunt OFF|0|ON|1`

Example Statements:
OUTPUT 719;":calc4:marker:fco OFF"
OUTPUT 719;"Calculate3:Mark:Fco OFF"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:FCOunt?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

For a description of this function, see Help Text under the Help hardkey.

CALCulate[1|2|3|4]:MARKer:FCOunt:RESult? query

Query results of the marker frequency counter measurement.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:FCOunt:RESult?`

Example Statements:

OUTPUT 719;"CALCULATE:MARK:FCOUNT:RES?"
OUTPUT 719;"calc:marker:fco:result?"

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

CALC:MARK:FCOUNT must be ON for there to be data available for this query.

CALCulate[1|2|3|4]:MARKer:FUNCtion — command/query

Initiate special band power marker functions.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:FUNCtion OFF|0|BPOWer|BRMS|BCN|BCNO`

Example Statements:

```
OUTPUT 719;":Calc2:Mark:Function BCN"
OUTPUT 719;"CALC2:MARKER:FUNC OFF"
```

Query Syntax: `CALCulate[1|2|3|4]:MARKer:FUNCtion?`

Return Format: `OFF|BPOW|BRMS|BCN|BCNO`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  OFF
SCPI Compliance:  instrument-specific
```

Description:

This command starts a marker measurement whose type is as follows:

BPOW calculates band power. The marker positions are set with CALC:MARK:BAND:START and STOP.

BRMS calculates rms-square-root of the band power. The marker positions are set with CALC:MARK:BAND:START and STOP.

BCN calculates the band carrier-to-noise ratio (C/N). The carrier position value used is the position of the X marker (set with CALC:MARK:X). If the marker is off, this command calculates noise instead of carrier-to-noise ratio.

BCNO calculates the band carrier-to-noise density ratio. The carrier position value used is the position of the X marker (set with CALCMARKX). If the marker is off, this command calculates noise density instead of carrier-to-noise density ratio.

The selected measurement runs and updates itself with every time record gathered (or, for cases like averaging where multiple time records are needed to create a measurement result, every time a measurement result is available). To run the same measurement on another set of data, change the variable of interest, synchronize, and query again.

The query returns the type of measurement that is currently being calculated. To query the data, use CALC:MARK:FUNC:RES?.

CALCulate[1|2|3|4]:MARKer:FUNCtion:DDEMod:RESult? query

Queries error parameters for digital or video demodulation or wideband CDMA.

Query Syntax:

```
CALCulate[1|2|3|4]:MARKer:FUNCtion:DDEMod:RESult? <param>
<param>
::=ADR|DEV|EVRM|EVPK|EVPS|EVPA|EVNF|FERR|FSRM|FSPK|FSPS|IQOF|
LCOD|MERM|MEPK|MEPS|PERM|PBIT|PEPK|PEPS|PLEV|RHO|RHOE|SNR|
SLOT|TDPC|TFRM|TSLT|TTRG
```

Example Statements:

OUTPUT 719;"calc3:marker:func:ddemod:res? SNR"
OUTPUT 719;"Calc:Marker:Func:Ddemod:Res? FERR"

Return Format: `Real`

Attribute Summary:

```
Option:  AYA (vector modulation analysis) and
B7A (Enhanced Data rates for GSM Evolution (EDGE)) or
B73 (Digital Wideband CDMA analysis) or
080 Digital 3GPP Wideband CDMA analysis

Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

ADR is the amplitude droop in dB/symbol.

DEV is the FSK deviation in Hz.

EVRM is error vector magnitude in %rms units.

EVNF is the error vector magnitude (EVM) below which 95% of the individual EVM's occur (option B7A).

EVPA is the mean (average) of the peak error vector magnitudes—- one per measurment (option B7A).

EVPK is the peak (largest) error vector magnitude in %pk units (option B7A).

EVPS is the symbol number at which the peak (largest) error vector magnitude occurred.

FERR is the frequency error (carrier frequency offset) in Hz.

FSPK is the peak FSK error in %pk units.

FSPS is the symbol number at which the peak FSK error occured.

FSRM is the FSK error in %rms units.

IQOF is the IQ offset in dB.

LCOD is a Wideband CDMA parameter that shows the scrambling code (also called long code) used to despread the signal.

MERM is the magnitude error in %rms units.

MEPK is the peak (largest) magnitude error in %pk units.MEPS is the symbol number at which the peak (largest) magnitude error occurred.

PBIT is the number of pilot bits detected in the DCPH for the selected channel and code layer (Option 080).

PERM is phase error in degrees-rms units.

PEPK is the peak (largest) phase error in degrees-pk.

PEPS is the symbol number at which the peak (largest) phase error occurred.

PLEV is the VSB pilot level.

RHO is the modulation accuracy.

RHOE is a Wideband CDMA parameter that shows the estimation of RHO for the W-CDMA composite signal.

SLOT is a Wideband CDMA parameter that shows the slot number associated with the start of the displayed result.

SNR is the signal-to-noise ratio.

TDPC is the tDPCH timing value for the DPCH for the selected channel and code layer (Option 080)

TFRM is a Wideband CDMA parameter that shows the selected code channel offset, T_{Frame} (option B73 or B79).

TSLT is a Wideband CDMA parameter that shows the selected code channel offset, $T_{Slot(option}$ (options B73 or B79).

TTRG is a Wideband CDMA parameter that shows the amount of time, in chips, from the trigger to the start of the sync symbol.

For more information, see online help (press the **[Help**] key on the front panel, then press the softkey of interest) and see the concepts discussion in the *Operator's Guide*.

CALCulate[1|2|3|4]:MARKer:FUNCtion:RESult? query

Retrieves the marker measurement data created by the CALC:MARK:FUNC command.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:FUNCtion:RESult?`

Example Statements:
OUTPUT 719;":CALC:MARKER:FUNC:RESULT?"
OUTPUT 719;"calc3:mark:function:res?"

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

CALC:MARK:FUNC selects the type of measurement to make and it begins running immediately. The selected measurement runs and updates itself with every time record gathered (or, for cases like averaging where multiple time records are needed to create a measurement result, every time a measurement result is available). To run the same measurement on another set of data, change the variable of interest, synchronize, and query again.

CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics — command/query

Selects or turns off statistical-power measurements.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics OFF|0|PEAK|PKAV|AVER`

Example Statements:

```
OUTPUT 719;"Calculate:Mark:Func:Statistics OFF"
OUTPUT 719;"CALC:MARKER:FUNC:STAT PKAV"
```

Query Syntax: `CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics?`

Return Format: `OFF|PEAK|PKAV|AVER`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  OFF
SCPI Compliance:  instrument-specific
```

Description:

This command selects a statistical-power measurement or turns the selected statistical-power measurement off. You can select peak power, average power, and peak-to-average power (the ratio of peak-to-average). CALC1 selects trace A for the measurement, CALC2 selects trace B, and so forth. If no trace is specified, CALC1 is assumed.

To compute the selected power measurement, the trace must contain time-domain data and the instrument mode cannot be Scalar. You select trace data with the CALCulate:FEED command. You select the instrument mode with the INSTrument[:SELect] command.

For peak and peak-to-average power, the analyzer uses the value specified by CALC:MARK:FUNC:STAT:PPCT to compute peak power. For details, see online help for the [peak percent] softkey and the [statistics on/off] softkey (on the analyzer, press [Help], [Marker Function], [peak/average statistics], [statistics on/off] or [peak percent]).

Related Commands:

- To return the statistical-power value, use CALC:MARK:FUNC:STAT:RES?.
- To set peak percent used in peak-power measurements, use CALC:MARK:FUNC:STAT:PPCT.
- To return the number of samples used in the power measurement, use CALC:MARK:FUNC:STAT:SMPL?.

CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics:PPCT command/query

Specifies the peak-percent value to use with peak power and peak-to-average power measurements.

Command Syntax:

```
CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics:PPCT
{<number>[<unit>]}|<step>|<bound>
```

<number> ::= a real number (NRf data)
limits: 90:99.99

<unit> ::= [PCT]

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

OUTPUT 719;"calc3:mark:function:stat:ppct 95"
OUTPUT 719;"CALC:MARK:FUNCTION:STAT:PPCT 99.8"

Query Syntax:

```
CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics:PPCT?
```

Return Format:

```
Real
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  99.99
SCPI Compliance:  instrument-specific
```

Description:

You enable peak-power or peak-to-average power measurements with the CALC:MARK:FUNC:STAT command. Peak percent is used in peak-power measurements to specify the probability of peak power. Since peak-to-average power is the ratio of peak power to average power, peak percent affects peak-to-average power measurements as well as peak-power measurements.

Peak percent specifies the probability of peak power. For example, a peak percent of 99.8% means that the instantaneous power will be less-than-or-equal-to the peak power reported by the analyzer 99.8% of the time. Alternatively, you could say that the instantaneous power will exceed the peak power reported by the analyzer 0.2% of the time (100% - 99.8%).

CALC 1 sets the peak percent for trace A, CALC2 sets it for trace B, and so forth. If no trace is specified, CALC1 is assumed. The value is used only when peak power or peak-to-average power is enabled (by sending CALC[1|2|3|4]:MARK:FUNC:STAT PEAK or CALC[1|2|3|4]:MARK:FUNC:STAT PKAV).

For additional details, see online help for the [peak power] softkey (on the analyzer, press [Help], [Marker Function], [peak/average statistics], [peak power]).

CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics:RESult? query

Returns the results of statistical-power measurements.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics:RESult?`

Example Statements:

```
OUTPUT 719;":calculate3:mark:function:stat:res?"
OUTPUT 719;"Calculate4:Mark:Function:Stat:Res?"
```

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command returns the results of a statistical-power measurement for the selected trace. CALC1 returns the results for trace A, CALC2 for trace B, and so forth. If no trace is specified, CALC1 is assumed.

For details about statistical-power measurements, see the CALC:MARK:FUNC:STAT command.

CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics:SMPL? query

Returns the number of samples used in statistical-power measurements.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:FUNCtion:STATistics:SMPL?`

Example Statements:

```
OUTPUT 719;"CALCULATE4:MARK:FUNCTION:STAT:SMPL?"
OUTPUT 719;"calculate:mark:function:stat:smpl?"
```

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

You use the CALC:MARK:FUNC:STAT command to enable or disable statistical-power measurements. These measurements compute power using samples from all time records since the statistical-power measurement began.

The number of samples used is set to zero when you start or restart the statistical-power measurement. You start the measurement by sending CALC:MARK:FUNC:STAT and enabling a statistical-power measurement. You restart the measurement by sending the ABOR;*WAI command or most commands that change measurement parameters (such as range, trigger, or frequency).

Note that changing the peak percent (CALC:MARK:FUNC:STAT:PPCT) does not restart the power measurement. Also note that selecting another statistical-power measurement does not restart the power measurement (for example, switching from peak power to average or peak-to-average power).

Use CALC:MARK:FUNC:STAT:SMPL? to report the number of samples used in the statistical-power measurement. CALC1 reports the number of samples used in the statistical-power measurement on trace A, CALC2 reports it for trace B, and so forth. If no trace is specified, CALC1 is assumed. If no statistical-power measurement is selected for the specified trace, the command returns zero.

CALCulate[1|2|3|4]:MARKer:MAXimum **command**

Moves the marker to the highest peak (largest value) in the active trace.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:MAXimum`

Example Statements:
OUTPUT 719;":Calculate4:Mark:Maximum"
OUTPUT 719;"CALC:MARK:MAXIMUM"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If the marker is OFF when this command is issued, the marker is turned ON. Therefore, it is not necessary to check the marker state before using this command.

If no number is given in the CALC branch, “1" is assumed.

See Marker Search in Help Text for hints on peak searching.

Related Commands:

- To read marker position, use CALC:MARK:X? and CALC:MARK:Y?.
- To find the minimum value of the trace, use CALC:MARK:MIN.
- To continually move the marker to the maximum value of the trace as it is updated, use CALC:MARK:MAX:TRAC.
- To find other peaks in the trace, use CALC:MARK:MAX:NEXT, CALC:MARK:MAX:LEFT, and CALC:MARK:MAX:RIGH.
- To search for a specific value, use the CALC:MARK:SEAR commands.

CALCulate[1|2|3|4]:MARKer:MAXimum:LEFT command

Moves the marker left to the next local maximum in the trace.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:MAXimum:LEFT`

Example Statements:
OUTPUT 719;"calc:marker:max:left"
OUTPUT 719;"Calculate:Mark:Maximum:Left"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If the marker is OFF when this command is issued, the marker is turned ON. Therefore, it is not necessary to check the marker state before using this command.

If no number is given in the CALC branch, “1" is assumed.

See Marker Search in Help Text for hints on peak searching.

Related Commands:

- To read marker position, use CALC:MARK:X? and CALC:MARK:Y?.
- To find the maximum value of the trace, use CALC:MARK:MAX.
- To find the minimum value of the trace, use CALC:MARK:MIN.
- To continually move the marker to the maximum value of the trace as it is updated, use CALC:MARK:MAX:TRAC.
- To find other peaks in the trace, use CALC:MARK:MAX:RIGH and CALC:MARK:MAX:NEXT.
- To search for a specific value, use the CALC:MARK:SEAR commands.

CALCulate[1 | 2 | 3 | 4]:MARKer:MAXimum:NEXT command

Moves the marker to the next-highest peak in the active trace.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:MAXimum:NEXT`

Example Statements:

```
OUTPUT 719;":CALC:MARKER:MAX:NEXT"
OUTPUT 719;"calc4:mark:maximum:next"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If the marker is OFF when this command is issued, the marker is turned ON. Therefore, it is not necessary to check the marker state before using this command.

If no number is given in the CALC branch, “1" is assumed.

See Marker Search in Help Text for hints on peak searching.

Related Commands:

- To read marker position, use CALC:MARK:X? and CALC:MARK:Y?.
- To find the maximum value of the trace, use CALC:MARK:MAX.
- To find the minimum value of the trace, use CALC:MARK:MIN.
- To continually move the marker to the maximum value of the trace as it is updated, use CALC:MARK:MAX:TRAC.
- To find other peaks in the trace, use CALC:MARK:MAX:LEFT and CALC:MARK:MAX:RIGH.
- To search for a specific value, use the CALC:MARK:SEAR commands.

CALCulate[1 | 2 | 3 | 4]:MARKer:MAXimum:RIGHt **command**

Moves the marker right to the next local maximum in the trace.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:MAXimum:RIGHt`

Example Statements:
OUTPUT 719;"Calculate:Mark:Max:Right"
OUTPUT 719;"CALC:MARKER:MAX:RIGH"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If the marker is OFF when this command is issued, the marker is turned ON. Therefore, it is not necessary to check the marker state before using this command.

If no number is given in the CALC branch, “1" is assumed.

See Marker Search in Help Text for hints on peak searching.

Related Commands:

- To read marker position, use CALC:MARK:X? and CALC:MARK:Y?.
- To find the maximum value of the trace, use CALC:MARK:MAX.
- To find the minimum value of the trace, use CALC:MARK:MIN.
- To continually move the marker to the maximum value of the trace as it is updated, use CALC:MARK:MAX:TRAC.
- To find other peaks in the trace, use CALC:MARK:MAX:LEFT and CALC:MARK:MAX:NEXT.
- To search for a specific value, use the CALC:MARK:SEAR commands.

CALCulate[1|2|3|4]:MARKer:MAXimum:TRACk command/query

Turns peak tracking on and off.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:MAXimum:TRACk OFF|0|ON|1`

Example Statements:

```
OUTPUT 719;":calculate3:mark:maximum:trac ON"
OUTPUT 719;"Calc4:Marker:Max:Track ON"
```

Query Syntax: `CALCulate[1|2|3|4]:MARKer:MAXimum:TRACk?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

When peak tracking is on, the analyzer continuously moves the marker to the peak value in the active trace. If peak track is on for more than one trace, each marker follows the peak for its respective trace, unless marker coupling is on. If peak tracking is on and markers are coupled, only the peak on the active trace is tracked and the X-position of the other marker(s) follows that of the active trace.

CALCulate[1|2|3|4]:MARKer:MINimum[:GLOBal] command

Moves the active marker to the minimum value in the active trace.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:MINimum[:GLOBal]`

Example Statements:

OUTPUT 719;"CALC2:MARK:MINIMUM:GLOB"
OUTPUT 719;"calculate:mark:min"

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If the marker is OFF when this command is issued, the marker is turned ON. Therefore, it is not necessary to check the marker state before using this command.

If no number is given in the CALC branch, “1" is assumed.

See Marker Search in Help Text for hints on peak searching.

Related Commands:

- To read marker position, use CALC:MARK:X? and CALC:MARK:Y?.
- To find the maximum value of the trace, use CALC:MARK:MAX.
- To find other peaks in the trace, use CALC:MARK:MAX:RIGH and CALC:MARK:MAX:NEXT.
- To search for a specific value, use the CALC:MARK:SEAR commands.

CALCulate[1 | 2 | 3 | 4]:MARKer:OFFSet[:STATe] command/query

Turns the offset marker on and off.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:OFFSet[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Calculate2:Mark:Offset OFF"
OUTPUT 719;"CALC3:MARK:OFFSET:STAT OFF"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:OFFSet[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Description:

If no number is given in the CALC branch, “1" is assumed.

Absolute marker values may be read without turning the offset marker off with CALC:MARK:Y?.

Related Commands:

- To set the offset marker’s X and Y values, use CALC:MARK:OFFS:X and CALC:MARK:OFFS:Y.
- To read out the offset values, use CALC:MARK:OFFS:X? and CALC:MARK:OFFS:Y?.
- To change the offset marker’s X and Y values to those of the current marker values, use CALC:MARK:OFFS:ZERO.

CALCulate[1|2|3|4]:MARKer:OFFSet:X command/query

Specifies the X value of the offset marker.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:OFFSet:X <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -3.40282e+38:3.40282e+38
<unit>	::=	HZ|S|CODE
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;":calc:marker:offs:x 2 MHZ"
OUTPUT 719;"Calculate3:Mark:Offset:X 3.14159E+006"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:OFFSet:X?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0 (HZ or s)
SCPI Compliance:  instrument-specific
```

Description:

If no number is given in the CALC branch, “1" is assumed.

Related Commands:

- To turn on the offset marker, use CALC:MARK:OFFS 1.
- To specify the Y value of the offset marker, use CALC:MARK:OFFS:Y.
- To position the offset marker at the values of the main marker, use CALC:MARK:OFFSET:ZERO.
- To read the relative marker (offset) values, use CALC:MARK:X:REL? and CALC:MARK:Y:REL?.

CALCulate[1|2|3|4]:MARKer:OFFSet:Y **command/query**

Specifies the Y value of the offset marker.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:OFFSet:Y <param>`

<param> ::= <number>[<unit>]|<step>|<bound>

<number> ::= a real number (NRf data)
limits: (depends on configuration)

<unit> ::= (see listing below)

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"CALC:MARKER:OFFS:Y -10 DBM"
OUTPUT 719;"calc:mark:offset:y 5.13158 VRMS"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:OFFSet:Y?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (units vary with measurement type)
SCPI Compliance:  instrument-specific
```

Units List

am	dBampk2/Hz	dBradrms2/Hz	Hzpk2	radpk2	Vpk
am2	dBamrms2	dBV	Hzpk2/Hz	radpk2/Hz	Vpk/rtHz
ampk	dBamrms2/Hz	dBV/rtHz	Hzrms	radrms	Vpk2
ampk/rtHz	dBHzpk2	dBVpk	Hzrms/rtHz	radrms/rtHz	Vpk2/Hz
ampk2	dBHzpk2/Hz	dBVpk/rtHz	Hzrms2	radrms2	Vrms
ampk2/Hz	dbHzrms2	dBVrms	Hzrms2/Hz	radrms2/Hz	Vrms/rtHz
amrms	dBHzrms2/Hz	dBVrms/rtHz	MHz	s	Vrms2
amrms/rtHz	dBm	deg	pct	unitless	Vrms2/Hz
amrms2	dBm/Hz	Hz	rad	V	W
amrms2/Hz	dBradpk2	Hz2	rad2	V/rtHz	W/Hz
dB	dBradpk2/Hz	Hzpk	radpk	V2	Wrms
dBampk2	dBradrms2	Hzpk/rtHz	radpk/rtHz	V2/Hz	Wrms/Hz

Description:

If no number is given in the CALC branch, “1" is assumed.

Units depend on the measurement data type.

Related Commands:

- The measurement data type is specified with the CALC:FEED command.
- To turn on the offset marker, use CALC:MARK:OFFS 1.
- To position the offset marker at the values of the main marker, use CALC:MARK:OFFSET:ZERO.
- To read the relative marker (offset) values, use CALC:MARK:X:REL? and CALC:MARK:Y:REL?.

CALCulate[1|2|3|4]:MARKer:OFFSet:Z command/query

Specifies the Z value of the offset marker for waterfall and spectrogram displays.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:OFFSet:Z <param>`

<param> ::= <number>[<unit>]|<step>|<bound>

<number> ::= a real number (NRf data)
limits: (depends on configuration)

<unit> ::= S

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;"CALC:MARKER:OFFS:Z 1e-3S"
OUTPUT 719;"calc:mark:offset:z 2 S"
```

Query Syntax: `CALCulate[1|2|3|4]:MARKer:OFFSet:Z?`

Return Format: `Real`

Attribute Summary:

```
Option:  AYB (Waterfall and Spectrogram Analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command sets the offset marker's z-axis position when the selected trace is a waterfall or spectrogram display.

Each trace in a waterfall or spectrogram display has a time stamp. You specify the z-axis position of the offset marker by a trace's time stamp (in seconds).

The time stamp is the traces's z-axis value in seconds, and is the time that elapsed from when the measurement began (sending the ABORt command) to when the trace was acquired. The analyzer automatically selects the closest valid time stamp (the closest valid z-axis position).

CALCulate[1|2|3|4]:MARKer:OFFSet:ZERO command

Changes the offset marker's X and Y values to those of the marker.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:OFFSet:ZERO`

Example Statements:

OUTPUT 719;"calculate4:mark:offs:zero"
OUTPUT 719;"Calc:Marker:Offs:Zero"

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command moves the offset marker to the same position as the relative marker, setting the offset value to zero.

If either the main or offset markers are turned off, this command turns them on. Therefore, it is not necessary to turn them on prior to zeroing.

If no number is given in the CALC branch, "1" is assumed.

Related Commands:

- To position the relative marker, use CALC:MARK:X.
- To read the relative marker (offset) values, use CALC:MARK:X:REL? and CALC:MARK:Y:REL?.
- To position the offset marker at a position other than that of the main marker, use CALC:MARK:OFFSET:X and CALC:MARK:OFFSET:Y.

CALCulate[1 | 2 | 3 | 4]:MARKer:POLar:UNIT:POWer command/query

Specifies the marker readout units when the trace type is polar.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:POLar:UNIT:POWer DBM|WRMS|W|V`

Example Statements:

```
OUTPUT 719;":CALCULATE4:MARK:POLAR:UNIT:POW Wrms"
OUTPUT 719;"calculate:mark:polar:unit:pow W"
```

Query Syntax: `CALCulate[1|2|3|4]:MARKer:POLar:UNIT:POWer?`

Return Format: `CHAR`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  V
SCPI Compliance:  instrument-specific
```

Description:

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1|2|3|4]:MARKer:READout — command/query

Specifies marker readout type when trace type is polar.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:READout MPHase|RIMaginary`

Example Statements:
OUTPUT 719;"Calculate:Mark:Readout MPHASE"
OUTPUT 719;"CALC:MARK:READOUT MPHASE"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:READout?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  MPH
SCPI Compliance:  instrument-specific
```

Description:

MPHase specifies magnitude-phase coordinates.

RIMaginary specifies real-imaginary coordinates.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1|2|3|4]:MARKer:SEARch:BUFFer[:STATe] command/query

Controls buffer search for waterfall and spectrogram displays.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:SEARch:BUFFer[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;":calc4:marker:sear:buff OFF"
OUTPUT 719;"Calculate3:Mark:Search:Buff:Stat ON"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:SEARch:BUFFer[:STATe]?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  AYB (Waterfall and Spectrogram Analysis)
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

This command controls the analyzer's buffer search feature. When this feature is on, marker-to-peak, next peak, and marker-to-minimum search operations are performed over all traces in waterfall and spectrogram displays. When this feature is off, these search operations are only performed on the selected trace (the trace selected by CALC:MARK:Z).

Related Commands:

- To search for the highest peak, use CALC:MARK:MAX.
- To search for the next highest peak, use CALC:MARK:MAX:NEXT.
- To search for the smallest peak, use CALC:MARK:MIN.

CALCulate[1 | 2 | 3 | 4]:MARKer:SEARch:LEFT command

Moves the marker left from its present location to the first occurrence of the Y-axis target value.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:SEARch:LEFT`

Example Statements:

OUTPUT 719;"CALCULATE3:MARK:SEARCH:LEFT"
OUTPUT 719;"calc:marker:sear:left"

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Related Commands:

- To specify the target value, use CALC:MARK:SEAR:TARG.
- To move the marker to the largest value on the trace, use CALC:MARK:MAX.
- To turn on the offset marker and place it on the main marker, use CALC:MARK:OFFS:ZERO.
- To read marker position (X and Y-axis values), use CALC:MARK:X? and CALC:MARK:Y?.

CALCulate[1 | 2 | 3 | 4]:MARKer:SEARch:OFFSet command

Moves the regular marker to the position of the reference or offset marker.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:SEARch:OFFSet`

Example Statements:
OUTPUT 719;":Calc2:Mark:Search:Offs"
OUTPUT 719;"CALCULATE2:MARK:SEAR:OFFSET"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Related Commands:

- To move the offset marker to the current position of the regular marker (also called zero the offset marker), use CALC:MARK:OFFS:ZERO.

CALCulate[1 | 2 | 3 | 4]:MARKer:SEARch:RIGHt **command**

Moves the marker right from its present position to the first occurrence of the Y-axis target value.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:SEARch:RIGHt`

Example Statements:

OUTPUT 719;"calc2:marker:sear:righ"
OUTPUT 719;"Calculate:Mark:Search:Righ"

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Related Commands:

- To specify the target value, use CALC:MARK:SEAR:TARG.
- To move the marker to the largest value on the trace, use CALC:MARK:MAX.
- To turn on the offset marker and place it on the main marker, use CALC:MARK:OFFS:ZERO.
- To read marker position (X and Y-axis values), use CALC:MARK:X? and CALC:MARK:Y?.

CALCulate[1 | 2 | 3 | 4]:MARKer:SEARch:TARGet **command/query**

Specifies a Y-axis target for a marker search.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:SEARch:TARGet <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -600:600
<unit>	::=	(see listing below)
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;"Calculate1:Mark:Search:Targ -3 dBm"
OUTPUT 719;"CALC:MARKER:SEAR:TARGET 180 DEG"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:SEARch:TARGet?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  -3 (units depend on data and format)
SCPI Compliance:  instrument-specific
```

Units List

am	dBampk2/Hz	dBradrms2/Hz	Hzpk2	radpk2	Vpk
am2	dBamrms2	dBV	Hzpk2/Hz	radpk2/Hz	Vpk/rtHz
ampk	dBamrms2/Hz	dBV/rtHz	Hzrms	radrms	Vpk2
ampk/rtHz	dBHzpk2	dBVpk	Hzrms/rtHz	radrms/rtHz	Vpk2/Hz
ampk2	dBHzpk2/Hz	dBVpk/rtHz	Hzrms2	radrms2	Vrms
ampk2/Hz	dbHzrms2	dBVrms	Hzrms2/Hz	radrms2/Hz	Vrms/rtHz
amrms	dBHzrms2/Hz	dBVrms/rtHz	MHz	s	Vrms2
amrms/rtHz	dBm	deg	pct	unitless	Vrms2/Hz
amrms2	dBm/Hz	Hz	rad	V	W
amrms2/Hz	dBradpk2	Hz2	rad2	V/rtHz	W/Hz
dB	dBradpk2/Hz	Hzpk	radpk	V2	Wrms
dBampk2	dBradrms2	Hzpk/rtHz	radpk/rtHz	V2/Hz	Wrms/Hz

Description:

When the offset marker is on, this value is relative to the offset (or reference) Y-axis position. Otherwise, it represents the absolute values. A query returns a real value in the current Y-axis units, regardless of which units were used to specify the target.

Related Commands:

- To determine the current Y-axis units, use the CALC:UNIT commands.
- To search for the target value, use CALC:MARK:SEAR:LEFT and CALC:MARK:SEAR:RIGH.
- To move the marker to the largest value on the trace, use CALC:MARK:MAX.
- To turn on the offset marker and place it on the main marker, use CALC:MARK:OFFS:ZERO.
- To read marker position (X and Y-axis values), use CALC:MARK:X? and CALC:MARK:Y?.

CALCulate[1 | 2 | 3 | 4]:MARKer[:STATe] command/query

Turns markers on and off.

Command Syntax: `CALCulate[1|2|3|4]:MARKer[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;":CALC3:MARKER ON"
OUTPUT 719;"calc:marker:stat ON"

Query Syntax: `CALCulate[1|2|3|4]:MARKer[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1 (on)
SCPI Compliance:  instrument-specific
```

Description:

Turns the marker on or off for the specified trace. If no number is given in the CALC branch, “1" is assumed.

The trace is specified with the numbers in CALC[1|2|3|4] which correspond to the front-panel trace designations A, B, C, and D. If no trace number is used with this command, a “1" is assumed and the marker on trace A is turned on or off.

- To find the X and Y values of the marker’s position, use CALC:MARK:X? and CALC:MARK:Y?.
- To specify the marker’s X position, use CALC:MARK:X.
- To make marker measurements relative to a reference other than 0,0, see the CALC:MARK:OFFSet commands.

Note Measurements run faster with markers off.

CALCulate[1|2|3|4]:MARKer:TRACe command/query

Turns the waterfall/spectrogram trace select marker on and off.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:TRACe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Calc:Marker:Trac ON"
OUTPUT 719;"CALCULATE:MARK:TRAC OFF"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:TRACe?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

The trace select marker allows selection of a scan of data in the z-direction (time history) for further analysis.

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

CALCulate[1|2|3|4]:MARKer:X[:ABSolute] command/query

Places the main marker at the specified X value.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:X[:ABSolute] <param>`

```
<param>  ::= <number>[<unit>]|<step>|<bound>
<number> ::= a real number (NRf data)
<unit>   ::= HZ|S|CODE
<step>   ::= UP|DOWN
<bound>  ::= MAX|MIN
```

Example Statements:

OUTPUT 719;":CALCULATE4:MARK:X 7.5 MHZ"
OUTPUT 719;"calc2:mark:x:abs 3.00E+006"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:X[:ABSolute]?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  nearest bin to 5 MHz
SCPI Compliance:  instrument-specific
```

Description:

If no trace number is specified, “1" is assumed and the command affects the position of the trace A marker.

Marker X-Axis Limits and Preset Values

	89410 or 89441 IF	89441 RF
value limits	0:10 MHz	2 MHz:2.650 GHz

IF indicates ROUT:REC IF or RF1 is active.
RF indicates ROUT:REC RF2 is active.

Related Commands:

- To turn the marker on, use CALC:MARK[:STAT].
- To determine the units to send with the marker position, use TRACe:X:UNIT?.

CALCulate[1|2|3|4]:MARKer:X:ACHannel? query

Returns the total number of active channels in a code-domain power display.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:X:ACHannel?`

Example Statements:

OUTPUT 719;":calculate4:mark:x:ach?"
OUTPUT 719;"Calc3:Marker:X:Achannel?"

Return Format: `Integer`

Attribute Summary:

```
Option:  B73 (Wideband CDMA analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command returns the total number of active channels for the active trace when the trace is a code-domain power display. If the trace isn't a code-domain power display, the command returns zero.

CALCulate[1 | 2 | 3 | 4]:MARKer:X:CCHannel? query

Returns the x-axis code-channel marker value.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:X:CCHannel?`

Example Statements:
OUTPUT 719;":calculate4:mark:x:cch?"
OUTPUT 719;"Calc3:Marker:X:Cchannel?"

Return Format: `Integer`

Attribute Summary:
```
Option:  B73 (Wideband CDMA Analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

In code-domain power displays, the marker shows the current code layer and code channel. This command returns the code channel for the marker's current position.

If the active trace is a composite code-domain power display, this command returns the code channel for the appropriate code layer. For example, if the marker is on channel 16 in code layer 6, this command returns "6".

CALCulate[1|2|3|4]:MARKer:X:CLAYer? query

Returns the x-axis code-layer marker value.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:X:CLAYer?`

Example Statements:
OUTPUT 719;"CALC:MARK:X:CLAY?"
OUTPUT 719;"calculate:mark:x:clayer?"

Return Format: `Integer`

Attribute Summary:
```
Option:  B73 (Wideband CDMA Analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

In code-domain power displays, the marker shows the current code layer and code channel. This command returns the code layer for the marker's current position.

CALCulate[1|2|3|4]:MARKer:X:CSTatus? query

Returns the marker status for code-domain power displays.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:X:CSTatus?`

Example Statements:
OUTPUT 719;":Calc4:Marker:X:Cst?"
OUTPUT 719;"CALCULATE4:MARK:X:CST?"

Return Format: `Integer`

Attribute Summary:
```
Option:  B73 (Wideband CDMA Analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

This command is applicable only when the active trace is code-domain power. It returns 0 (zero) if the marker is not on an active channel; 1 if the marker is on an active channel.

CALCulate[1 | 2 | 3 | 4]:MARKer:X:RELative? query

Returns the X-axis offset marker value (relative to the offset marker position).

Query Syntax: `CALCulate[1|2|3|4]:MARKer:X:RELative?`

Example Statements:

OUTPUT 719;"calc3:marker:x:relative?"
OUTPUT 719;"Calc:Mark:X:Rel?"

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Related Commands:

- To return the absolute marker value, use CALC:MARK:Y?.
- To determine the X-axis units, use TRAC:X:UNIT?.

CALCulate[1|2|3|4]:MARKer:Y? query

Reads the Y-axis value of the main marker.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:Y?`

Example Statements:

OUTPUT 719;":CALCULATE2:MARK:Y?"
OUTPUT 719;"calculate2:mark:y?"

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command is used to read the Y-axis value of the main marker. It reads out the absolute value regardless of the offset marker state.

Related Commands:

- To determine the Y-axis units, use CALC:UNIT:<unit>?
- To read the X-axis value, use CALC:MARK:X?.
- To determine the X-axis units, use TRAC:X:UNIT?.
- To move the main marker, use CALC:MARK:X.
- To turn the marker on/off, use CALC:MARK[:STATE].

CALCulate[1 | 2 | 3 | 4]:MARKer:Y:RELative? query

Returns the Y-axis offset marker value (relative to the reference marker position.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:Y:RELative?`

Example Statements:
OUTPUT 719;"Calc:Mark:Y:Rel?"
OUTPUT 719;"CALCULATE:MARK:Y:RELATIVE?"

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Related Commands:

- To return the absolute marker value, use CALC:MARK:Y?.
- To determine the X-axis units, use TRAC:X:UNIT?.

CALCulate[1|2|3|4]:MARKer:Z[:ABSolute] command/query

Specifies which trace in a waterfall or spectrogram display is being measured with the marker(s).

Command Syntax: `CALCulate[1|2|3|4]:MARKer:Z[:ABSolute] <param>`

<param> ::= <number>|<step>|<bound>

<number> ::= a real number (NRf data)
limits: 1:(see discussion)

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"calculate2:mark:z:absolute 23"
OUTPUT 719;"Calc2:Marker:Z 99"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:Z[:ABSolute]?`

Return Format: `Real`

Attribute Summary:
```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  (see discussion)
SCPI Compliance:  instrument-specific
```

Description:

This command is useful when a waterfall or spectrogram measurement has been made and the measurement is paused. Before using it, you must turn the trace-analysis feature on; see related commands listed below.

You can select a trace by its trace number or its time stamp. Trace number 1 represents the earliest (oldest) data and larger numbers represet more-recent data. If the buffer depth is 100 and 100 traces are displayed, then sending `calc:mark:z 100` specifies analysis of the most-recent measurement trace.

The time stamp is the traces's z-axis value in seconds, and is the time that elapsed from when the measurement began (sending the ABORt command) to when the trace was acquired.

The MARKer:Z:UNIT command determines the units returned with CALC:MARK:Z? The number returned is the absolute number, even if the offset marker is on. If the offset marker is on and you want to query the marker position relative to the offset marker, send CALC:MARK:Z:REL?.

Limits:

The buffer depth determines the largest trace number or time stamp that you can specify (DISP:WIND:TRAC:BUFF). If larger numbers are sent, they are interpreted as the largest valid number and no error messages are generated. The buffer depth is limited by available memory.

Preset Value:

On preset, waterfall and spectrogram displays are turned off. A query issued immediately after a preset returns the buffer depth value, which is not affected by preset. For a *running* waterfall/spectrogram measurement:

- If the number of traces displayed hasn't reached the buffer depth, the value returned with a query is that of the most recent trace.

- If the number of traces displayed has reached the buffer depth, the value returned is the buffer depth.

Related Commands:

- To set the buffer depth, use DISP:WIND:TRAC:BUFF.
- To turn on the spectrogram/waterfall trace-analysis feature, use CALC:MARK:TRAC.

CALCulate[1 | 2 | 3 | 4]:MARKer:Z:RELative? query

Returns the Z-axis marker value (in seconds) relative to the offset marker position.

Query Syntax: `CALCulate[1|2|3|4]:MARKer:Z:RELative?`

Example Statements:
OUTPUT 719;":calc4:marker:z:rel?"
OUTPUT 719;"Calculate4:Mark:Z:Rel?"

Return Format: `Real`

Attribute Summary:
```
Option:  AYB (Waterfall and Spectrogram Analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command returns the value of the z-axis marker, relative to the offset marker, when the selected trace is a waterfall or spectrogram display.

Related Commands:

- To return the absolute marker value, use CALC:MARK:Z?.

CALCulate[1 | 2 | 3 | 4]:MARKer:Z:UNIT **command/query**

Sets the units for commands that query the z-axis marker in waterfall or spectrogram displays.

Command Syntax: `CALCulate[1|2|3|4]:MARKer:Z:UNIT S|UNITLESS`

Example Statements:
OUTPUT 719;"CALC4:MARKER:Z:UNIT s"
OUTPUT 719;"calc:mark:z:unit unitless"

Query Syntax: `CALCulate[1|2|3|4]:MARKer:Z:UNIT?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  AYB (Waterfall and Spectrogram Analysis)
Synchronization Required:  no
Preset State:  UNITLESS
SCPI Compliance:  instrument-specific
```

Description:

Sending CALC:MARK:Z:UNIT S sets the z-axis units to seconds, which is the trace’s time stamp. Sending CALC:MARK:Z:UNIT UNITLESS sets the z-axis units to a unitless number, which is the trace number.

This command determines the units returned with the CALC:MARK:Z? query.

CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5] command/query

Defines a complex constant in real,imaginary form.

Command Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:STEP <r>, <i>`

<r> ::= <number>|<bound>

<i> ::= <number>|<bound>

<number> ::= a real number (NRf data)
limits: -3.40282e+38:3.40282e+38

<bound> ::= MAX|MIN

Example Statements: OUTPUT 719;":CALCULATE:MATH:CONSTANT2 -2560, 19.84"
OUTPUT 719;"calc:math:cons3 7.53334e6, 2.45e3"

Query Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]?`

Return Format: `Real, Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

`<r>` represents the real part and `<i>` represents the imaginary part.

Complex constants may be defined with either real/imaginary or magnitude and phase values.

The numbers in the calculate node have no effect on this command.

CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:IMAG command/query

Defines the imaginary part of a math constant.

Command Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:IMAG <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -3.40282e+38:3.40282e+38
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":CALCULATE:MATH:CONSTANT2:IMAG -256"
OUTPUT 719;"calc:math:cons:imag 7.53334e6"
```

Query Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:IMAG?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

Complex constants may be defined with either real/imaginary or magnitude and phase values.

The numbers in the calculate node have no effect on this command.

- To specify step size for use with UP|DOWN, use CALC:MATH:CONS:STEP.

If the step size is zero (0), the UP|DOWN argument changes the value by incrementing/decrementing the third-most-significant digit. For example, a value of 6027 would be incremented to 6037 or decremented to 6017; a value of 53 would be incremented to 53.1 or decremented to 52.9. If you increment/decrement the value of the step size, it changes in the same manner.

CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:MAG command/query

Defines the magnitude value of a complex math constant.

Command Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:MAG <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0.0:3.40282e+38
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":CALCULATE:MATH:CONSTANT2:MAG 256"
OUTPUT 719;"calc:math:cons:mag 7.53334e6"
```

Query Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:MAG?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

Complex constants may be defined with either real/imaginary or magnitude and phase values.

The numbers in the calculate node have no effect on this command.

CALCulate[1 | 2 | 3 | 4]:MATH:CONStant[1 | 2 | 3 | 4 | 5]:PHASe command/query

Defines the phase value of a complex math constant.

Command Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:PHASe <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -3.40282e+38:3.40282e+38
<unit>	::=	DEG|RAD
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":CALCULATE:MATH:CONSTANT2:PHASE -2.56e3"
OUTPUT 719;"calc:math:cons:phas 7.53334e6"
```

Query Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:PHASe?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

Complex constants may be defined with either real/imaginary or magnitude and phase values.

The numbers in the calculate node have no effect on this command.

- To specify step size for use with UP|DOWN, use CALC:MATH:CONS:STEP.

If the step size is zero (0), the UP|DOWN argument changes the value by incrementing/decrementing the third-most-significant digit. For example, a value of 6027 would be incremented to 6037 or decremented to 6017; a value of 53 would be incremented to 53.1 or decremented to 52.9. If you increment/decrement the value of the step size, it changes in the same manner.

CALCulate[1 | 2 | 3 | 4]:MATH:CONStant[1 | 2 | 3 | 4 | 5]:REAL command/query

Defines the real part of a math constant.

Command Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:REAL <param>`

<param> ::= <number>|<step>|<bound>

<number> ::= a real number (NRf data)
limits: -3.40282e+38:3.40282e+38

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;":CALCULATE:MATH:CONSTANT2:REAL -2560"
OUTPUT 719;"calc:math:cons:real 7.53334e6"
```

Query Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:REAL?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

Complex constants may be defined with either real/imaginary or magnitude and phase values.

The numbers in the calculate node have no effect on this command.

- To specify step size for use with UP|DOWN, use CALC:MATH:CONS:STEP.

If the step size is zero (0), the UP|DOWN argument changes the value by incrementing/decrementing the third-most-significant digit. For example, a value of 6027 would be incremented to 6037 or decremented to 6017; a value of 53 would be incremented to 53.1 or decremented to 52.9. If you increment/decrement the value of the step size, it changes in the same manner.

CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:STEP command/query

Specifies the step size used to change math constant values with the UP|DOWN parameters.

Command Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:STEP <param>`

<param> ::= <number>|<step>|<bound>

<number> ::= a real number (NRf data)
limits: -3.40282e+38:3.40282e+38

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

OUTPUT 719;":CALCULATE:MATH:CONSTANT2:STEP -256"
OUTPUT 719;"calc:math:cons:step 7.53334e6"

Query Syntax: `CALCulate[1|2|3|4]:MATH:CONStant[1|2|3|4|5]:STEP?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

If the step size is zero (0), the UP|DOWN argument changes the value by incrementing/decrementing the third-most-significant digit. For example, a value of 6027 would be incremented to 6037 or decremented to 6017; a value of 53 would be incremented to 53.1 or decremented to 52.9. If you increment/decrement the value of the step size, it changes in the same manner.

Complex constants may be defined with either real/imaginary or magnitude and phase values.

The numbers in the calculate and constant nodes have no effect on this command.

CALCulate[1 | 2 | 3 | 4]:MATH[:EXPRession[1 | 2 | ... | 6]] command/query

Defines math functions.

Command Syntax: `CALCulate[1|2|3|4]:MATH[:EXPRession[1|2|...|6]] <EXPR>`

Example Statements:
OUTPUT 719;"CALCULATE:MATH:EXPR2 (SPEC2/D2)"
OUTPUT 719;"calc4:math:expr (ACORR1-K1)"
OUTPUT 719;"calc:math:expr3 (MEASTIME1_XC_D1)"

Query Syntax: `CALCulate[1|2|3|4]:MATH[:EXPRession[1|2|...|6]]?`

Return Format: `STRING`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

The numbers in the calculate node have no effect on this command. If no number is used in the expression node, 1 is assumed. Math expressions may be formed with the following:

- Measurement Data (for channel 1 or 2):
  - SPEC1 or SPEC2 (spectrums)
  - PSD1 or PSD2 (power spectral density)
  - TIME1 or TIME2 (main time)
  - GTIME1 or GTIME2 (gated time)
  - ACORR1 or ACORR2 (auto correlation)
  - ISPEC1 or ISPEC2 (instantaneous spectrum)

  If vector modulation analysis (option AYA) is installed, measurement data choices include:
  - MEASTIME (measured time)
  - MEASSPEC (measured spectrum)
  - REFTIME (reference time)
  - REFSPEC (reference spectrum)
  - MAGERR (magnitude error)
  - PHASEERR (phase error)
  - ERRTIME (error time)
  - ERRSPEC (error spectrum)

  If the second input channel is installed (option AY7), measurement data choices include:
  - FRES (frequency response)
  - COH (coherence)
  - XPOWER (cross spectrum)
  - XCORR (cross correlation)
- Data registers: D1, D2, D3, D4, D5, D6
- Math constants: K1, K2, K3, K4, K5
- Math functions: F1, F2, F3, F4, F5, F6
- $j\omega$
- Operations: CONJ, MAG, PHASE, REAL, IMAG, SQRT, FFT, IFFT, ln, exp
- Math operators: +, –, *, /, (,), _XC_ (cross correlation; see examples)

CALCulate[1 | 2 | 3 | 4]:MATH:SELect command/query

Specifies a user-defined math function to be displayed on the specified trace.

Command Syntax: `CALCulate[1|2|3|4]:MATH:SELect F1|F2|F3|F4|F5|F6`

Example Statements:
OUTPUT 719;":Calculate:Math:Sel F5"
OUTPUT 719;"CALCULATE3:MATH:SELECT F2"

Query Syntax: `CALCulate[1|2|3|4]:MATH:SELect?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  trace 1 is F1, 2 is F2, 3 is F3, 4 is F4
SCPI Compliance:  instrument-specific
```

Description:

Selecting a math function turns math on (CALC:MATH:STATE ON) if it is not already on.

Related Commands

- To define a math function, use CALC:MATH:EXPR.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1 | 2 | 3 | 4]:MATH:STATe command/query

Turns math on or off.

Command Syntax: `CALCulate[1|2|3|4]:MATH:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"calc3:math:state ON"
OUTPUT 719;"Calc:Math:Stat OFF"

Query Syntax: `CALCulate[1|2|3|4]:MATH:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

When math is turned on, the selected math function (CALC:MATH:SEL) is displayed for the specified trace (1, 2, 3, or 4). Must be ON for math to be performed.

Related Commands

- To define a function, use CALC:MATH:EXPR.
- To define a constant, use CALC:MATH:CONS.
- To display a math function, use CALC:MATH:SEL.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1|2|3|4]:STATe **command/query**

Turns postprocessing calculations on and off.

Command Syntax: `CALCulate[1|2|3|4]:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;":CALC:STATE ON"
OUTPUT 719;"calc:state ON"

Query Syntax: `CALCulate[1|2|3|4]:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  ON (all traces)
SCPI Compliance:  confirmed
```

Description:

When calculation is off, data postprocessing is not computed or displayed for the trace. This improves measurement speed. When calculation is on, data postprocessing is performed.

Data postprocessing is the computation of measurement data, it’s coordinate transformation, and display. When CALC:STATE is OFF, data for the specified trace is invalid for GPIB commands that try to access it (such as CALC:DATA? and TRAC:DATA?). All other CALC commands are received and processed but no data is computed unless CALC:STAT is ON.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1 | 2 | 3 | 4]:UNIT:AM — command/query

Specifies the default y-axis units for amplitude-modulated measurements.

Command Syntax: `CALCulate[1|2|3|4]:UNIT:AM <unit>`

<unit> ::= (see units list below)

Example Statements:
OUTPUT 719;"Calc:Unit:Am dBamrms2"
OUTPUT 719;"CALC:UNIT:AM amrms2/Hz"

Query Syntax: `CALCulate[1|2|3|4]:UNIT:AM?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  AM
SCPI Compliance:  instrument-specific
```

Units List

am	ampk/rtHz	amrms	amrms2/Hz	dBamrms2
am2	ampk2	amrms/rtHz	dBampk2	dBamrms2/Hz
ampk	ampk2/Hz	amrms2	dBampk2/Hz	pct

Description:

The UNIT subsystem is used to change default parameter units. The specified units apply to both the command and query. Default units are temporarily overridden by using other (compatible) units as a suffix to a command parameter, as when ampk is sent while the default is am.

The absolute value of *am* units represents the instantaneous modulation index where $-1 \le am \le +1$. The units may be linear or logarithmic (dB). Units with /rtHz or 2/Hz (squared-per-Hz) are PSD data. When am is used without an rms or pk designation, peak units are assumed, just as is done with volts.

Unit queries must be in the current “family” of units. That is, the CALC:UNIT:AM? query should be used only when the default Y-axis units are known to be one of the units in the AM units list.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1|2|3|4]:UNIT:ANGLe command/query

Specifies the y-axis units for phase-angle data format and phase deviation for PM demodulation.

Command Syntax: `CALCulate[1|2|3|4]:UNIT:ANGLe <unit>`

<unit> ::= (see units list below)

Example Statements:
OUTPUT 719;"Calc3:Unit:Angle deg"
OUTPUT 719;"CALC:UNIT:ANGL rad"

Query Syntax: `CALCulate[1|2|3|4]:UNIT:ANGLe?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  deg
SCPI Compliance:  confirmed
```

Units List

dBradpk2	dBradrms2/Hz	rad2	radpk2	radrms/rtHz
dBradpk2/Hz	deg	radpk	radpk2/Hz	radrms2
dBradrms2	rad	radpk/rtHz	radrms	radrms2/Hz

Description:

The UNIT subsystem is used to change default parameter units. The specified units apply to both the command and query. Default units are temporarily overridden by using other (compatible) units as a suffix to a command parameter, as when uV are sent while the default is V.

These units are phase angle. Degrees and radians are units for phase angle data format. All are associated with PM demodulation spectrums. Units with /rtHz or 2/Hz are PSD data.

Unit queries must be in the current "family" of units. For example, the CALC:UNIT:ANGLE? query should be used only when the default Y-axis units are known to be one of the units in the ANGLE units list.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, "CALC1" is assumed and the command affects trace A.

CALCulate[1|2|3|4]:UNIT:FREQuency command/query

Specifies the default y-axis units for frequency measurements.

Command Syntax: `CALCulate[1|2|3|4]:UNIT:FREQuency <unit>`

<unit> ::= (see units list below)

Example Statements:
OUTPUT 719;":calc4:unit:freq Hzpk"
OUTPUT 719;"Calculate4:Unit:Freq dBHzpk2"

Query Syntax: `CALCulate[1|2|3|4]:UNIT:FREQuency?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  HZ
SCPI Compliance:  instrument-specific
```

Units List

dBHzpk2	dBHzrms2/Hz	Hzpk	Hzpk2/Hz	Hzrms2
dBHzpk2/Hz	Hz	Hzpk/rtHz	Hzrms	Hzrms2/Hz
dBHzrms2	Hz2	Hzpk2	Hzrms/rtHz	

Description:

The UNIT subsystem is used to change default parameter units. The specified units apply to both the command and query. Default units are temporarily overridden by using other (compatible) units as a suffix to a command parameter, as when uV are sent while the default is V.

The Y-axis units are Hz when the instrument mode is FM demodulation.

- When the measurement data is (power) spectrum with log magnitude, the default units are $dBHz_{rms}^2$; with linear magnitude, they are Hz_{rms}.
- When the measurement data is main time, the default units are Hz (peak).
- Units with /rtHz or 2/Hz (squared-per-Hz) are PSD data.
- When Hz is used without a peak or rms designation, peak is assumed, as is done for volts.

Unit queries must be in the current "family" of units. That is, the CALC:UNIT:FREQ? query should be used only when the default Y-axis units are known to be one of the units in the FREQ units list.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, "CALC1" is assumed and the command affects trace A.

CALCulate[1|2|3|4]:UNIT:POWer command/query

Specifies the default y-axis units for power measurements.

Command Syntax: `CALCulate[1|2|3|4]:UNIT:POWer <unit>`

<unit> ::= (see units list below)

Example Statements:
OUTPUT 719;"CALCULATE:UNIT:POWER Vrms2"
OUTPUT 719;"calc:unit:power Wrms"

Query Syntax: `CALCulate[1|2|3|4]:UNIT:POWer?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  (see table in discussion)
SCPI Compliance:  confirmed
```

Units List

dB	dBVrms	V2/Hz	Vrms2
dBm	dBVrms/rtHz	Vpk	Vrms2/Hz
dBm/Hz	pct	Vpk/rtHz	W
dBV	unitless	Vpk2	W/Hz
dBV/rtHz	V	Vpk2/Hz	Wrms
dBVpk	V/rtHz	Vrms	Wrms/Hz
dBVpk/rtHz	V2	Vrms/rtHz	

Description:

The UNIT subsystem is used to change default parameter units. The specified units apply to both the command and query. Default units are temporarily overridden by using other (compatible) units as a suffix to a command parameter, as when dBVrms is sent while the default is V.

The Y-scale units are volts or power when the instrument mode is not demodulation.

- Power units (dBm or V^2) apply to (power) spectrum measurement data.
- Volt units apply to time measurement data.
- Units with /rtHz or 2/Hz (squared-per-Hz) are PSD measurement data.
- When V is used without a peak or rms designation, peak is assumed.

Unit queries must be in the current “family” of units. For example, the CALC:UNIT:POW? query should be used only when the default Y-axis units are known to be one of the units in the POWER units list.

Preset Power Units: Measurement Data vs. Data Format

Measurement Data Types	Data Formats: log magnitude	linear magnitude	real or imaginary
spectrum	dBm	Vrms	Vrms
PSD	dBm/Hz	Vrms/rtHz	Vrms/rtHz
time	dBVpk	V	V

CALCulate[1|2|3|4]:UNIT:TIME — command/query

Specifies the default y-axis units for time measurements.

Command Syntax: `CALCulate[1|2|3|4]:UNIT:TIME S`

Example Statements:
```
OUTPUT 719;":Calc2:Unit:Time S"
OUTPUT 719;"CALC3:UNIT:TIME S"
```

Query Syntax: `CALCulate[1|2|3|4]:UNIT:TIME?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  S (seconds)
SCPI Compliance:  instrument-specific
```

Description:

The UNIT subsystem is used to change default parameter units. The specified units apply to both the command and query. Default units are temporarily overridden by using other (compatible) units as a suffix to a command parameter, as when uV are sent while the default is V.

The Y-axis units are time when the data format is group delay. The list of time units consists of one entry; seconds. It is provided for consistency in program queries written to determine whether units of seconds, minutes, hours, days, weeks, ... are default.

Unit queries must be in the current "family" of units. For example, the CALC:UNIT:ANGLE? query should be used only when the default Y-axis units are known to be one of the units in the ANGLE units list.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, "CALC1" is assumed and the command affects trace A.

CALCulate[1|2|3|4]:UPHase:CREFerence command/query

Specifies the x-axis value about which the phase values are to be unwrapped.

Command Syntax: `CALCulate[1|2|3|4]:UPHase:CREFerence <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0.0:3.40282347E+38
<unit>	::=	HZ|S|CODE
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":calculate2:uph:cref 5E-6 HZ"
OUTPUT 719;"CALC:UPH:CREF 10E-6 S"
```

Query Syntax: `CALCulate[1|2|3|4]:UPHase:CREFerence?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Description:

This would typically be a well-known point in a measurement such as the passband of a band-pass filter.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1 | 2 | 3 | 4]:UPHase:OFFSet **command/query**

Specifies a phase offset to be applied to all unwrapped phase values.

Command Syntax: `CALCulate[1|2|3|4]:UPHase:OFFSet <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (range depends on configuration)
<unit>	::=	DEG|RAD
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"calculate3:uph:offs 0"
OUTPUT 719;"Calculate:Uph:Offset 0"
```

Query Syntax: `CALCulate[1|2|3|4]:UPHase:OFFSet?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Description:

Specifies a phase offset to be applied to all unwrapped phase values.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1|2|3|4]:X:UNIT:CODE **command/query**

Specifies the default x-axis units when the x axis is the code domain.

Command Syntax: `CALCulate[1|2|3|4]:X:UNIT:CODE CODE`

Example Statements:
OUTPUT 719;":CALC4:X:UNIT:CODE Code"
OUTPUT 719;"calculate4:x:unit:code Code"

Query Syntax: `CALCulate[1|2|3|4]:X:UNIT:CODE?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  B73 (Wideband CDMA Analysis)
Synchronization Required:  no
Preset State:  CODE
SCPI Compliance:  instrument-specific
```

Description:

The UNIT subsystem is used to change default parameter units. The specified units apply to both the command and query. The only units you can specify are units of CODE, which is the default.

CALCulate[1 | 2 | 3 | 4]:X:UNIT:FREQuency command/query

Specifies the default x-axis units when the x axis is the frequency domain.

Command Syntax: `CALCulate[1|2|3|4]:X:UNIT:FREQuency HZ`

Example Statements:
OUTPUT 719;"Calc4:X:Unit:Freq Hz"
OUTPUT 719;"CALCULATE:X:UNIT:FREQ Hz"

Query Syntax: `CALCulate[1|2|3|4]:X:UNIT:FREQuency?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  HZ
SCPI Compliance:  instrument-specific
```

Description:

The UNIT subsystem is used to change default parameter units. The specified units apply to both the command and query.

Unit queries must be in the current “family” of units. That is, the CALC:X:UNIT:FREQ? query should be used only when the default x-axis units are known to be one of the units in the FREQ units list.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, “CALC1" is assumed and the command affects trace A.

CALCulate[1|2|3|4]:X:UNIT:POW command/query

Specifies the default x-axis units when the x axis is the code domain.

Command Syntax: `CALCulate[1|2|3|4]:X:UNIT:POW DB`

Example Statements:

```
OUTPUT 719;":calc4:x:unit:pow Db"
OUTPUT 719;"Calc2:X:Unit:Pow Db"
```

Query Syntax: `CALCulate[1|2|3|4]:X:UNIT:POW?`

Return Format: `CHAR`

Attribute Summary:

```
Option:  B73 (Wideband CDMA Analysis)
Synchronization Required:  no
Preset State:  CODE
SCPI Compliance:  instrument-specific
```

Description:

The UNIT subsystem is used to change default parameter units. The specified units apply to both the command and query. The only units you can specify are DB, which are the default units.

CALCulate[1|2|3|4]:X:UNIT:TIME **command/query**

Specifies the default x-axis units when the x axis is the time domain.

Command Syntax: `CALCulate[1|2|3|4]:X:UNIT:TIME S|SYM|SLOT`

Example Statements:
OUTPUT 719;"CALCULATE4:X:UNIT:TIME s"
OUTPUT 719;"calc:x:unit:time sym"

Query Syntax: `CALCulate[1|2|3|4]:X:UNIT:TIME?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  HZ
SCPI Compliance:  instrument-specific
```

Description:

The UNIT subsystem is used to change default parameter units. The specified units apply to both the command and query.

S specifies units of seconds and **SYM** specifies units of symbols, as defined for digital modulation.

Unit queries must be in the current "family" of units. That is, the CALC:X:UNIT:TIME? query should be used only when the default x-axis units are known to be one of the units in the TIME units list.

The numbers 1 through 4 used with the CALCulate node specify which trace (A, B, C, or D) is to be affected by the command. If no number is used in the command line, "CALC1" is assumed and the command affects trace A.

The SLOT unit is only allowed in the W-CDMA instrument mode.

CALibration:AUTO — command/query

Enable or disable the analyzer's autocalibration function or perform a single calibration.

Command Syntax: `CALibration:AUTO OFF|0|ON|1|ONCE`

Example Statements:
OUTPUT 719;":Calibration:Auto OFF"
OUTPUT 719;"CALIBRATION:AUTO OFF"

Query Syntax: `CALibration:AUTO?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

When calibration occurs, all amplitude ranges and all frequencies are calibrated regardless of instrument setup. The current value is stored in non-volatile RAM and is not affected by preset.

When autocal is enabled, the analyzer automatically calibrates several times during the first two hours of operation. After that, it automatically calibrates at intervals of 2 hours 20 minutes. When autocal is disabled, the analyzer calibrates only when CAL:AUTO ONCE is used.

If calibration occurs while an averaged measurement is paused, then CONTinue has the same effect as ABORt; the averaging process is restarted (average counter set to zero).

The autocalibration state value is stored in non-volatile memory and is not affected by preset.

Note During calibration, a small ac voltage (approximately 2 mV) appears at the source output connector.

CALibration:ZERO:AUTO **command/query**

Controls the auto-zero calibration function.

Command Syntax: `CALibration:ZERO:AUTO OFF|0|ON|1|ONCE`

Example Statements:
OUTPUT 719;"cal:zero:auto ON"
OUTPUT 719;"Calibration:Zero:Auto OFF"

Query Syntax: `CALibration:ZERO:AUTO?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1
SCPI Compliance:  instrument-specific
```

Description:

Auto zero corrects for residual DC in the input amplifiers. If zeroing is not performed, the DC value adds to the input signal giving incorrect measurement values and (worst case) may become large enough to cause ADC overloads or incorrect triggering. Residual DC is the calibration element most likely to change with time or operating temperature.

The differences between calibration and auto zeroing are:

- Auto zeroing takes much less time for the analyzer to perform than a complete calibration requires.
- Auto zeroing has less effect on measurement accuracy than calibration.

The auto-zero feature is a part of calibration that may be activated when automatic calibration is off. If calibration is on, zeroing cannot be turned off. Turning auto zero off allows you to control the analyzer without interruptions caused by automatically-initiated operations. If you intend to run a series of measurements with auto zero turned off, you may want to turn auto zero off and then perform CAL:ZERO:AUTO ONCE just before beginning.

Note Selecting ONCE zeros the input and then turns auto zero off. Single auto-zero calibrations should be performed every 30 minutes or whenever there is a significant change in the analyzer's operating temperature.

CONTinue command

Continue a paused measurement.

Command Syntax: `CONTinue`

Example Statements:
OUTPUT 719;":CONTINUE"
OUTPUT 719;"cont"

Attribute Summary:
```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Characteristics:

- Continues a paused measurement.
- Continues a completed averaged measurement allowing an additional N (number of) averages (measurement scans) to be acquired.
- CONT is ignored if measurement is not paused.
- Restarts data acquisition stopped by PAUSE.

DISPlay:ANNotation[:ALL] — command/query

Turns display annotation on and off.

Command Syntax: `DISPlay:ANNotation[:ALL] OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Display:Ann:All ON"
OUTPUT 719;"DISPLAY:ANN OFF"

Query Syntax: `DISPlay:ANNotation[:ALL]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +1 (ON)
SCPI Compliance:  confirmed
```

Description:

When display annotation is off, no frequency/time (below grid) or marker (above grid) annotation is displayed; nor does it appear in plots.

Annotation that is not affected by this command includes amplitude scale information, trace title information, measurement status, average count, and overload information.

- To blank the entire display, use DISP:ENAB OFF.

DISPlay:BRIGhtness command/query

Specifies the display brightness.

Command Syntax: `DISPlay:BRIGhtness {<number>[<unit>]}|<step>|<bound>`

<number>	::=	a real number (NRf data) limits: 20:100
<unit>	::=	[PCT]
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;":display:brig 40.8674"
OUTPUT 719;"Disp:Brightness 44.2263"

Query Syntax: `DISPlay:BRIGhtness?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Controls the intensity of the display. The range of the parameter is 20 to 100, where 100 is full intensity and 20 is dim.

- To blank the entire display, use DISP:ENAB OFF.

DISPlay:CMAP:COLor[1|2|...|256]:HSL **command/query**

Specifies the hue, saturation, and luminosity of the colors.

Command Syntax:

```
DISPlay:CMAP:COLor:HSL [1|2|...|256] <hue>, <sat>, <lum>
```

<hue> ::= <number>|<step>|<bound>

<sat> ::= <number>|<step>|<bound>

<lum> ::= <number>|<step>|<bound>

<number> ::= a real number (NRf data)
limits: 0.0:1.0

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;":disp:cmap:col1:hsl 0, 0, 0"
OUTPUT 719;"Disp:Cmap:Col3:hsl .25, .85, .4"
```

Query Syntax:

```
DISPlay:CMAP:COLor[1|2|...|256]:HSL?
```

Return Format:

```
Real, Real, Real
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not documented
SCPI Compliance:  confirmed
```

Description:

The CMAP subsystem controls the physical color associated with each logical color number in the color map of the display. The numbers 1 through 256 correspond to various display elements as described in Help Text.

The HSL command sets the instrument's color map based on the Hue/Saturation/Luminance levels color model for the user-defined color maps. Before issuing color changes with this command, select one of the two user-defined color maps with the DISP:WIND:SPEC:MAP command.

Hue ranges from zero to one, circularly, with a value of zero resulting in the same hue as a value of one. The approximate color progression is (starting at zero): red, orange, yellow, green, cyan, blue, magenta, and back to red.

Saturation is the amount of pure color versus white. The saturation value ranges from zero to one, with zero specifying no color (all white or gray, depending on brightness*) and one specifying no white.

Luminance specifies the brightness per unit area of the color. A luminance of zero results in black; a luminance of one results in the brightest color available.

* See the DISP:BRIG command.

DISPlay:CMAP:DEFault **command**

Returns all color settings to their default settings.

Command Syntax: `DISPlay:CMAP:DEFault`

Example Statements:
OUTPUT 719;"DISP:CMAP:DEF"
OUTPUT 719;"disp:cmap:def"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The display color map is loaded with predefined color definitions.

DISPlay:ENABle command/query

Turns the display on and off.

Command Syntax: `DISPlay:ENABle OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Display:Enab ON"
OUTPUT 719;"DISP:ENABLE OFF"

Query Syntax: `DISPlay:ENABle?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +1 (ON)
SCPI Compliance:  confirmed
```

Description:

The message ‘‘Display Blanking On’’ appears in the center of the display when the display is off. Plots that occur when the display is off also show only the blanking message.

Menu information appears only when analyzer is in local (front-panel) operation. Issuing any remote command blanks the softkey menus.

DISPlay:FORMat command/query

Specifies the number of grids on which to display traces.

Command Syntax: `DISPlay:FORMat SINGle|TWO|FOUR|QUAD`

Example Statements: OUTPUT 719;"disp:format FOUR"
OUTPUT 719;"Disp:Form TWO"

Query Syntax: `DISPlay:FORMat?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  SING
SCPI Compliance:  instrument-specific
```

Description:

When more than one grid is turned on, the same number of traces are also turned on. More than one trace may be displayed on a grid in SINGLE or TWO grid formats.

The selections SINGLE, TWO, and FOUR are full screen-width grids. QUAD displays four grids in a quadrant arrangement in which the traces are a half screen-width wide with two grids on the right and two on the left.

Traces are turned on and off with DISP:WIND:TRAC[:STAT].

DISPlay:MFUNction command/query

Turns display of the math function definitions on and off.

Command Syntax: `DISPlay:MFUNction OFF|0|ON|1`

Example Statements:
OUTPUT 719;":DISPLAY:MFUN ON"
OUTPUT 719;"display:mfun ON"

Query Syntax: `DISPlay:MFUNction?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

DISPlay:PROGram[:MODE] command/query

Selects the portion of the analyzers screen to be used for Instrument BASIC program output.

Command Syntax: `DISPlay:PROGram[:MODE] OFF|0|FULL|UPPer|LOWer`

Example Statements:

```
OUTPUT 719;"Disp:Program:Mode FULL"
OUTPUT 719;"DISPLAY:PROG LOWER"
```

Query Syntax: `DISPlay:PROGram[:MODE]?`

Return Format: `CHAR`

Attribute Summary:

```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  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:TCAPture:ENVelope — command/query

Turns the time capture envelope (detector) function on/off.

Command Syntax: `DISPlay:TCAPture:ENVelope OFF|0|ON|1`

Example Statements:
OUTPUT 719;":disp:tcapture:env ON"
OUTPUT 719;"Display:Tcap:Env OFF"

Query Syntax: `DISPlay:TCAPture:ENVelope?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1 (ON)
SCPI Compliance:  instrument-specific
```

Description:

The time capture buffer usually holds many more points than can be displayed simultaneously with CALC:FEED ‘TCAP1’, so some are not displayed. This command determines whether every n^{th} point is displayed (OFF) or points of maximum magnitude (taken n-at-a-time) are displayed (ON). In this discussion, n is the ratio of points in the capture buffer to the maximum number of frequency points defined with MEM:MALL:MEAS:FPO, which is (`MaxNumFreqPts` – 1)×2.56 for baseband and (`MaxNumFreqPts` – 1)×1.28 for zoom.

The envelope (ON) shows transients. To view periodic waveforms, you may want to turn the envelope OFF.

The numbers 1 and 2 used with the TCAPture node specify which input channel (1 or 2) is to be affected by the command. If no number appears in the node, “TCAP1" is assumed and the command affects channel 1.

DISPlay[:WINDow[1|2|3|4]]:ACTive command/query

Controls active state of the four traces.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:ACTive OFF|0|ON|1|ONCE`

Example Statements:
OUTPUT 719;"DISPLAY:WIND4:ACTIVE OFF"
OUTPUT 719;"disp:act ON"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:ACTive?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1 ON; 2, 3, and 4 OFF
SCPI Compliance:  instrument-specific
```

Description:

Windows 1–4 correspond to traces A, B, C, D. If no number is specified in the WINDOW node or the node is not included, the command will affect trace A.

If more than one trace is displayed (DISP:FORM is TWO or FOUR, or more than one DISP:WIND:TRAC:STAT is ON) this command specifies which are **active**.

ON activates the trace corresponding to the window specified. More than one trace may be active at the same time. If only one trace is displayed, it is (by definition) active.

OFF deactivates the trace corresponding to the window specified. At least one trace is always active.

ONCE activates the trace corresponding to the window specified *and* sets all others to ACTIVE OFF.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:SPECtrogram:COLors command/query

Specifies the number of colors in the spectrogram display.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:SPECtrogram:COLors <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 2:64
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;":Display:Spec:Colors 20"
OUTPUT 719;"DISP:WIND2:SPECTROGRAM:COL 43"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:SPECtrogram:COLors?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command affects *all* active traces regardless of whether a specific window/trace is used (e.g. DISP:WIND2).

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:SPECtrogram:ENHance command/query

Specifies the color mapping for image enhancement purposes.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:SPECtrogram:ENHance <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:100
<unit>	::=	PCT
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;"display:wind3:spec:enhance 71"
OUTPUT 719;"Disp:Spectrogram:Enh 56"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:SPECtrogram:ENHance?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  +50
SCPI Compliance:  instrument-specific
```

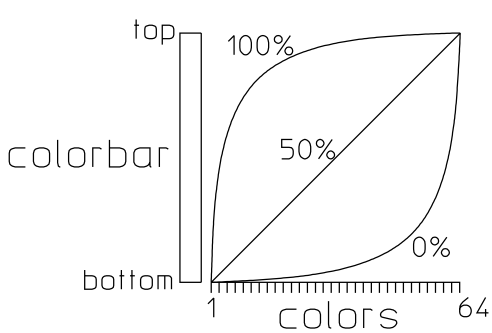
Description:

This command affects *all* active traces regardless of whether a specific window/trace is used (e.g. DISP:WIND2).

The number specified with this command determines the distribution of colors along the spectrogram colorbar as a percentage. See illustration.

50% defines a straight-line distribution of colors along the colorbar such that all colors have the same range on the colorbar.

0% defines a distribution curve that exponentially compresses the colors into the bottom of the colorbar and 100% compresses the colors into the top of the colorbar.



For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:SPECtrogram:MAP command/query

Specifies the color map for all active spectrogram displays.

Command Syntax: `DISPlay[:WINDow]:SPECtrogram:MAP <param>`

<param> ::= COLor|RCOLor|GREY|RGRey|MAP1|MAP2

Example Statements:
OUTPUT 719;":DISP:SPECTROGRAM:MAP MAP1"
OUTPUT 719;"display:wind:spec:map GREY"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:SPECtrogram:MAP?`

Return Format: `CHAR`

Attribute Summary:

```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command affects *all* active traces regardless of whether a specific window/trace is used (e.g. DISP:WIND2).

The color map determines the colors that may be used in the spectrogram display. A color map consists of 64 colors, all of which may or may not be used, as specified by DISP:SPEC:COL command.

COLOR is a map of 64 colors. This map has a maximum amplitude hue of 0% (red) and a minimum amplitude hue of 70% (blue). The hue of the other 62 colors are linearly distributed between 0% and 70%. The saturation and luminosity remain constant for all 64 colors.

RCOLOR is identical to COLOR except that the colors are reversed. The maximum amplitude hue is 70% and the minimum amplitude hue is 0%.

GREY is a color map of 64 shades of grey. The maximum amplitude has a luminosity of 100% (lightest shade of grey) and the minimum amplitude has a luminosity of 10% (darkest shade of grey). The other 62 colors' luminosity are linearly distributed between 10% and 100%. The hue and saturation remain constant for all 64 colors.

RGREY is identical to GREY except that the shades of grey are reversed.

MAP1 and **MAP2** are used to specify a user-defined color map. These maps may be defined with the DISP:CMAP:COLOR:HSL command for colors numbered 129 through 192. If you want to define MAP1, select it with this command, then send the color:hsl commands to define the 64 colors. These definitions are stored in non-volatile RAM.

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:SPECtrogram[:STATe] command/query

Turns spectrogram display on or off.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:SPECtrogram[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Disp:Window3:Spec:Stat ON"
OUTPUT 719;"DISPLAY:SPEC ON"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:SPECtrogram[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  0 (off)
SCPI Compliance:  instrument-specific
```

Description:

The number used in the WINDow node of the command specifies which trace becomes a spectrogram display. If no number is specified, “1" is assumed.

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator’s Guide*.

DISPlay[:WINDow[1|2|3|4]]:SPECtrogram:THReshold command/query

Sets the threshold for all active spectrogram displays.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:SPECtrogram:THReshold <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:100
<unit>	::=	PCT
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":display:spec:thr 18"
OUTPUT 719;"Display:Wind4:Spectrogram:Thr 94"
```

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:SPECtrogram:THReshold?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  0 (%)
SCPI Compliance:  instrument-specific
```

Description:

This command is useful for removing noise-floor clutter from the spectrogram display. The percentage reference is the height of the colorbar such that, with a threshold of 50%, the analyzer displays only the colors in the upper half of the colorbar. Trace signals below the threshold are displayed in the color at the bottom of the color bar.

This command affects *all* active traces regardless of whether a specific window/trace is used (e.g. DISP:WIND2).

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1|2|3|4]]:TRACe:BUFFer command/query

Specifies the number of waterfall/spectrogram traces stored in memory.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:BUFFer <param>`

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

Example Statements:

```
OUTPUT 719;"DISP:WINDOW:TRAC:BUFFER 0"
OUTPUT 719;"disp:trac:buffer 0"
```

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:BUFFer?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

The number of traces specified with this command are stored in measurement memory such that the measurement can be paused and the marker(s) used to make measurements on the stored traces. The number of traces buffered is not the same as the number of traces displayed which is determined by the vertical height of the grid and the elevation between traces.

Limits:

The maximum number of buffered traces is limited only by the amount of available memory. Press the memory useage softkey under the **System Utility** hardkey for a listing.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:DCARrier command/query

Turns the demodulate-carrier function on/off.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:DCARrier OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Disp:Trace:Dcar OFF"
OUTPUT 719;"DISP:WINDOW:TRAC:DCARRIER ON"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:DCARrier?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

For a description of this function, see Help Text under the Help hardkey.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:EYE:COUNt command/query

Specifies the length of the eye and trellis diagrams.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:EYE:COUNt <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0.1:40

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements: OUTPUT 719;"disp:wind4:trace:eye:count 35"
OUTPUT 719;"Disp:Trac:Eye:Coun 30"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:EYE:COUNt?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  +2
SCPI Compliance:  instrument-specific
```

Description:

For more information on digital demodulation, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1|2|3|4]]:TRACe:GRATicule:GRID[:STATe] command/query

Turns the trace grids on or off.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:GRATicule:GRID[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;":DISPLAY:TRAC:GRAT:GRID OFF"
OUTPUT 719;"disp:window3:trac:grat:grid:stat ON"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:GRATicule:GRID[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +1 (ON)
SCPI Compliance:  confirmed
```

Description:

When grids are off they are neither displayed nor plotted.

The numbers in the Window node have no effect on this command; *all* grids are turned on/off regardless of the window specified.

DISPlay[:WINDow[1|2|3|4]]:TRACe:INDicator command/query

Determines the character used to identify ideal states in a vector diagram when Digital Demodulation is selected.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:INDicator CROSs|CIRCle`

Example Statements:
OUTPUT 719;"Display:Wind2:Trac:Indicator CIRCLE"
OUTPUT 719;"DISP:TRACE:IND CROSS"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:INDicator?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  Cross
SCPI Compliance:  instrument-specific
```

Description:

This command affects all active traces regardless of whether a specific window/trace is used (e.g. DISP:WIND2).

This command lets you select a cross-hair or circle to represent the ideal states (ideal symbol locations) in a vector diagram. The size of the circle or cross-hair corresponds to some percentage of Error Vector Magnitude (EVM), and is set with DISP:WIND:TRAC:IND:SIZE.

For additional details, see online help for the [ideal state] softkey.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:INDicator:SIZE command/query

Determines the size, as a percentage of EVM, of ideal states in vector diagrams.

Command Syntax:

```
DISPlay[:WINDow[1|2|3|4]]:TRACe:INDicator:SIZE
{<number>[<unit>]}|<step>|<bound>
```

<number> ::= a real number (NRf data)
limits: 0.1:50

<unit> ::= [PCT]

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

OUTPUT 719;":disp:trace:ind:size 6"
OUTPUT 719;"Disp:Wind:Trace:Ind:Size 24"

Query Syntax:

```
DISPlay[:WINDow[1|2|3|4]]:TRACe:INDicator:SIZE?
```

Return Format:

```
Integer
```

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  +15.0
SCPI Compliance:  instrument-specific
```

Description:

This command affects all active traces regardless of whether a specific window/trace is used (e.g. DISP:WIND2).

This command determines the size of the cross-hairs or circles used to indicate ideal states in vector diagrams when Digital Demodulation is selected.

The size is specified as a percentage of Error Vector Magnitude (EVM), and determines the radius of the circle or cross-hair. For example, if you specify a size of 15%, the radius of the circle or cross-hair represents and EVM of 15%.

For additional details, see online help for the [state size] softkey.

DISPlay[:WINDow[1|2|3|4]]:TRACe:INFO command/query

Specifies user-defined alphanumeric content for trace information.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:INFO <STRING>`

Example Statements:

OUTPUT 719;":Disp:Trac:Info ‘Prototype 11, platform A7, test 53’"
OUTPUT 719;"DISP:WINDOW4:TRAC:INFO ‘3rd IF, repeater site 4’"
OUTPUT 719;"DISP:WINDOW4:TRAC:INFO ‘’"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:INFO?`

Return Format: `STRING`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This information appears on the same line as the trace title, at the right side of the display area. Maximum string size is 30 characters.

To remove this information from the screen, send a null string as shown in the third example above.

Related Commands:

- To define a trace title, use DISP:TRAC:LAB:USER.
- To switch between default and user-defined titles, use DISP:TRAC:LAB:AUTO.
- To query the currently displayed trace title, use DISP:TRAC:LAB.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:LABel? query

Returns the current trace title.DISPlay[[:WINDow]]:TRACe:LABel?[display:window:trace:label?]

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:LABel?`

Example Statements: OUTPUT 719;"DISP:WIND2:TRACE:LAB?"
OUTPUT 719;"display:trac:lab?"

Return Format: `STRING`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If the default trace title is ON, this query returns the default trace title. If the default trace title is OFF, this query returns the user-defined trace title.

Related Commands:

- To enter a user-defined trace title, use DISP:WIND:TRAC:LAB:USER.
- To display a user-defined trace title, use DISP:WIND:TRAC:LAB:AUTO OFF.

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

DISPlay[:WINDow[1|2|3|4]]:TRACe:LABel:AUTO command/query

Switches between displaying the default trace title and a user-defined trace title.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:LABel:AUTO OFF|0|ON|1`

Example Statements:

```
OUTPUT 719;":Display:Trac:Label:Auto OFF"
OUTPUT 719;"DISP:WINDOW:TRAC:LABEL:AUTO ON"
```

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:LABel:AUTO?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

ON displays the default trace title. OFF displays the user-defined trace title.

Related Commands:

- To define a trace title, use DISP:TRAC:LAB:USER.
- To query the currently displayed trace title, use DISP:TRAC:LAB?.

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, "WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, "1" is assumed and the command affects trace A.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:LABel:USER — command/query

Specifies the alphanumeric content of a user-defined display title.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:LABel:USER <STRING>`

Example Statements:
OUTPUT 719;"Disp:Window2:Trac:Lab:User ‘Phase Noise’"
OUTPUT 719;"DISP:TRAC:LAB:USER ‘Friday Data’"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:LABel:USER?`

Return Format: `STRING`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The alpha-numeric string replaces the default titles above the trace when DISP:TRAC:LAB:AUTO is OFF. Sending the user-defined trace label command automatically turns DISP:TRAC:LAB:AUTO OFF. Maximum string length is 15 characters.

The query form returns the user-defined trace title, whether or not it is currently displayed.

Related Commands:

- To switch between the default trace title and a user-defined trace title, use DISP:TRAC:LAB:AUTO.
- To query the currently-displayed trace title, use DISP:TRAC:LAB?.

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

DISPlay[:WINDow[1|2|3|4]]:TRACe[:STATe] command/query

Turns specific traces on and off.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;"disp:window:trac:state ON"
OUTPUT 719;"Disp:Trac OFF"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1 is ON; 2, 3, 4, are OFF
SCPI Compliance:  confirmed
```

Description:

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

Related Commands:

- To specify the number of grids on which the traces are displayed, use DISP:FORM.
- To control whether graticules are displayed, use DISP:TRAC:GRAT:GRID.

DISPlay[:WINDow[1|2|3|4]]:TRACe:SYMBol **command/query**

Specifies the use of either dots or bars to indicate symbol points on a time-domain display.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:SYMBol DOTS|BARS|OFF|0`

Example Statements:
OUTPUT 719;":DISPLAY:TRAC:SYMBOL OFF"
OUTPUT 719;"disp:wind:trace:symb DOTS"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:SYMBol?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  0 (off)
SCPI Compliance:  instrument-specific
```

Description:

For more information on digital demodulation, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:SYMBol:FORMat command/query

Determines the data format (hexadecimal or binary) for symbol tables.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:SYMBol:FORMat BIN|HEX`

Example Statements:
OUTPUT 719;"Display:Wind3:Trac:Symbol:Form HEX"
OUTPUT 719;"DISPLAY:TRAC:SYMB:FORMAT BIN"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:SYMBol:FORMat?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  BINary
SCPI Compliance:  instrument-specific
```

Description:

This command affects all active traces regardless of whether a specific window/trace is used (e.g. DISP:WIND2).

This command lets you choose the data format for symbol tables. You can select between hexadecimal and binary data formats. The default is binary.

DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:AUTO command/query

Controls automatic scaling of the X-axis.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:AUTO OFF|0|ON|1|ONCE`

Example Statements:
OUTPUT 719;":disp:trace:x:auto OFF"
OUTPUT 719;"Display:Wind2:Trace:X:Scal:Auto ONCE"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:AUTO?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1 (ON)
SCPI Compliance:  confirmed
```

Description:

X-axis autoscaling is similar to Y-axis scaling if you understand that the conventional method of displaying measurement information maps the first measurement data to the left-most X-axis value and the last measurement data to the right-most X-axis value. This corresponds to X-axis autoscaling being ON. When it is OFF, other time or frequency values may be mapped to these X-axis points, as defined by DISP:WIND:TRAC:X:LEFT and RIGHt, respectively.

ON changes the X scale to be the full range of the current data. Sending ON corresponds to pressing the softkey *X full scale* (under Ref Lvl/Scale, X scale markers keys). When the trace displays frequency information, for example, this corresponds to the start and stop frequencies being mapped to the left-most and right-most values on the X axis. This is the conventional method of displaying frequency domain data.

OFF corresponds to pressing the softkey *hold scale* (under Ref Lvl/Scale, X scale markers keys). This allows the display settings given by the DISP:WIND:TRAC:X:LEFT and RIGHt commands to be the left-most and right-most values on the X axis. When this is the active display state, the initial and last display values are not necessarily at the limits of the measurements.

ONCE has no corresponding softkey. It changes the values of the current measurement values for the left-most and right-most data values to the current values of the LEFT and RIGHT settings and then turns AUTO OFF.

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:X[:SCALe]:LEFT command/query

Specifies the first X-axis value on the display.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:LEFT <param>`

<param>	::=	<number>[<unit>]|<bound>
<number>	::=	a real number (NRf data) limits: -3.40282347E+38:3.40282347E+38
<unit>	::=	Hz|s|Code
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;":DISP:WIND4:TRACE:X:LEFT 10us; RIGH 15us"
OUTPUT 719;"disp:trace:x:scale:left 2.5025 MHz; right 2.5026 MHz"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:LEFT?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  25% of full scale
SCPI Compliance:  confirmed
```

Description:

This command specifies the value of the first (left-most) X-axis point on the display. This, combined with the RIGHT command, is equivalent to the X-Scale Markers feature in front-panel operation (under the Ref Lvl/Scale hardkey) which allows you to expand or magnify a narrow region of data. No new data is taken; existing data is displayed differently.

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

Related Commands:

- To reset the X-axis scale to full scale, use DISP:WIND:TRAC:X[:SCALE]:AUTO ONCE.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:X[:SCALe]:RIGHt command/query

Specifies the last X-axis value on the display.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:RIGHt`

<param> ::= <number>[<unit>]|<bound>

<number> ::= a real number (NRf data)
limits: -3.40282347E+38:3.40282347E+38

<unit> ::= Hz|s|Code

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;":DISP:WIND4:TRACE:X:LEFT 10us; RIGH 15us"
OUTPUT 719;"disp:trace:x:scale:left 2.5025 MHz; right 2.5026 MHz"
```

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:RIGHt?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  75% of full scale
SCPI Compliance:  confirmed
```

Description:

This command specifies the value of the last (most right) X-axis point on the display. This, combined with the LEFT command, is equivalent to the X-Scale Markers feature in front-panel operation (under the Ref Lvl/Scale hardkey) which allows you to expand or magnify a narrow region of data. No new data is taken; existing data is displayed differently.

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:X[:SCALe]:RLEVel command/query

Specifies the reference level value for the X axis or returns current setting.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:RLEVel <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: depends on configuration
<unit>	::=	(see listing below)
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;"DISP:WINDOW3:TRAC:X:SCALE:RLEV 0"
OUTPUT 719;"display:trac:x:rlevel 0"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:RLEVel?`

Return Format: `Real`

Attribute Summary:
```
Option:  AYA (vector analysis/digital demod)
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Units List

am	dBampk2/Hz	dBradrms2/Hz	Hzpk2	radpk2	Vpk
am2	dBamrms2	dBV	Hzpk2/Hz	radpk2/Hz	Vpk/rtHz
ampk	dBamrms2/Hz	dBV/rtHz	Hzrms	radrms	Vpk2
ampk/rtHz	dBHzpk2	dBVpk	Hzrms/rtHz	radrms/rtHz	Vpk2/Hz
ampk2	dBHzpk2/Hz	dBVpk/rtHz	Hzrms2	radrms2	Vrms
ampk2/Hz	dbHzrms2	dBVrms	Hzrms2/Hz	radrms2/Hz	Vrms/rtHz
amrms	dBHzrms2/Hz	dBVrms/rtHz	MHz	s	Vrms2
amrms/rtHz	dBm	deg	pct	unitless	Vrms2/Hz
amrms2	dBm/Hz	Hz	rad	V	W
amrms2/Hz	dBradpk2	Hz2	rad2	V/rtHz	W/Hz
dB	dBradpk2/Hz	Hzpk	radpk	V2	Wrms
dBampk2	dBradrms2	Hzpk/rtHz	radpk/rtHz	V2/Hz	Wrms/Hz

Description:

This command is effective only when CALC:FORM is COMPlex or CONStellation.

DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:SPACing command/query

Specifies the frequency axis of the active display to be either a linear or logarithmic format.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:SPACing LIN|LOG`

Example Statements:
OUTPUT 719;":Disp:Trace:X:Spac LINEAR"
OUTPUT 719;"DISPLAY:WIND:TRACE:X:SCAL:SPACING LINEAR"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:X[:SCALe]:SPACing?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  LIN
SCPI Compliance:  instrument-specific
```

Description:

Note that if two traces are displayed and one has a linear frequency axis and the other a logarithmic, marker coupling is still valid; the markers track the frequency value.

When a logarithmic x-axis is selected, frequency annotation is done only in terms of start and stop frequency, never in terms of span and center frequency.

DISPlay[:WINDow[1|2|3|4]]:TRACe:Y:RLINe command/query

Turns the reference line on and off.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y:RLINe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"disp:window2:trac:y:rline ON"
OUTPUT 719;"Disp:Trace:Y:Rlin OFF"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y:RLINe?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  confirmed
```

Description:

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

Related Commands:

- To select trace types (such as log magnitude, linear magnitude, phase, or group delay), use CALC:FORM.
- To specify the position of the reference line, use DISP[:WIND]:TRAC:Y:RPOS.
- To specify the reference level (value represented by the reference line), use DISP:WIND:TRAC:Y:RLEV.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:Y[:SCALe]:AUTO command/query

Performs a single, Y-axis autoscale on the specified trace.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:AUTO OFF|0|ONCE`

Example Statements:
OUTPUT 719;":DISPLAY:TRAC:Y:AUTO OFF"
OUTPUT 719;"disp:window3:trac:y:scal:auto OFF"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:AUTO?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  confirmed
```

Description:

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

Executing this command changes the Y scale to “fixed” (range tracking is turned off).

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:Y[:SCALe]:PDIVision command/query

Specifies the vertical scale value per division for the specified trace.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:PDIVision <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:3.40282347E+38
<unit>	::=	(see listing below)
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;"Display:Wind3:Trace:Y:Scal:Pdivision 0"
OUTPUT 719;"DISP:TRACE:Y:PDIV 0"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:PDIVision?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  10(dB/div)*
SCPI Compliance: confirmed
```

Units List

am	amrms2	Hzpk2	rad	radrms2	Vpk
am2	amrms2/Hz	Hzpk2/Hz	rad2	radrms2/Hz	Vpk/rtHz
ampk	dB	Hzrms	radpk	S	Vpk2
ampk/rtHz	deg	Hzrms/rtHz	radpk/rtHz	unitless	Vpk2/Hz
ampk2	Hz	Hzrms2	radpk2	V	Vrms
ampk2/Hz	Hz2	Hzrms2/Hz	radpk2/Hz	V/rtHz	Vrms/rtHz
amrms	Hzpk	MHz	radrms	V2	Vrms2
amrms/rtHz	Hzpk/rtHz	pct	radrms/rtHz	V2/Hz	Vrms2/Hz

Description:

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

Related Commands:

- To set the reference level, use DISP[:WIND]:TRAC:Y:RLEV.
- To change the position of the reference line, use DISP[:WIND]:TRAC:Y:RPOS.
- To autoscale the display, use DISP[:WIND]:TRAC:Y:AUTO.

* This is the preset value for log mag, spectrum. Other combinations of data format and measurement data have other preset values.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:TRACe:Y[:SCALe]:RLEVel **command/query**

Specifies the reference level value for the Y axis or returns current setting.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:RLEVel <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -3.40282347E+38:+3.40282347E+38
<unit>	::=	(see listing below)
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;"display:wind2:trac:y:scal:rlevel 0 dBm"
OUTPUT 719;"Disp:Wind:Trace:Y:Rlevel: -5E-003 VRMS"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:RLEVel?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (dBm)
SCPI Compliance:  confirmed
```

Units List

am	dBampk2/Hz	dBradrms2/Hz	Hzpk2	radpk2	Vpk
am2	dBamrms2	dBV	Hzpk2/Hz	radpk2/Hz	Vpk/rtHz
ampk	dBamrms2/Hz	dBV/rtHz	Hzrms	radrms	Vpk2
ampk/rtHz	dBHzpk2	dBVpk	Hzrms/rtHz	radrms/rtHz	Vpk2/Hz
ampk2	dBHzpk2/Hz	dBVpk/rtHz	Hzrms2	radrms2	Vrms
ampk2/Hz	dbHzrms2	dBVrms	Hzrms2/Hz	radrms2/Hz	Vrms/rtHz
amrms	dBHzrms2/Hz	dBVrms/rtHz	MHz	s	Vrms2
amrms/rtHz	dBm	deg	pct	unitless	Vrms2/Hz
amrms2	dBm/Hz	Hz	rad	V	W
amrms2/Hz	dBradpk2	Hz2	rad2	V/rtHz	W/Hz
dB	dBradpk2/Hz	Hzpk	radpk	V2	Wrms
dBampk2	dBradrms2	Hzpk/rtHz	radpk/rtHz	V2/Hz	Wrms/Hz

Description:

Sending units with this command does not change the Y-axis units displayed.

- To change the Y-axis units displayed, use the CALC:UNIT commands.
- To display the reference line (default = off), use DISP[:WIND]:TRAC:Y:RLIN.
- If range tracking is on (active), then changing the range setting changes the reference level. To change the range tracking status, use DISP:WIND:TRAC:Y:RLEV:AUTO.
- If range tracking is on *and* autoranging is on, then the reference level tracks the range and the range may change automatically, depending on the magnitude of the signal being measured. To control autoranging, use [SENS:]VOLT:RANG:AUTO.
- Changing the reference level doesn't affect the per-division setting. To specify the vertical scale per-division, use DISP[:WIND]:TRAC:Y:PDIV.
- To autoscale the display, use DISP[:WIND]:TRAC:Y:AUTO.

DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:RLEVel:AUTO command/query

Controls whether the reference level tracks the input range value.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:RLEVel:AUTO OFF|0|ON|1`

Example Statements:
OUTPUT 719;":display:trac:y:rlev:auto OFF"
OUTPUT 719;"Display:Wind:Trace:Y:Scal:Rlevel:Auto OFF"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:RLEVel:AUTO?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1 (ON)
SCPI Compliance:  confirmed
```

Description:

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

When range tracking is on, the reference level changes when the range setting changes. Immediately after power is turned on and after a preset, range tracking is “ON” and both values are 0 dBm (because measurement data is *spectrum* and data format is *log mag*). When range tracking is on, changes in the reference level do not affect the range setting. If the range value is incremented 4 dB, the reference level is also incremented 4 dB.

Range tracking is invalid for phase, demodulation, and group delay traces and when the measurement data is frequency response, coherence, or user math. Range tracking is turned OFF in these cases.

Notes

Range tracking is turned OFF when an autoscale is performed or when the Y scale is changed for a real, imaginary, or linear magnitude trace.

If range tracking is on and autoranging is on, the reference level may change when the signal level changes, depending on how autoranging is configured.

Related Commands:

- To specify or query range settings, use the [SENS:]VOLT:RANG commands.
- To specify or query the reference level, use DISP[:WIND]:TRAC:Y:RLEV.
- To specify the position of the reference line, use DISP[:WIND]:TRAC:Y:RPOS.
- To autoscale the display, use DISP[:WIND]:TRAC:Y:AUTO.
- To have the reference level value track the input range setting, use DISP[:WIND]:TRAC:Y:RLEV:AUTO.

DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:RPOSition command/query

Specifies the position of the reference line.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:RPOSition <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:100
<unit>	::=	PCT
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;":DISPLAY:WIND2:TRACE:Y:RPOS 66"
OUTPUT 719;"display:wind1:trace:y:scale:rposition 100 pct"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:TRACe:Y[:SCALe]:RPOSition?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  100% (for spectrum magnitude)
SCPI Compliance:  confirmed
```

Description:

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

The reference line’s position is set as a percentage of the distance to the top of the trace area: 100% places the line at the top of the trace area, 50% places it at the middle, and 0% places it at the bottom.

The preset value for time, real, imaginary, or phase data is 50%. When power is first turned on and after preset, the reference line is off.

Related Commands:

- To turn the reference line on, use DISP[:WIND]:TRAC:Y:RLIN.
- To specify the value of the reference line, use
- To autoscale the display, use DISP[:WIND]:TRAC:Y:AUTO.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:WATerfall:AZIMuth — command/query

Specifies the Z-axis skew for all active waterfall traces.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:AZIMuth <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -99:99
<unit>	::=	PIXEL
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"DISPLAY:WIND2:WAT:AZIMUTH -16"
OUTPUT 719;"disp:waterfall:azim 39"
```

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:AZIMuth?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

The waterfall display draws a new trace at the top of the screen as older traces move toward the bottom. The *azimuth* value specifies how many pixels each successively-lower trace is shifted sideways. The top trace is not shifted by the azimuth value.

If the azimuth is 0, the waterfall traces do not shift sideways as they move down the trace box. Positive azimuth values shift the traces to the right as they move toward the bottom of the trace box. Negative azimuth values shift the traces to the left as they move toward the bottom of the trace box.

Related Commands:

- To turn waterfall displays on/off, use DISP:WAT.
- To set the elevation, use DISP:WAT:ELEV.
- To set the trace height, use DISP:WAT:HEIG
- To set the threshold, use DISP:WAT:THR.

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1|2|3|4]]:WATerfall:BLINe[:STATe] command/query

Turns waterfall display baseline on/off.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:BLINe[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Disp:Waterfall:Blin OFF"
OUTPUT 719;"DISPLAY:WIND:WAT:BLINE:STAT ON"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:BLINe[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

Baselines are used in conjunction with threshold settings to define the floor of the current scaling view. This is especially useful when the elevation setting is such that the display consists of many, overlapping traces.

Related Commands:

- To turn waterfall displays on/off, use DISP:WAT.
- To set the elevation, use DISP:WAT:ELEV.
- To set the trace height, use DISP:WAT:HEIG
- To set the threshold, use DISP:WAT:THR.

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1|2|3|4]]:WATerfall:ELEVation command/query

Specifies the number of pixels between traces (vertically) in the waterfall dislay.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:ELEVation <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:(see discussion)
<unit>	::=	PIXEL
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"display:wind3:wat:elevation 0"
OUTPUT 719;"Disp:Waterfall:Elev 0"
```

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:ELEVation?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

The waterfall display draws a new trace at the top of the screen as older traces move toward the bottom. The height of a grid is determined by the number of vertically stacked grids. When the number of grids is single, the vertical height is approximately 300 pixels. When two grids are displayed, vertical height is approximately 140 pixels.

Limits:

The *maximum* number of pixels accepted for this command is the number defined under DISP:FORM SING, which is about 300. This is true, regardless of the active display format (number of grids). The above discussion is a description of *practical* limits; setting the elevation and trace height to values that add to more than what is available just means part of the trace is not displayed.

- To set the number of grids, use DISP:FORM.
- To turn waterfall displays on/off, use DISP:WAT.
- To set the elevation, use DISP:WAT:ELEV.
- To set the azimuth, use DISP:WAT:AZIM.
- To set the threshold, use DISP:WAT:THR.
- To set the trace height, use DISP:WAT:HEIG

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1|2|3|4]]:WATerfall:HEIGht command/query

Specifies the vertical height (in pixels) of all active waterfall traces.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:HEIGht <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 1:(see discussion)
<unit>	::=	PIXEL
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":DISP:WATERFALL:HEIG 0"
OUTPUT 719;"display:wind:wat:height 0"
```

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:HEIGht?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

The waterfall display draws a new trace at the top of the screen as older traces move toward the bottom. The height of a grid is determined by the number of vertically stacked grids. When the number of grids is single, the vertical height is approximately 300 pixels. When two grids are displayed, vertical height is approximately 140 pixels.

Limits:

The *maximum* number of pixels accepted for this command is the number defined under DISP:FORM SING, which is about 300. This is true, regardless of the active display format (number of grids). The above discussion is a description of *practical* limits; setting the elevation and trace height to values that add to more than what is available just means part of the trace is not displayed.

Related Commands:

- To set the number of grids, use DISP:FORM.
- To turn waterfall displays on/off, use DISP:WAT.
- To set the elevation, use DISP:WAT:ELEV.
- To set the azimuth, use DISP:WAT:AZIM.
- To set the threshold, use DISP:WAT:THR.

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1|2|3|4]]:WATerfall:HLINe[:STATe] command/query

Turns waterfall hidden lines on/off.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:HLINe[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Disp:Window4:Wat:Hlin:State OFF"
OUTPUT 719;"DISP:WATERFALL:HLIN ON"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:HLINe[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

If enabled, the part of the current measurement trace that is obscured by the previous trace because the elevation between traces was not sufficient to separate the two, is blanked or hidden. This allows signals to become significantly more visible when the waterfall is displayed with many traces and a small elevation value is active.

This command affects *all* active traces regardless of whether a specific window/trace is used (e.g. DISP:WIND2).

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1|2|3|4]]:WATerfall[:STATe] command/query

Turns waterfall display on/off for active traces.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;":disp:waterfall OFF"
OUTPUT 719;"Disp:Window:Wat:Stat OFF"

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  0 (off)
SCPI Compliance:  instrument-specific
```

Description:

The waterfall display draws new traces at the top of the screen as older traces move toward the bottom. The height of each trace, and elevation between spaces may be user-defined as well as features such as hidden line, threshold, and baseline.

Related Commands:

- To specify which traces are active, use DISP:ACT.
- To set the threshold, use DISP:WAT:THR.
- To set the elevation, use DISP:WAT:ELEV.
- To set the trace height, use DISP:WAT:HEIG

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

DISPlay[:WINDow[1 | 2 | 3 | 4]]:WATerfall:THReshold — command/query

Sets the threshold for all active waterfall displays.

Command Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:THReshold <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:100
<unit>	::=	PCT
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"DISPLAY:WIND2:WATERFALL:THR 21"
OUTPUT 719;"disp:waterfall:thr 63"
```

Query Syntax: `DISPlay[:WINDow[1|2|3|4]]:WATerfall:THReshold?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYB (waterfall & spectrogram display)
Synchronization Required:  no
Preset State:  0 (%)
SCPI Compliance:  instrument-specific
```

Description:

This command is useful for removing noise-floor clutter from the waterfall display. The threshold level is the percentage of the trace height, below which the signal is not displayed.

This command affects *all* active traces regardless of whether a specific window/trace is used (e.g. DISP:WIND2).

Related Commands:

- To turn waterfall displays on/off, use DISP:WAT.
- To set the elevation, use DISP:WAT:ELEV.
- To set the azimuth, use DISP:WAT:AZIM.
- To set the trace height, use DISP:WAT:HEIG

For more information on waterfall and spectrogram display types, see online help and the concepts discussion in the *Operator's Guide*.

FORMat[:DATA] command/query

Specifies the format (ASCii or REAL) for block data transferred across the GPIB.

Command Syntax: `FORMat[:DATA] (ASCii|REAL) [, <number>|<bound>]`

<number> ::= a real number (NRf data)
limits: ASCii: 3:15
REAL: 32|64

Example Statements:
OUTPUT 719;":FORMAT ASC, 8"
OUTPUT 719;"form:data REAL, 64"
OUTPUT 719;"Form asc"

Query Syntax: `FORMat[:DATA]?`

Return Format: `CHAR, Integer`

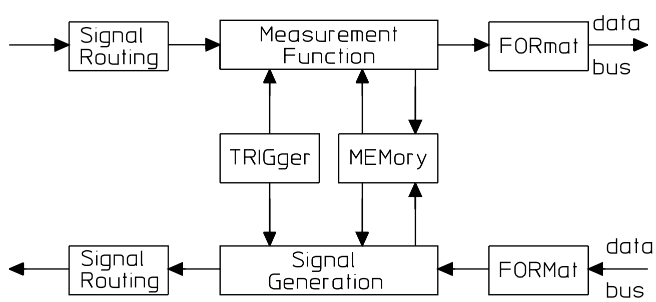
Attribute Summary:
```
Synchronization Required:  no
Preset State:  ASC,+15
SCPI Compliance:  confirmed
```

Description:

FORMat[:DATA] affects data transfers initiated with the following commands:

- CALC:DATA?
- PROG[:SEL]:NUMB
- TRACe[:DATA]
- TRACe:X:[:DATA]

The FORMat block of the Instrument Model converts between data representations, especially on the data that is transferred over the external interface. If REAL is chosen, 32-bit or 64-bit numbers may be specified. If ASCII is chosen, the number specified is the number of significant digits. Real data is IEEE 754 format, MSB first.



Model of a Programmable Instrument (from SCPI Command Reference)

HCOPy:ABORt **command**

Aborts a plot or print operation.

Command Syntax: `HCOPy:ABORt`

Example Statements:

OUTPUT 719;":Hcopy:Abor"
OUTPUT 719;"HCOP:ABORT"

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

HCOPy:DATA? query

Queries the plot data which is returned in an indefinite length data block.

Query Syntax: `HCOPy:DATA?`

Example Statements:
OUTPUT 719;"hcop:data?"
OUTPUT 719;"Hcop:Data?"

Return Format: `DEF_USER`

Attribute Summary:
```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

HCOPy:DESTination **command/query**

Selects the I/O port for hard copy output.

Command Syntax: `HCOPy:DESTination <CMDSTR>`

<CMDSTR> ::= 'SYST:COMM:GPIB'
'SYST:COMM:SER1'
'SYST:COMM:SER2'
'SYST:COMM:CENT'
'MMEM'

Example Statements: OUTPUT 719;":hcop:destination 'syst:comm:gpib'"
OUTPUT 719;"Hcop:Dest 'syst:comm:cent'"

Query Syntax: `HCOPy:DESTination?`

Return Format: `STRING`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  confirmed
```

Description:

GPIB is the GPIB connector for which an address must be supplied. If you choose the GPIB port, control of the bus must be passed to the analyzer so that it can perform the plotting.

SER1 is the first RS-232 serial interface port; it is labeled COM 1 on the rear panel.

- If the RF section of the 89441 is in use, SER1 is the only serial port available for hard copy output; SER2 (COM2) is used to control the RF section.

SER2 is the second RS-232 serial interface port; it is labeled COM 2 on the rear panel.

- If the RF section of the 89441 is in use, SER2 is *not* available for hard copy output; SER2 (COM2) is used to control the RF section.

CENT is the Centronics (parallel) port on the rear panel.

MMEM is the mass memory specifier. This signifies plotting or printing to a file. Use MMEM:NAME to specify the filename.

HCOPy:DEVice:CMAP:DEFault — command

Specifies that default values be used for the plotter pen assignments.

Command Syntax: `HCOPy:DEVice:CMAP:DEFault`

Example Statements:

```
OUTPUT 719;":HCOPY:DEV:CMAP:DEF"
OUTPUT 719;"hcop:device:cmap:default"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  (see discussion)
SCPI Compliance:  confirmed
```

Description:

The default plotter pen assignments are:

text	6
grid	1
marker1	2
marker2	2
marker3	7
marker4	2
trace1	3
trace2	7
trace3	2
trace4	6

HCOPy:DEVice:COLor — command/query

Controls color (on/off) for TIFF output over the GPIB.

Command Syntax: `HCOPy:DEVice:COLor OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Hcop:Dev:Color OFF"
OUTPUT 719;"HCOP:DEVICE:COL ON"

Query Syntax: `HCOPy:DEVice:COLor?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  yes
Preset State:  1 (on)
SCPI Compliance:  instrument-specific
```

Description:

ON specifies color for TIFF files. To override this, use this command *after* the language selection is made.

OFF specifies that TIFF color is turned off; the images are black and white.

Note TIFF files transferred with the LAN file-transfer protocol, FTP (available in option UG7) are *always* color, regardless of this command setting.

HCOPy:DEVice:LANGuage command/query

Selects type of output from the hard copy output.

Command Syntax: `HCOPy:DEVice:LANGuage PCL|HPGL|PHPGl|TIFF`

Example Statements:
OUTPUT 719;":hcop:device:lang PCL"
OUTPUT 719;"Hcopy:Dev:Lang HPGL"

Query Syntax: `HCOPy:DEVice:LANGuage?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  HPGL
SCPI Compliance:  confirmed
```

Description:

PCL is the language used by Hewlett-Packard LaserJet printers.

HPGL is the graphics language used by Hewlett-Packard plotters.

PHPGL is the HP-GL language as supported by some Hewlett-Packard PCL LaserJet printers (such as Series III).

TIFF is the Tagged Information File Format (TIFF is a trademark of Aldus Corporation).

HCOPy:DEVice:RESolution **command/query**

Specifies dots per inch for screen dumps to PCL printers.

Command Syntax: `HCOPy:DEVice:RESolution <number>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:600

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":Hcop:Device:Res 150"
OUTPUT 719;"HCOPY:DEV:RES 300"

Query Syntax: `HCOPy:DEVice:RESolution?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  confirmed
```

Description:

This information is not used for HPGL language output.

HCOPy:DEVice:SPEed command/query

Specifies the pen-movement speed (cm/s) for plot operations initiated by the analyzer.

Command Syntax: `HCOPy:PLOT:SPEed <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:100

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"Hcop:Dev:Spe 50"
OUTPUT 719;"HCOP:DEVICE:SPE 10"

Query Syntax: `HCOPy:DEVice:SPEed?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +100
SCPI Compliance:  confirmed
```

Description:

This information is not used for PCL language output.

This command specifies the pen-movement speed in centimeters per second (cm/s). Check the plotter's documentation to be sure that it supports the specified speed.

The value 0 causes the plotter to use it's default speed, which is normally its highest speed capability.

HCOPy[:IMMediate] command

Plots or prints the entire screen.

Command Syntax: `HCOPy[:IMMediate]`

Example Statements:

```
OUTPUT 719;"HCOPY:IMM"
OUTPUT 719;"hcopy"
```

Attribute Summary:

```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The entire screen is output to the port defined by HCOP:DEST using the language defined by HCOP:DEV:LANG. If the GPIB port is the destination, the data is directed to an GPIB device whose address is specified with HCOP:PLOT:ADDR or HCOP:PRIN:ADDR. If one of the serial ports is the destination, serial port parameter are defined with the SYST:COMM:SER commands.

HPGL will use HCOP:PLOT:ADDR for the GPIB address, and PCL and PHPGL will use HCOP:PRINT:ADDR for the GPIB address.

This command has the same effect as HCOP:ITEM:ALL[:IMM].

Note To Plot or print to the GPIB port, the analyzer must be the active controller on the bus. The active controller must pass control to the analyzer. After the command has been executed, the analyzer will return control to the controller.

Whole-screen graphics files may also be created in any of the supported graphics output formats/languages and moved to the controller with the LAN file-transfer utility, FTP (option UG7).

HCOPy:ITEM:ALL[:IMMediate] command

Plots or prints the entire screen.

Command Syntax: `HCOPy:ITEM:ALL[:IMMediate]`

Example Statements:

```
OUTPUT 719;":Hcop:Item:All"
OUTPUT 719;"HCOP:ITEM:ALL:IMM"
```

Attribute Summary:

```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The entire screen is output to the port defined by HCOP:DEST using the language defined by HCOP:DEV:LANG. If the GPIB port is the destination, the data is directed to an GPIB device whose address is specified with HCOP:PLOT:ADDR or HCOP:PRIN:ADDR. If one of the serial ports is the destination, serial port parameter are defined with the SYST:COMM:SER commands.

HPGL will use HCOP:PLOT:ADDR for the GPIB address, and PCL and PHPGL will use HCOP:PRINT:ADDR for the GPIB address.

This command has the same effect as HCOP[:IMM].

Note To Plot or print to the GPIB port, the analyzer must be the active controller on the bus. The active controller must pass control to the analyzer. After the command has been executed, the analyzer must pass control back to the controller.

Whole-screen graphics files may also be created in any of the supported graphics output formats/languages and moved to the controller with the LAN file-transfer utility, FTP (option UG7).

HCOPy:ITEM:ANNotation:COLor **command/query**

Specifies the pen used to plot miscellaneous annotations.

Command Syntax: `HCOPy:ITEM:ANNotation:COLOR <number>|<step>|<bound>`

<number>	::=	a real number (NRf data) limits: 0:16
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;"hcop:item:annotation:colo 3"
OUTPUT 719;":HCOP:ITEM:ANN:COLOR 15"

Query Syntax: `HCOPy:ITEM:ANNotation:COLor?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  +4
SCPI Compliance:  confirmed
```

Description:

The annotation pen is used to plot the following:

- Instrument state.
- Disk catalog.
- State information.
- Time stamp.
- Status or error messages.

Nothing is plotted for items with pen values of 0 or for which the pen value exceeds the limits of the plotter in use.

All output device configuration information is sent when the HCOP command is executed.

This information is not used for PCL language output.

HCOPy:ITEM:FFEed:STATe **command/query**

Specifies a (plotter) eject or (printer) form feed after the operation is complete.

Command Syntax: `HCOPy:ITEM:FFEed:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"hcopy:item:ffeed:stat OFF"
OUTPUT 719;"Hcop:Item:Ffe:State ON"

Query Syntax: `HCOPy:ITEM:FFEed:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

Enables or disables the plotter page-eject feature or a printer form feed feature. The specified state is used for all plotting and printing operations initiated by the analyzer. Check the output device’s documentation to be sure that it supports the requested the page-eject state.

HCOPy:ITEM:TDSTamp:CFORmat — command/query

Specifies the clock format (12 or 24-hour) for the time/date stamp.

Command Syntax: `HCOPy:ITEM:TDSTamp:CFORmat HR12|HR24`

Example Statements:
OUTPUT 719;":HCOP:ITEM:TDSTAMP:CFOR HR24"
OUTPUT 719;"hcopy:item:tdst:cformat HR12"

Query Syntax: `HCOPy:ITEM:TDSTamp:CFORmat?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

HCOPy:ITEM:TDSTamp:FORMat **command/query**

Specifies the year-month-day format for the time/date stamp.

Command Syntax: `HCOPy:ITEM:TDSTamp:FORMat <param>`

<param> ::= FORMat1|FORMat2|FORMat3|FORMat4|FORMat5|FORMat6

Example Statements:
OUTPUT 719;"Hcop:Item:Tdst:Form FORMAT6"
OUTPUT 719;"HCOPY:ITEM:TDSTAMP:FORM FORMAT5"

Query Syntax: `HCOPy:ITEM:TDSTamp:FORMat?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  FORM4
SCPI Compliance:  instrument-specific
```

Description:

FORMAT1 = DD/MM/YY
FORMAT2 = DD.MM.YY
FORMAT3 = YY MM DD
FORMAT4 = YY-MM-DD
FORMAT5 = MM-DD-YY
FORMAT6 = MM/DD/YY

HCOPy:ITEM:TDSTamp:STATe **command/query**

Turns time/date stamp on or off for print and plot operations.

Command Syntax: `HCOPy:ITEM:TDSTamp:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;":HCOP:ITEM:TDSTAMP:STATE OFF"
OUTPUT 719;"hcop:item:tdst:stat on"

Query Syntax: `HCOPy:ITEM:TDSTamp:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +1
SCPI Compliance:  instrument-specific
```

Description:

When the time stamp is on, time and date information is printed with the screen data.

When the time stamp is off, time and date information is *not* printed or plotted.

HCOPy:ITEM[:WINDow[1 | 2 | 3 | 4]]:TRACe:COLor **command/query**

Specifies the pen used to plot the selected trace and annotation.

Command Syntax: `HCOPy:ITEM:[WINDOW[1|2|3|4]]:TRACE:COLOR <param>`

<param> ::= <number>|<step>|<bound>

<number> ::= a real number (NRf data)
limits: 0:16

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"hcop:item:window1:trac:color 20"
OUTPUT 719;"HCOPY:ITEM:WINDOW4:TRACE:COLOR 4"

Query Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:COLor?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  trace/pen 1/2 2/3 3/2 4/3
SCPI Compliance:  confirmed
```

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.

<number> is the pen number, which corresponds to the slot number in the plotter pen carrousel.

All output device configuration information is sent when the HCOP command is executed.

This information is not used for PCL language output.

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:GRATicule:COLor command/query

Specifies the pen used to plot the trace graticule.

Command Syntax: `HCOPy:ITEM[:WINDOW[1|2|3|4]]:TRACe:GRATicule:COLor <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:16
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;":Hcopy:Item:Trace:Grat:Color 1"
OUTPUT 719;"HCOP:ITEM:WIND3:TRAC:GRAT:COL 12"

Query Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:GRATicule:COLor?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

The grid, or graticule, pen is used to plot the graticule overlaying the trace, the border around the instrument state, and the border around the disk catalog.

All output device configuration information is sent when the HCOP command is executed.

This information is not used for PCL language output.

The Window node has no effect in this command. Whatever value is set for any trace is the active value for all traces.

HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:GRATicule[:IMMediate] command

Plots or prints the graticule, only.

Command Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:GRATicule[:IMMediate]`

Example Statements:
OUTPUT 719;":hcop:item:trac:graticule"
OUTPUT 719;"Hcop:Item:Window:Trac:Graticule:Imm"

Attribute Summary:
```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Whole-screen graphics files may also be created in any of the supported graphics output formats/languages and moved to the controller with the LAN file-transfer utility, FTP (option UG7).

HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe[:IMMediate] command

Plots the specified trace, only (not applicable to printing).

Command Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe[:IMMediate]`

Example Statements:
OUTPUT 719;"HCOP:ITEM:WIND3:TRACE:IMM"
OUTPUT 719;"hcop:item:trac"

Attribute Summary:
```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command plots *all* traces currently displayed.

If the current language is PCL, this command returns an error.

Whole-screen graphics files may also be created in any of the supported graphics output formats/languages and moved to the controller with the LAN file-transfer utility, FTP (option UG7).

HCOPy:ITEM[:WINDow[1 | 2 | 3 | 4]]:TRACe:LTYPe command/query

Specifies the plot line type for the selected trace.

Command Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]:TRACe:LTYPe <param>`

<param> ::= SOLid|DOTTed|DASHed|STY0|STY1|STY2|STY3|STY4|STY5|STY6

Example Statements: OUTPUT 719;":Hcopy:Item:Trac:Ltype SOLID"
OUTPUT 719;"HCOP:ITEM:WIND4:TRAC:LTYPE STYLE2"

Query Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:LTYPe?`

Return Format: `CHAR`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  SOL
SCPI Compliance:  confirmed
```

Description:

DOTTed is the same as STYLe1 and DASHed is the same as STYLe2. The other style types are device-dependent; see the plotter documentation (linetype HPGL command LT) to determine how these are interpreted.

All output device configuration information is sent when the HCOP command is executed.

This information is not used for PCL language output.

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, "WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, "1" is assumed and the command affects trace A.

HCOPy:ITEM[:WINDow[1 | 2 | 3 | 4]]:TRACe:MARKer:COLor — command/query

Specifies the pen used to plot markers for the selected trace.

Command Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:MARKer:COLor <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:16
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"hcopy:item:wind2:trace:mark:color 6"
OUTPUT 719;"Hcop:Item:Wind3:Trac:Marker:Col 9"
```

Query Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:MARKer:COLor?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  trace/pen 1/5 2/6 3/5 4/6
SCPI Compliance:  confirmed
```

Description:

<number> is the pen number corresponding to the slot number in the plotter pen carrousel. The pen specified with this command is used to plot main marker, reference marker, and marker function text and graphics for the specified trace.

The Window node is default (use is optional). This specifies which trace (A, B, C, or D) is to be affected by the command. If the Window node is not used in the command line, “WIND1" is assumed and the command affects trace A. If WIND is used but no trace number is specified, ”1" is assumed and the command affects trace A.

Nothing is plotted for items with pen values of 0 or for which the pen value exceeds the limits of the plotter in use.

All output device configuration information is sent when the HCOP command is executed.

This information is not used for PCL language output.

HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:MARKer[:IMMediate] command

Plots or prints the specified marker, only.

Command Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:MARKer[:IMMediate]`

Example Statements:

OUTPUT 719;"hcop:item:wind:trac:marker:imm"
OUTPUT 719;"Hcopy:Item:Trac:Marker"

Attribute Summary:

```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command plots *all* trace markers currently displayed.

If the current language is PCL, this command returns an error.

Whole-screen graphics files may also be created in any of the supported graphics output formats/languages and moved to the controller with the LAN file-transfer utility, FTP (option UG7).

HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:MARKer:OFFSet[:IMMediate] command

Plots or prints the specified offset marker, only.

Command Syntax: `HCOPy:ITEM[:WINDow[1|2|3|4]]:TRACe:MARKer:OFFSet[:IMMediate]`

Example Statements:
OUTPUT 719;":HCOP:ITEM:TRAC:MARK:OFFSET"
OUTPUT 719;"hcop:item:wind:trac:marker:offs:immediate"

Attribute Summary:
```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command plots *all* offset markers currently displayed.

If the current language is PCL, this command returns an error.

Whole-screen graphics files may also be created in any of the supported graphics output formats/languages and moved to the controller with the LAN file-transfer utility, FTP (option UG7).

HCOPy:PAGE:DIMensions:AUTO command/query

Specifies whether default or user-defined settings for P1 and P2 are used to define page dimensions.

Command Syntax: `HCOPy:PAGE:DIMensions:AUTO OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Hcop:Page:Dimensions:Auto OFF"
OUTPUT 719;"HCOPY:PAGE:DIM:AUTO ON"

Query Syntax: `HCOPy:PAGE:DIMensions:AUTO?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

All output device configuration information is sent when the HCOP command is executed. If HCOP:PAGE:DIM:AUTO is on, no page dimension commands are sent; the current plotter settings are used. If AUTO is OFF, the user-defined values are sent.

- To define values for P1 and P2, use HCOP:PAGE:DIM:USER:LLEFt and URIGht.

This information is not used for PCL language output.

HCOPy:PAGE:DIMensions:USER:LLEFt command/query

Sets or queries the x,y position of the lower-left corner of the plotter page (P1).

Command Syntax: `HCOPy:PAGE:DIMensions:USER:LLEFt <param>, <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -32767:32767
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;":hcop:page:dim:user:lleft 12683, 22198"
OUTPUT 719;"Hcop:Page:Dim:User:Lleft 15708, 9913"

Query Syntax: `HCOPy:PAGE:DIMensions:USER:LLEFt?`

Return Format: `Integer, Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  +250,+596
SCPI Compliance:  instrument-specific
```

Description:

All output device configuration information is sent when the HCOP command is executed.

This information is not used for PCL language output.

HCOPy:PAGE:DIMensions:USER:URIGht **command/query**

Sets or queries the x,y position of the upper-right corner of the plotter page (P2).

Command Syntax: `HCOPy:PAGE:DIMensions:USER:URIGht <param>, <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -32767:32767
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;"HCOP:PAGE:DIM:USER:URIGHT 11675, 5060"
OUTPUT 719;"hcop:page:dim:user:uright -14868, 28532"

Query Syntax: `HCOPy:PAGE:DIMensions:USER:URIGht?`

Return Format: `Integer, Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  +10250,+7796
SCPI Compliance:  instrument-specific
```

Description:

All output device configuration information is sent when the HCOP command is executed.

This information is not used for PCL language output.

HCOPy:PLOT:ADDRess **command/query**

Specifies the GPIB address assigned to the plot device.

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 719;":HCOPY:PLOT:ADDR 5"
OUTPUT 719;"hcopy:plot:address 9"

Query Syntax: `HCOPy:PLOT:ADDRess?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This address is used for HPGL and PHPGL output formats and data output to the GPIB port. The plotter address must be specified with this command before the HCOP[:IMM] command is executed. If a plotter device is not at the specified address, the plot operation times out after about 20 seconds.

HCOPy:PRINt:ADDRess **command/query**

Specifies the GPIB address assigned to the print device.

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 719;":hcopy:prin:addr 3"
OUTPUT 719;"Hcopy:Prin:Address 1"

Query Syntax: `HCOPy:PRINt:ADDRess?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This address is used for PCL and TIF output formats and data output to the GPIB port. The printer address must be specified with this command before the HCOP[:IMM] command is executed. If a printer device is not at the specified address, the print operation times out after about 20 seconds.

INITiate:CONTinuous command/query

Specifies continuous or single sweeps.

Command Syntax: `INITiate:CONTinuous OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Init:Continuous ON"
OUTPUT 719;"INIT:CONT OFF"

Query Syntax: `INITiate:CONTinuous?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  ON (continuous)
SCPI Compliance:  instrument-specific
```

Description:

ON is continuous-sweep operation.

OFF is single-sweep operation. When this state is active, a single sweep occurs when INIT[:IMM] is sent.

INITiate[:IMMediate] command

Forces the trigger system to exit the idle state.

Command Syntax: `INITiate[:IMMediate]`

Example Statements:

OUTPUT 719;"initiate:imm"
OUTPUT 719;"Initiate"

Attribute Summary:

```
Synchronization Required:  yes
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

Characteristics:

- Ignored unless INIT:CONT is OFF (in single sweep operation).
- Forces the process to exit idle state (in SCPI trigger model) and the trigger system to initiate and complete one full trigger cycle.
- Ensures that changes to the analyzer's state are incorporated in the collection and presentation of new measurement results. See Synchronization in the *GPIB Programmer's Guide*.
- Equivalent to the Single hardkey when in single-sweep mode.

INPut[1 | 2]:COUPling — command/query

Selects ac or dc coupling for the selected channel.

Command Syntax: `INPut[1|2]:COUPling AC|DC`

Example Statements:

```
OUTPUT 719;":INP2:COUP AC"
OUTPUT 719;"input:coup DC"
```

Query Syntax: `INPut[1|2]:COUPling?`

Return Format: `CHAR`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  AC
SCPI Compliance:  confirmed
```

Description:

Note Use dc coupling for small spans at low frequencies to avoid the filter effect associated with ac coupling (signal is down 3 dB at 1 Hz).

INPut[1 | 2]:FILTer[:LPASs][:STATe] command/query

Enables or disables the anti-alias circuits.

Command Syntax: `INPut[1|2]:FILTer[:LPASs][:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Input2:Filt:Lpas:State OFF"
OUTPUT 719;"INP:FILTER OFF"

Query Syntax: `INPut[1|2]:FILTer[:LPASs][:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1
SCPI Compliance:  confirmed
```

Description:

By default, the analyzer’s anti-alias filter is enabled. Use this command to disable the anti-alias filter.

Notes

If the anti-alias filter is off, published specifications for the analyzer are not guaranteed.

The analyzer always applies dc offset correction, regardless of the anti-alias filters status.

INPut[1 | 2]:IMPedance command/query

Selects the internal termination impedance.

Command Syntax: `INPut[1|2]:IMPedance <number>[<unit>]`

<number> ::= 50|75|1000000
<unit> ::= OHM

Example Statements:
OUTPUT 719;":input1:imp 50"
OUTPUT 719;"Inp2:Imp 1MOHM"
OUTPUT 719;"INPUT1:IMP 1e6"

Query Syntax: `INPut[1|2]:IMPedance?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  50
SCPI Compliance:  confirmed
```

Description:

Related Commands:

- When 1 MΩ is active, you can specify the reference impedance used to calculate dBm with the CORR:IMP command.

Note For the 89441A, the 1 MΩ selection is not valid for channel 1 (input1) if the receiver type is RF (ROUT:REC RF2). Channel 2 impedance is not limited by receiver type, but may not be installed.

INPut[1|2][:STATe] command/query

This key turns on/off the selected channel.

Command Syntax: `INPut[1|2][:STATe]  OFF|0|ON|1`

Example Statements:
OUTPUT 719;"INP1:STATE ON"
OUTPUT 719;"inp2 on"

Query Syntax: `INPut[1|2][:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +1
SCPI Compliance:  confirmed
```

Description:

Turning a channel off increases the measurement rate in some cases, since no calculations are made for a channel that is off. It also doubles the time capture record length for the channel that is on.

- If you send a command without specifying a channel, the command affects channel 1.
- If you send a query without specifying a channel, the analyzer returns the state of channel 1.
- If you query a channel that is not installed, the analyzer returns a blank field.

Note The second input channel is optional and may not be installed on the analyzer in your system. To determine which options are installed, use the *OPT? query.

INSTrument:NSELect — command/query

Select instrument mode by sending an integer number.

Command Syntax: `INSTrument:NSELect <number>|<bound>`

<number> ::= a real number (NRf data)
limits: 1:5

<bound> := MAX|MIN

Example Statements:
OUTPUT 719;"INSTRUMENT:NSEL 1"
OUTPUT 719;"inst:nselect 3"

Query Syntax: `INSTrument:NSELect?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  89410: 3, 89441: 1
SCPI Compliance:  confirmed
```

Description:

This command performs the same function as INST:SEL except that it sends numbers to specify the mode type as follows:

1 = SCALar
2 = ADEModulation
3 = VECTor
4 = DDEModulation
5 = VDEModulation
6 = WCDma (Wideband CDMA)

Note Digital demodulation is valid only when option AYA is installed. Video demodulation is valid only when both option AYA and option AYH are installed. Wideband CDMA is valid only when option B73 is installed.

INSTrument[:SELect] command/query

Selects the instrument mode.

Command Syntax: `INSTrument[:SELect] SCALar|DEMod|ADEMod|DDEMod|VECTor|VDEMod|WCDMa`

Example Statements:
OUTPUT 719;":Inst ADEM"
OUTPUT 719;"INSTRUMENT:SEL VECTOR"

Query Syntax: `INSTrument[:SELect]?`

Return Format: `CHAR`

Attribute Summary:
```
Option: AYA (Vector Mod. Analysis) for DDEMod
        AYH (Digital Video Analysis) for VDEMod
        B73 (Wideband CDMA analysis) for WCDMa
Synchronization Required:  no
Preset State:  89410: VECTor, 89441: SCALar
SCPI Compliance:  confirmed
```

Description:

SCAL selects Scalar Measurements as the instrument mode. This mode configures the analyzer to perform scalar measurements (no phase) on either channel. This mode uses stepped FFT measurement techniques to achieve lower resolution bandwidths. Gated time is NOT available in Scaler mode.

ADEM and **DEM** selects the Analog Demodulation Measurements mode. This mode configures the analyzer to allow AM, PM, and FM demodulation measurements. **DEM** provides compatibility with earlier software revisions. Gated time is available in the Analog Demod mode.

DDEM selects Digital Demodulation as the instrument mode. This mode exists only in analyzers that have option AYA. This mode configures the analyzer to examine signals such as QPSK, DQPSK, $\frac{\pi}{4}$DQPSK, BPSK, 8PSK, 16 QAM, 32 QAM, MSK, and FSK modulation formats. It offers automatic carrier and symbol locking. Gated time is NOT available in Digital Demod mode.

VECT selects Vector Measurements as the instrument mode. This mode configures the analyzer to perform vector measurements (magnitude and phase) on either channel. This mode does *not* use stepped FFT, and thus can not achieve the low resolution bandwidths possible in the Scalar mode. Gated time is available in the Vector mode.

VDEM selects Video Demodulation Measurements as the instrument mode. This mode exists only in analyzers that have both options AYA and AYH.

WCDMa selects Wideband CDMA Measurements as the instrument mode. This mode exists only in analyzers that have option B73.

Note The instrument mode may also be specified using integer numbers with the INST:NSEL command.

MEMory:DELete[:NAME] command

Purges the memory allocated for a specific item.

Command Syntax: `MEMory:DELete[:NAME] RDISk|D1|D2|D3|D4|D5|D6|DREG|MSCan`

Example Statements:
OUTPUT 719;"MEM:DEL RDIS"
OUTPUT 719;"mem:delete:name rdisk"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command deletes an item from existence, freeing the memory allotted to it for reuse.

RDISK identifies the (nonvolatile) RAM disk.

D1|D2|D3|D4|D5|D6 identifies one of the six data registers.

DREG identifies all data registers.

MEMory:MALLocate:APPLication — command/query

Specifies amount of memory to allocate for downloadable programs.

Command Syntax: `MEMory:MALLocate:APPLication <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:3.40282347E+38

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

OUTPUT 719;":mem:mall:application 2147483647"
OUTPUT 719;"Mem:Mallocate:Appl 2147483647"

Query Syntax: `MEMory:MALLocate:APPLication?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This GPIB command specifies how much memory to allocate (in bytes) for applications. For best performance, set application memory to zero if there are no applications in the analyzer.

Before you load an application into the analyzer, see the documentation shipped with the application to determine the amount of memory required for the application.

MEMory:MALLocate:MEASurement:DEFault — command

Reset the memory configuration to default setup.

Command Syntax: `MEMory:MALLocate:MEASurement:DEFault`

Example Statements:

```
OUTPUT 719;"MEM:MALLOCATE:MEAS:DEFAULT"
OUTPUT 719;"mem:mall:measurement:def"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command allocates memory not already allocated to Instrument BASIC or as RAM disk, to data and math registers. Data registers may be used to save traces, to define math functions, and as a source of arbitrary source signals. Math registers hold interim results of math calculations. The size of these registers depends on the largest number of points in a trace that a user may wish to use. This parameter is called the Maximum Number of Frequency Points. The default configuration is:

Data Registers: 6

Math Registers: 6

Maximum Number of Frequency Points: 3201

This command does not change the settings for the RAM disk or the amount of memory allocated for IBASIC programming. If the amount of memory allocated to these two uses is sufficiently large, the DEFAULT command may not be able to allocate the configuration specified above. If this occurs, the value of the Maximum Number of Frequency Points is reduced until the configuration fits into the available memory.

Note The number of frequency points *used* is different from the *maximum* number of frequency points in that the latter is used to define the amount of memory allocated to the math and data registers. Each register is configured to accept the maximum number of frequency points, so making this number larger causes the register configuration to use more memory.

Related Commands:

- To specify the number of frequency points in a trace, use SWE:POIN.
- To specify the maximum number of frequency points in a trace, use MEM:MALL:MEAS:FPO.
- To specify the Max number of time points, use MEM:MALL:MEAS:TPO.
- To specify the number of data registers, use MEM:MALL:MEAS:REG.
- To specify the number of math temporary registers, use MEM:MALL:MTEM.
- If the IBASIC option (1C2) is installed, program memory is allocated with the MEM:MALL:PROG command. Option UTH adds 20MB of RAM that may be allocated to IBASIC programming or a RAM disk volume.
- To specify the RAM disk size, use MMEM:INIT.

MEMory:MALLocate:MEASurement:FPOints **command/query**

Specifies the maximum number of frequency points.

Command Syntax: `MEMory:MALLocate:MEASurement:FPOints <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 51:3201

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements: OUTPUT 719;"MEM:MALL:MEAS:FPO 801"
OUTPUT 719;"memory:mall:meas:fpoints 201"

Query Syntax: `MEMory:MALLocate:MEASurement:FPOints?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

Valid values for this parameter are 51, 101, 201, 401, 801, 1601, or 3201. The current value is stored in non-volatile RAM and is not affected by preset. The factory default is 1601.

The value specified with this command must be equal to or greater than:

- The number of frequency points for the trace.
- The number of points in any stored trace that you wish to recall.

Note The number of frequency points *used* is different from the *maximum* number of frequency points in that the latter is used to define the amount of memory allocated to the math and data registers as well as setting a limit on the number of frequency points currently selected. Each register is configured to accept the maximum number of frequency points, so making this number larger causes the register configuration to use more memory.

The standard amount of RAM is supplemented with a 20 MB RAM board (option UTH), which includes additional I/O (another GPIB connector and a LAN interface).

Related Commands:

- To allocate memory for IBASIC programs, use MEM:MALL:PROG.
- To specify the Max number of time points, use MEM:MALL:MEAS:TPO.
- To specify the number of data registers, use MEM:MALL:MEAS:REG.
- To specify the number of math temporary registers, use MEM:MALL:MEAS:MTEM.
- To specify the RAM disk size, use MMEM:INIT.
- To remove the RAM disk, use MEM:DEL RDIS.
- To specify the number of frequency points in a trace, use SWE:POIN.

MEMory:MALLocate:MEASurement:MAXSpan — command/query

Sets the maximum frequency span for W-CDMA measurements.

Command Syntax:

```
MEMory:MALLocate:MEASurement:MAXSpan
{<number>[<unit]}|<step>|<bound>
```

<number>	::=	a real number (NRf data) limits: 5000000:20000000
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;"mem:mallocate:meas:maxs 5000000"
OUTPUT 719;"Memory:Mall:Measurement:Maxspan 10000000"

Query Syntax:

```
MEMory:MALLocate:MEASurement:MAXSpan?
```

Return Format:

```
Real
```

Attribute Summary:

```
Option:  B73 (Wideband CDMA analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

W-CDMA measurements require large amounts of memory. The analyzer allocates enough memory to accomodate the frequency span set by this command. Reducing the amount of memory allocated for W-CDMA measurments frees memory for other purposes.

For best performance, set the maximum W-CDMA span to the smallest frequency span that you will measure.

The maximum W-CDMA span may limit the maximum chip rate. If you cannot set the chip rate to its maximum value, increase the maximum W-CDMA span.

MEMory:MALLocate:MEASurement:MSSRate command/query

Specifies the ratio MaxSpan/SymbolRate which affects memory allocation.

Command Syntax: `MEMory:MALLocate:MEASurement:MSSRate <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 16.0:100.0

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"mem:mallocate:meas:mssr 16.0"
OUTPUT 719;"Memory:Mall:Measurement:Mssr 50.8"

Query Syntax: `MEMory:MALLocate:MEASurement:MSSRate?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

All digital demodulation types have span limitations defined by a ratio of the maximum span per symbol rate. For a given symbol rate, this ratio limits the maximum span to conserve memory usage. All digital demodulation types except FSK have a fixed MSSR ratio of 15.625. Only the MSSR for FSK can be user-defined. Increasing the ratio value increases the amount of memory allocated for the measurement.

This memory configuration command is valid only for the FSK modulation type. The current value is stored in non-volatile RAM and is not affected by preset. The factory default is 30.

The standard amount of RAM is supplemented with a 20 MB RAM board (listed in the options as option UTH). The 20 MB RAM board includes additional I/O (another GPIB connector and a LAN interface).

Related Commands:

- To specify the frequency span, use FREQ:SPAN.
- To specify the symbol rate, use DDEM:SRAT.
- To allocate memory for IBASIC programs, use MEM:MALL:PROG.
- To specify the Max number of time points, use MEM:MALL:MEAS:TPO.
- To specify the number of data registers, use MEM:MALL:MEAS:REG.
- To specify the number of math temporary registers, use MEM:MALL:MEAS:MTEM.
- To specify the RAM disk size, use MMEM:INIT.
- To remove the RAM disk, use MEM:DEL RDIS.

MEMory:MALLocate:MEASurement:MTEMp — command/query

Specifies the number of temporary math buffers.

Command Syntax: `MEMory:MALLocate:MEASurement:MTEMp <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:100

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"memory:mall:measurement:mtem 74"
OUTPUT 719;"Mem:Mallocate:Meas:Mtemp 25"

Query Syntax: `MEMory:MALLocate:MEASurement:MTEMp?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  (see note below)
SCPI Compliance:  instrument-specific
```

Description:

Temporary math buffers are used to hold interim math results needed to evaluate complex expressions. They are not directly related to math functions even though there are six functions and the default number of math buffers is six.

The math buffer size is specified by the value entered for Maximum Number of Frequency Points. See the discussion of the MEM:MALL:MEAS:DEF command for more information on memory configuration.

Note The *default* number of math registers is six. This configuration is created by MEM:MALL:MEAS:DEF. The configuration is stored in non-volatile RAM which is not affected by *preset*.

MEMory:MALLocate:MEASurement:TPOints — command/query

Specifies the maximum number of time points in Digital Demod instrument mode.

Command Syntax: `MEMory:MALLocate:MEASurement:TPOints <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 64:4096

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;"MEM:MALL:MEAS:TPO 512"
OUTPUT 719;"memory:mall:meas:tpoints 4096"
```

Query Syntax: `MEMory:MALLocate:MEASurement:TPOints?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

Valid values for this parameter are 64, 128, 256, 512, 1024, 2048, or 4096. The current value is stored in non-volatile RAM and is not affected by preset. The factory default is 2048.

The value specified with this command must be equal to or greater than:

- The number of time points for the trace.
- The number of points in any stored trace that you wish to recall.

The standard amount of RAM is supplemented with a 20 MB RAM board (listed under the options as option UTH). The 20 MB RAM board includes additional I/O (another GPIB connector and a LAN interface).

Related Commands:

- To specify the maximum number of frequency points in a trace, use MEM:MALL:MEAS:FPO.
- To allocate memory for IBASIC programs, use MEM:MALL:PROG.
- To specify the number of math temporary registers, use MEM:MALL:MEAS:MTEM.
- To specify the RAM disk size, use MMEM:INIT.
- To remove the RAM disk, use MEM:DEL RDIS.
- To specify the number of frequency points in a trace, use SWE:POIN.

MEMory:MALLocate:PROGram

command/query

Specifies amount of memory to allocate for BASIC programs.

Command Syntax: `MEMory:MALLocate:PROGram <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:3.40282347E+38

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;":MEM:MALL:PROGRAM 2147483647"
OUTPUT 719;"mem:mallocate:prog 2147483647"
```

Query Syntax: `MEMory:MALLocate:PROGram?`

Return Format: `Integer`

Attribute Summary:

```
Option:  1C2
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Note This command is available only if the Instrument BASIC option (1C2) is installed. To determine whether or not the option is installed use the *OPT? query.

RAM may be used for measurements, data registers, math registers, and RAM disk, as well as for Instrument BASIC programs.

The limits specified in the command syntax define what numbers are syntactically correct. The *real* limits are defined by how much memory is installed and how much is presently unallocated. If you need more memory than is available, consider reducing the Max frequency points value, the number of data registers, the number of math temporary registers, or reducing the size of the RAM disk.

The standard amount of RAM is supplemented with a 20 MB RAM board (listed under the options as option UTH). The 20 MB RAM board includes additional I/O (another GPIB connector and a LAN interface).

Related Commands:

- To specify the Max number of frequency points, use MEM:MALL:MEAS:FPO.
- To specify the Max number of time points, use MEM:MALL:MEAS:TPO.
- To specify the number of data registers, use MEM:MALL:MEAS:REG.
- To specify the number of math temporary registers, use MEM:MALL:MEAS:MTEM.
- To specify the RAM disk size, use MMEM:INIT.
- To remove the RAM disk, use MEM:DEL RDIS.

MMEMory:COPY — command

Copies the contents of one disk to another or one file to another.

Command Syntax: `MMEMory:COPY <source>,<destination>`

```
<source>           ::=  '<device>[<filename>]'
<destination>      ::=  '<device>[<filename>]'
<device>           ::=  NVRAM:|RAM:|INT:|EXT[,<device_selector>[,<unit_number>]]:
<device_selector>  ::=  a real number (NRf data)
                          limits:  700:730
<unit_number>      ::=  a real number (NRf data)
                          limits:  0:3
<filename>         ::=  ASCII characters
                          (see MMEM:NAME for filename conventions)
```

Example Statements:

```
OUTPUT 719;"mmem:copy 'int:file1', 'ext:file1'"
OUTPUT 719;"MMEM:COPY 'EXT:', 'INT:'"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

To copy a file, include a filename with the source device. To copy a file and change the name, include filenames with both source and destination devices. <device> values are defined under MMEM:MSIS. If the destination filename is *not* specified, the source and destination *must* be different devices. Files may also be moved with the LAN file-transfer utility, FTP (option UG7).

To copy an entire disk, use the mass storage unit specifier (device) for the <source> and <destination> with no filenames. The device selector and unit number specifiers are valid only with the EXT: specifier. See MMEM:NAME for filename conventions and wildcard usage.

Caution **All files on the destination are overwritten by a disk copy operation.**

Related Commands:

- To rename a file, use the MMEM:MOVE command.
- To specify the default device, use MMEM:MSIS (mass storage is).
- To display directory listings, use SCR:CONT MMEM.

Notes When the control program instructs the analyzer to access the external mass storage device (EXT:), the controller must temporarily pass control (of the bus) to the analyzer. After the command has been executed, the analyzer passes control back. See *GPIB Programmer's Guide* for more on passing control.

Be aware of the filename and data format differences when copying between LIF and DOS file systems. The copy is performed but no conversions occur.

MMEMory:DATA **command/query**

Transfer a file between a disk and the GPIB port.

Command Syntax: `MMEMory:DATA '<device>[<filename>]'`

<device> ::= NVRAM:|RAM:|INT:

<filename> ::= ASCII characters
(see MMEM:NAME for filename conventions)

Example Statements:
OUTPUT 719;"mmem:data 'FILENAME' "
OUTPUT 719;"Mmem:Data 'INT:FILENAME' "

Query Syntax: `MMEMory:DATA? '<device>[<filename>]'`

Return Format: `DEF_USER`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Use this command to transfer a file from a disk to the GPIB port or from the GPIB port to a disk. Use MMEM:DATA to send a file to the GPIB port, use MMEM:DATA? to read a file from the GPIB port. The file is transferred as a definite-length block of data. For information on data block transfers, see "Block Parameters" in chapter 4 of the Programmer's Guide.

You cannot use this command to transfer a file to or from an external disk.

MMEMory:DELete **command**

Deletes one file or the entire contents of a disk.

Command Syntax: `MMEMory:DELete '<device>[<filename>]'`

```
<device>           ::=  NVRAM:|RAM:|INT:|EXT[,<device_selector>[,<unit_number>]]:
<device_selector>  ::=  a real number (NRf data)
                          limits:  700:730
<unit_number>      ::=  a real number (NRf data)
                          limits:  0:3
<filename>         ::=  ASCII characters
                          (see MMEM:NAME for filename conventions)
```

Example Statements:

```
OUTPUT 719;"MMEMory:DEL 'INT:JUNK'"
OUTPUT 719;"mmem:del 'ext:/path1/state.sta'"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

To delete all files from a mass storage device, specify only the disk (mass storage unit). The device selector and unit number specifiers are valid only with the EXT: disk specifier.

Wildcard characters are accepted in filenames, as in *.DAT. See MMEM:NAME for filename conventions and wildcard usage.

The <device> values are defined under MMEM:MSIS.

Related Commands:

- To rename a file, use the MMEM:MOVE command.
- To specify the default drive, use MMEM:MSIS (mass storage is).
- To display the directory contents, use SCR:CONT MMEM.

Note When the control program instructs the analyzer to access the external mass storage device (EXT:), the controller must temporarily pass control (of the bus) to the analyzer. After the command has been executed, the analyzer must pass control back. See *GPIB Programmer's Guide* for more on passing control.

MMEMory:FSYStem? query

Returns the type of file system for the default disk.

Query Syntax: `MMEMory:FSYStem?`

Example Statements:

```
OUTPUT 719;":MMEMORY:FSYS?"
OUTPUT 719;"mmem:fsystem?"
```

Return Format: `CHAR`

CHAR ::= LIF|DOS

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query returns the type of file system on the default disk.

Related Commands:

- To specify the default disk, use the MMEMory:MSIS command.
- To display the directory contents, use SCR:CONT MMEM.

MMEMory:INITialize **command**

Formats a mass storage unit.

Command Syntax: `MMEMory:INITialize [‘<device>’[,(LIF|DOS)[,<format>[,<i>]]]]`

<device>	::=	NVRAM:|RAM:|INT:|EXT[,<device_selector>[,<unit_number>]]:
<device_selector>	::=	a real number (NRf data) limits: 700:730
<unit_number>	::=	a real number (NRf data) limits: 0:3
<format>	::=	a real number (NRf data) limits: 0:(see discussion on <format>)
<i>	::=	a real number (NRf data) limits: 0:256

Example Statements:

```
OUTPUT 719;"mmem:init ‘NVRAM’, LIF"
OUTPUT 719;"MMEM:INIT ‘RAM:’, DOS, 100000"
OUTPUT 719;"mmemory:initialize ‘int:’, lif, 0, 3"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

<device> is the mass storage unit specifier; usually a disk drive, but also used to indicate RAM or NVRAM volumes (which perform as very fast drives) or any external mass storage device (EXT:).

<device_selector> and **<unit_number>** specifiers are valid only with the EXT: device specifier. A *device_selector* is composed of the *select_code* (usually 7) and the GPIB address (0-30). The *unit-number* specifies the drive within a drive unit as would be the case for a dual floppy drive.

LIF|DOS specifies the disk file system type. The default selection is DOS. LIF (logical interchange format) is used by most HP hard disks. DOS (disk operating system) is used on IBM personal computers. The RAM disk is formatted every time power is turned on, using a volume size stored in NVRAM and the same file system type as NVRAM.

Note If you plan to move data in NVRAM or RAM to an external system, it is best to initialize NVRAM and RAM disks with the same file system type (LIF or DOS) as the destination system uses to avoid having to convert filenames and data formats.

<format> specifies the tracks/sector-format for a flexible disk, either internal (INT:) or external (EXT:). The default value is 0. See table. For a hard drive, <format> should be 0.

- For flexible disk drives, the value determines the disk’s capacity as shown in the following table. For DOS disks, <format> should be 16. If DOS is specified and <format> is not, 16 is used.
- For device = NVRAM, <format> is ignored. See first example.
- For device = RAM, <format> specifies the amount of memory to allocate (for a RAM volume)

in bytes. See second example. The amount of RAM available is dependent on the amount of RAM installed, whether IBASIC and programs are loaded, the number of channels in use, the maximum number of frequency points, and the number of data registers. For information on memory configuration, see "Memory" in the Help Text.

<i> is the disk interleave value. It is ignored for RAM and NVRAM. Default is 1 for LIF; 3 for DOS.

Note When the control program instructs the analyzer to access the external mass storage device (EXT:), the controller must temporarily pass control (of the bus) to the analyzer. After the command has been executed, the analyzer passes control back. See *GPIB Programmer's Guide* for more on passing control.

Flexible Disk Format Options

Media	< format >	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 (DOS DSDD)	512	9	80	737,280
2-MByte	0	256	32	77	1,261,568
	1‡	256	32	77	1,261,568
	2	512	18	77	1,419,264
	3	1,024	10	77	1,576,960
	4‡	256	32	77	1,261,568
	16 (DOS HD)	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 media.

Note Don't confuse the colons (:) used in SCPI syntax with those of the device names. SCPI uses colons to separate command branches while the device names use a colon to separate the device name from the path name and/or file name.

Related Commands:

- Use MMEM:FSYS? to query the existing file systems type.
- You may change the current/default device with the MMEM:MSIS command and then initialize without specifying a device.
- See the MEMory commands.

MMEMory:LOAD:APPLication command

Loads the specified application.

Command Syntax:

```
MMEMory:LOAD:APPLication '[<msus>]<filespec>'
    <msus>    ::= RAM:|NVRAM:|INT:|EXT:
    <filespec>    ::= ASCII characters
(see MMEM:NAME for filename restrictions)
```

Example Statements:

OUTPUT 719;"mmem:load:application 'int:hp89450a.app'"
OUTPUT 719;"MMEM:LOAD:APPL 'HP89450A.APP'"

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads an application (such as the Agilent 89450A Radio Test Personality) into the analyzer. The application is loaded into the analyzer's application memory. Before loading the application, you must allocate sufficient memory for the application—for details, see MEM:MALL:APPL.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

MMEMory:LOAD:CONTinue command/query

Continues the load operation of time capture files stored on multiple disks.

Command Syntax: `MMEMory:LOAD:CONTinue`

Example Statements:
OUTPUT 719;"Mmem:Load:Cont"
OUTPUT 719;"MMEMORY:LOAD:CONT"

Query Syntax: `MMEMory:LOAD:CONTinue?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Time capture files that are too large to fit on the current disk may be split and stored on more than one disk with the MMEM:STOR:CONT command. The MMEM:LOAD:CONT command is used to load these split files back into the analyzer.

When the analyzer has completed loading the first part of the split file (for example, filename_1), it generates the message, ''Media full; Insert next disk with 'filename_2'.'' Insert the disk containing the file named filename_2 and send MMEM:LOAD:CONT to continue loading the file.

Use MMEM:LOAD:CONT? to verify that the file has been transferred. If the query returns a +1, the transfer is not complete. The query returns a 0 when the transfer is complete.

When a trace buffer or time capture file is merged, you are prompted to change media at the end of each file. Pressing the Local/Setup key (or *any* key) on the front panel in Local mode causes the file merging operation to terminate. SCPI commands, however, sent between operations do *not* terminate the file operation.

Large files may also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

For more information on the generation of split files, see the MMEM:STOR:CONT command.

MMEMory:LOAD:MATH command

Loads a complete set of math definitions into the analyzer from the specified file.

Command Syntax: `MMEMory:LOAD:MATH '[<msus>]<filespec>'`

<msus> ::= RAM:|NVRAM:|INT:|EXT:

<filespec> ::= ASCII characters
(see MMEM:NAME for filename restrictions)

Example Statements:

```
OUTPUT 719;"Mmem:Load:Math 'EXT:math'"
OUTPUT 719;"MMEMORY:LOAD:MATH 'RAM:math1'"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads a file to define the analyzer's math functions (F1 through F6) and constants (K1 through K5). The file must have been created with the MMEM:STOR:MATH command.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

The math function definitions can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

MMEMory:LOAD:PROGram command

Loads an Instrument BASIC program into the analyzer from a file on the mass storage unit specified (msus).

Command Syntax: `MMEMory:LOAD:PROGram '[<msus>]<filespec>'`

<msus>	::=	RAM:|NVRAM:|INT:|EXT:
<filespec>	::=	ASCII characters (see MMEM:NAME for filename restrictions)

Example Statements:

```
OUTPUT 719;"Mmem:Load:Prog 'myprog'"
OUTPUT 719;"MMEMORY:LOAD:PROG 'RAM:prog1'"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads an Instrument BASIC program into the program buffer previously selected with the PROG:NAME command. Only one program may be loaded at a time (there is only one program buffer).

- To load an Instrument BASIC program directly from the controller, use PROG[:SEL]:DEF.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

MMEMory:LOAD:STATe — command

Loads an instrument state into the analyzer from the mass storage unit specified (msus).

Command Syntax: `MMEMory:LOAD:STATe <number>,<filename>`

<number> ::= a real number (state number)
limits: 1:1

<filename> ::= ‘[<msus>]<filespec>’

<msus> ::= RAM:|NVRAM:|INT:|EXT:

<filespec> ::= ASCII characters
(see MMEM:NAME for filename restrictions)

Example Statements:

OUTPUT 719;"mmemory:load:state 1, ‘ext:mystate’"
OUTPUT 719;"MMEM:LOAD:STAT 1, ‘INT:IFsetup.sta’"

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command uses the contents of a file to redefine the instrument state. The file can only be created by saving an existing state with the MMEM:STOR:STAT command.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

The instrument state can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

MMEMory:LOAD:TCAPture **command**

Loads a time capture file from the mass storage unit specified (msus).

Command Syntax: `MMEMory:STORe:TCAPture '[<msus>]<filespec>'`

<msus> ::= RAM:|NVRAM:|INT:|EXT:

<filespec> ::= ASCII characters
(see MMEM:NAME for filename restrictions)

Example Statements:
OUTPUT 719;"mmem:load:tcap, 'int:tcap-01'"
OUTPUT 719;"MMEM:LOAD:TCAPTURE, 'TIME-REF'"

Attribute Summary:
```
Synchronization Required:  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 created by saving with the MMEM:STOR:TCAP command.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

The time capture buffer data can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

Related Commands:

- To display the time capture buffer in one of the four traces, use the CALC:FEED 'TCAP[1|2|3|4]' command.
- To continue loading a file stored on more than one disk, use the MMEM:LOAD:CONT command.

MMEMory:LOAD:TRACe **command**

Loads a trace into the analyzer from the mass storage unit specified.

Command Syntax: `MMEMory:LOAD:TRACe <label>,<filename>`

<label> ::= {D1|D2|D3|D4|D5|D6|}
<filename> ::= '[<msus>]<filespec>'
<msus> ::= RAM:|NVRAM:|INT:|EXT:
<filespec> ::= ASCII characters
(see MMEM:NAME for filename restrictions)

Example Statements:

```
OUTPUT 719;"mmem:load:trac D1, 'int:test.trc'"
OUTPUT 719;"MMEM:LOAD:TRACE D3, 'MYTRACE'"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command loads the contents of a file into one of the analyzer's six data registers. The first parameter specifies the destination. The second parameter specifies the source.

The file must have been created by saving a trace with the MMEM:STOR:TRAC command. After loading the data register, it is displayed with the CALC:FEED 'D{1|2...|6}' command.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

The trace data can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

MMEMory:LOAD:TRACe:BUFFer **command**

Loads a waterfall/spectrogram trace buffer into the analyzer from the mass storage unit specified (msus).

Command Syntax: `MMEMory:LOAD:TRACe:BUFFer <label>,<filename>`

<label>	::=	{D1|D2|D3|D4|D5|D6|}
<filename>	::=	'[<msus>]<filespec>'
<msus>	::=	RAM:|NVRAM:|INT:|EXT:
<filespec>	::=	ASCII characters (see MMEM:NAME for filename restrictions)

Example Statements:

OUTPUT 719;"mmem:load:trac:buff D1, 'int:test.trc'"
OUTPUT 719;"MMEM:LOAD:TRACE:BUFFER D3, 'MYTRACE'"

Attribute Summary:

```
Option:  AYB (waterfall/spectrogram display)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command loads the contents of a file into one of the analyzer's six data registers. The first parameter specifies the destination. The second parameter specifies the source.

The file must have been created by saving a trace with the MMEM:STOR:TRAC:BUFF command. After loading the data register, it is displayed with the CALC:FEED 'D{1|2...|6}' command.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

This command differs from the TRAC:BUF:COPY command in that MMEM:LOAD:TRAC:BUF loads data into a data register from a file and TRAC:BUF:COPY copies trace data to one of the data registers.

The waterfall/spectrogram data can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

MMEMory:MOVE command

Moves a file from one device to another or renames a file.

Command Syntax: `MMEMory:MOVE <source>,<destination>`

<source>	::=	'[<device>]<filename>'
<destination>	::=	'[<device>]<filename>'
<device>	::=	NVRAM:|RAM:|INT:|EXT[,<device_selector>[,<unit_number>]]:
<device_selector>	::=	a real number (NRf data) limits: 700:730
<unit_number>	::=	a real number (NRf data) limits: 0:3
<filename>	::=	ASCII characters (see MMEM:NAME for information on filename conventions)

Example Statements:

```
OUTPUT 719;"mmemory:move 'int:file1', 'myfile'"
OUTPUT 719;"MMEM:MOVE 'TESTFILE', 'FILE3'"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command may be used to copy one or more files to another device or directory, automatically deleting the source file(s), or to rename a file. It may be also be used to move a file *and* rename it. Wildcards may be used to specify filenames in the <source> but not in the <destination>. See MMEM:NAME for filename conventions and wildcard usage.

The <device> values are defined under MMEM:MSIS.

Related Commands:

- Use MMEM:MSIS (mass storage is) to specify the default device.
- Use MMEM:COPY to copy files from one device and/or path to another without deleting the source.
- Use SCR:CONT MMEM to display directory listings.

Note When the control program instructs the analyzer to access the external mass storage device (EXT:), the controller must temporarily pass control (of the bus) to the analyzer. After the command has been executed, the analyzer must pass control back. See *GPIB Programmer's Guide* for more on passing control.

MMEMory:MSIS **command/query**

Specifies a default mass storage device (disk) and (for DOS) directory.

Command Syntax: `MMEMory:MSIS '<device>'`

<device>	::=	NVRAM:|RAM:|INT:|EXT[,<device_selector>[,<unit_number>]]:
<device_selector>	::=	a real number (NRf data) limits: 700:730
<unit_number>	::=	a real number (NRf data) limits: 0:3

Example Statements:

```
OUTPUT 719;":Mmem:Msis 'INT:'"
OUTPUT 719;"MMEMORY:MSIS 'RAM:'"
OUTPUT 719;"mmem:msis 'int:/data/friday'"
```

Query Syntax: `MMEMory:MSIS?`

Return Format: `STRING`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  saved in non-volatile memory
SCPI Compliance:  confirmed
```

Description:

<device> is the mass storage unit specifier; usually a disk drive, but also used to indicate RAM or NVRAM volumes (which perform as very fast drives) or any external mass storage device (EXT:). For DOS file systems, a directory may also be specified (see third example). The device types are:

NVRAM: selects the *non-volatile* RAM drive.

RAM: selects the *volatile* RAM drive.

INT: selects the *internal* drive.

EXT: selects the *external* drive.

<device_selector> and **<unit_number>** specifiers are valid only with the EXT: device specifier. A *device_selector* is composed of the *select_code* (usually 7) and the GPIB address (0-30). The *unit-number* specifies the subunit within a volume as for a dual floppy drive. If device_selector and unit_number are specified with this command, they become the *default* for the external (EXT:) specifier.

Related Commands:

- To determine the type of file system for the default disk, send the MMEM:FSYStem? query.
- To display the directory contents, use SCR:CONT MMEM.

Note When the control program instructs the analyzer to access the external mass storage device (EXT:), the controller must temporarily pass control (of the bus) to the analyzer. After the command has been executed, the analyzer must pass control back. See *GPIB Programmer's Guide* for more on passing control.

MMEMory:NAME **command/query**

Specifies the filename for printing or plotting to a file.

Command Syntax: MMEMory:NAME ‘[<device>]<filename>’

<device> ::= RAM:|NVRAM:|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 description for <filename> conventions)

Example Statements:

```
OUTPUT 719;":Mmemory:Name ‘plot1.hpg’"
OUTPUT 719;"MMEM:NAME ‘INT:3RD-IF_PCL’"
```

Query Syntax: MMEMory:NAME?

Return Format: STRING

Attribute Summary:

```
Synchronization Required:  no
Preset State:  “plotfile”
SCPI Compliance:  confirmed
```

Description:

This command is used to specify a filename when printing or plotting to a file. Specifying FILE as the output destination is done with the HCOP:DEST FILE command.

If no mass storage unit specifier (msus) is given, the device specified with the last MSIS (mass storage is) command is used. If MSIS has not been used since the last preset, then msus = NVRAM (non-volatile RAM).

If EXT is specified, control of the bus must be passed to the analyzer.

Filename and Wildcard Conventions:

DOS filenames are limited in length to 8 ASCII characters followed by a period and an (optional) extension of one to three characters. Both the period and extension are optional. The period is not part of the filename; it separates the name and extension. DOS filenames are not case sensitive. Valid special characters are _, ^, $, ~, !, #, %, &, -, {, }, (,), @, ‘, ‘. A name may not contain spaces, commas, backslashes, or periods (except the period that separates the name and extension). Reserved names that may not be used are CLOCK$, CON, AUX, COM*n* (*n* = 1–4), LPT*n* (*n* = 1–3), NUL, and PRN.

LIF filenames are limited in length to 10 characters which may include any character *except* :, <, and |. The first character must be a letter. LIF filenames are case sensitive.

Wildcards allow you to identify more than one file when a filename is used as a command’s argument, such as for copy or delete. A wildcard character can match part or all of the filename. Two characters are used as wildcards: ? matches any single character and * matches all characters to the end of the name. To match all filenames starting with B, use B*. You can use the asterisk to identify files that have common first characters, but not files with matching character patterns in the middle or end of the name. Characters that appear after the * are ignored because the asterisk is expanded to all possible combinations of the question-mark wildcard(s) to the end of the name. For example, *ANT matches all filenames, just as * would (see following table). Question-mark wildcards may be used if the filename pattern to be matched has a fixed number of characters that vary.

Wildcards work differently for DOS than for LIF because DOS has two fields to use wildcards in; the filename and the extension. To match all files in a DOS system, use *.*; in LIF just * does it. In DOS, a single * matches all filenames that have no extension. To match all files (in DOS) with a particular extension, use * (for the filename) and the extension, as in *.DAT. Remember that, in LIF, the period character (.) is just another ASCII character, may be used more than once in a file, and may be matched by a wildcard. In DOS, a period cannot be matched by a wildcard because it isn't part of the filename.

Wildcard Examples

Wildcard Pattern	DOS FIle Matches	LIF File Matchs
.	all files	all files
*	all files without extensions	all files
2.	all files	all files
*2	files that have no extension	all files
B*	files beginning with B	files beginning with B
*.DAT	files with the extension DAT	all files
???.??	files with a 3-character name and 2-char extension	files with a 6-character name, 4th character is .
???2	files with 4-character name, last character is 2	files with 4-character name, last character is 2
*.?	files with a 1-character extension	all files

MMEMory:STORe:CONTinue **command/query**

Allows a large time capture file to be split over multiple disks.

Command Syntax: `MMEMory:STORe:CONTinue`

Example Statements:
OUTPUT 719;":mmemory:stor:continue"
OUTPUT 719;"Mmem:Stor:Continue"

Query Syntax: `MMEMory:STORe:CONTinue?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If the amount of time capture data to be stored is more than the available space on the destination disk, the error message, ‘’Media full; File too large’’ is generated. This error message is generated before the store operation begins. Send the MMEM:STOR:CONT message to begin the store operation.

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:STOR:CONT? returns a +1 when queried before the transfer is complete. Insert another disk and send MMEM:STOR:CONT to continue the store operation. The query returns a 0 when the entire file has been stored.

When a trace buffer or time capture file is split, you are prompted to change media at the end of each file. Pressing the Local/Setup key (or *any* key) on the front panel in Local mode causes the file splitting operation to terminate. SCPI commands, however, sent between operations do *not* terminate the file operation.

Large files may also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

MMEMory:STORe:MATH command

Saves a complete set of math definitions to the mass storage unit specified (msus).

Command Syntax: `MMEMory:STORe:MATH '[<msus>]<filespec>'`

<msus>	::=	RAM:|NVRAM:|INT:|EXT:
<filespec>	::=	ASCII characters (see MMEM:NAME for filename restrictions)

Example Statements:

```
OUTPUT 719;"Mmem;Store:Math 'EXT:math'"
OUTPUT 719;"MMEMORY:STOR:MATH 'RAM:math1'"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

A complete set of math definitions includes the current values in all of the function registers (F1 through F6) and all of the constant registers (K1 through k5). If the filename you specify with this command matches the name of another file on the mass storage unit specified, the existing file is overwritten.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

The math function definitions can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

MMEMory:STORe:PROGram command

Stores an Instrument BASIC program to a file on the specified disk.

Command Syntax: `MMEMory:STORe:PROGram '[<msus>]<filespec>'`

<msus> ::= RAM:|NVRAM:|INT:|EXT:

<filespec> ::= ASCII characters
(see MMEM:NAME for filename restrictions)

Example Statements:
OUTPUT 719;"Mmem:Stor:Prog 'EXT:ibprog'"
OUTPUT 719;"MMEMORY:STORE:PROGRAM 'int:myprog'"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command stores the currently active Instrument BASIC program to the mass storage unit specified. 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 the error message, "Program Error, No program exists."

If the specified filename is the same as an existing file, the analyzer generates the error message, "Program Error, Duplicate file name," and it aborts the store operation. This command will not overwrite an existing file. See the MMEM:DEL command for information about deleting existing files from a disk.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

MMEMory:STORe:STATe **command**

Saves the instrument state to a file on the specified mass storage unit.

Command Syntax: `MMEMory:STORe:STATe <number>,<filename>`

<number>	::=	a real number (state number) limits: 1:1
<filename>	::=	'[<msus>]<filespec>'
<msus>	::=	RAM:|NVRAM:|INT:|EXT:
<filespec>	::=	ASCII characters (see MMEM:NAME for filename restrictions)

Example Statements:

```
OUTPUT 719;"mmemory:store:state 1, 'ext:mystate'"
OUTPUT 719;"MMEM:STOR:STAT 1, 'INT:IFsetup.sta'"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

If the filename matches that of another file on the mass storage unit specified (*msus*), this command overwrites the existing file.

If *msus* is not included in the syntax, the currently selected mass storage unit (disk) is assumed.

The instrument state can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

Related Commands:

- To query or change the currently selected msus, use MMEM:MSIS (mass storage is).

MMEMory:STORe:TCAPture **command**

Stores the time capture buffer to a file on the mass storage unit specified (msus).

Command Syntax: `MMEMory:STORe:TCAPture '[<msus>]<filespec>'`

<msus> ::= RAM:|NVRAM:|INT:|EXT:

<filespec> ::= ASCII characters
(see MMEM:NAME for filename restrictions)

Example Statements:
OUTPUT 719;"mmem:stor:tcap, 'int:tcap-01'"
OUTPUT 719;"MMEM:STORE:TCAPTURE, 'TIME-REF'"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

If the contents of the time capture buffer is too large for the mass storage unit, the error message ''Media full; File too large'' is generated. Use the MMEM:STOR:CONT command to split the file over multiple disks. See MMEM:STOR:CONT command for more information about splitting a time capture file on multiple disks.

If the filename matches the name of another file on the mass storage unit, this command overwrites the old file.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

The time capture buffer data can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

Note If you plan to transfer this file to a PC, refer to the *Standard Data Format Utilities Users Guide* for information on format conversion.

MMEMory:STORe:TRACe **command**

Saves the trace to a file on the mass storage unit specified (msus).

Command Syntax: `MMEMory:STORe:TRACe <label>,<filename>`

<label>	::=	{TRACe1|TRACe2|TRACe3|TRACe4}
<filename>	::=	'[<msus>]<filespec>'
<msus>	::=	RAM:|NVRAM:|INT:|EXT:
<filespec>	::=	ASCII characters (see MMEM:NAME for filename restrictions)

Example Statements:

```
OUTPUT 719;"mmem:stor:trac trac1, 'int:test.trc'"
OUTPUT 719;"MMEM:STORE:TRACE TRACE3, 'MYTRACE'"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The first parameter specifies which trace is stored. The second parameter specifies the mass storage unit (disk) and filename. If the filename matches the name of another file on the disk, this command overwrites the existing file.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

This command differs from the TRAC:DATA command in that MMEM:STOR:TRAC stores data to a file and TRAC:DATA stores trace data to one of the data registers.

The trace data can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

Note If you plan to transfer this file to a PC, refer to the *Standard Data Format Utilities Users Guide* for information on format conversion.

MMEMory:STORe:TRACe:BUFFer **command**

Saves the waterfall/spectrogram trace buffer to a file on the mass storage unit specified (msus).

Command Syntax: `MMEMory:STORe:TRACe:BUFFer <label>,<filename>`

```
<label>    ::= {TRACe1|TRACe2|TRACe3|TRACe4}
<filename> ::= '[<msus>]<filespec>'
<msus>     ::= RAM:|NVRAM:|INT:|EXT:
<filespec> ::= ASCII characters
               (see MMEM:NAME for filename restrictions)
```

Example Statements:

```
OUTPUT 719;"mmem:stor:trac;buff trac1, 'int:test.buf'"
OUTPUT 719;"MMEM:STORE:TRACE:BUFF TRACE3, 'SPECTRO1'"
```

Attribute Summary:

```
Option:  AYB (waterfall/spectrogram display)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The first parameter specifies which trace buffer is stored. The second parameter specifies the mass storage unit (disk) and filename. If the filename matches the name of another file on the disk, this command overwrites the existing file.

If <msus> is not included in the syntax, the currently selected mass storage unit (disk) is assumed. To query or change the currently selected msus, use MMEM:MSIS.

This command differs from the TRAC:BUF:COPY command in that MMEM:STOR:TRAC:BUF stores data to a file and TRAC:BUF:COPY stores trace data to one of the data registers.

The waterfall/spectrogram data can also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

Note If you plan to transfer this file to a PC, refer to the *Standard Data Format Utilities Users Guide* for information on format conversion.

OUTPut:FILTer[:LPASs][:STATe] command/query

Turns the source output lowpass filter on and off.

Command Syntax: `OUTPut:FILTer[:LPASs][:STATe] OFF|0|ON|1`

Example Statements:
```
OUTPUT 719;"OUTP:FILTER:LPAS:STAT OFF"
OUTPUT 719;"output:filt OFF"
```

Query Syntax: `OUTPut:FILTer[:LPASs][:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1
SCPI Compliance:  confirmed
```

Description:

This command is available only for the 89410A or for the 89441A when ROUT:REC is IF.

This 10 MHz low-pass filter limits the bandwidth of the source output signal as the final step in converting a digital signal to an analog signal. Some arbitrary source signals may require that this filter be removed.

OUTPut:IMPedance command/query

Selects the output termination impedance.

Command Syntax: `OUTPut:IMPedance <number>`

<number> ::= 50 | 75

Example Statements:
OUTPUT 719;"Outp:Imp 50"
OUTPUT 719;"outp:imp 75"

Query Syntax: `OUTPut:IMPedance?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  50
SCPI Compliance:  confirmed
```

Description:

This command is available for the 89441A only when ROUT:REC is IF.

OUTPut[:STATe] command/query

Turns the source output on and off.

Command Syntax: `OUTPut[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Output ON"
OUTPUT 719;"OUTP:STAT OFF"

Query Syntax: `OUTPut[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  confirmed
```

Description:

The source is off when the analyzer is first turned on and is turned off whenever the analyzer is preset (with the *RST command).

- Source type is selected with the SOUR:FUNC command.
- Source level is set with the SOUR:VOLT command.

Note It is good practice to specify the source level *before* the source is turned on. When the source is turned on, the output voltage level is whatever value was active when it was turned off. At turn-on and preset, the level is –10 dBm (100 mV). The active value may be large enough to damage sensitive devices.

PAUSe **command**

Pause the measurement in progress.

Command Syntax: `PAUSe`

Example Statements:

```
OUTPUT 719;"pause"
OUTPUT 719;"Paus"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Characteristics:

- Pauses a measurement process.
- In Vector or Demod modes, current average is completed before pause occurs.
- In Scalar mode, analyzer may pause in the middle of a sweep, even if averaging is on.
- Data is not affected by PAUSE. It is valid and may be manipulated (with CALC:FORM, CALC:FEED, or CALC:MATH).
- CONT restarts a paused measurement without affecting data.
- ABORT, ABORT;INIT, and *RST destroy existing data and restart a paused measurement.
- A paused measurement does not respond to INIT:IMM.

PROGram:EXPLicit:DEFine **command/query**

Loads an Instrument BASIC program into a program buffer from an external controller.

Command Syntax: `PROGram:EXPLicit:DEFine <prog_list>,<program>`

<prog_list> ::= {PROGram1 }
<program> ::= <block>

Example Statements: For an indefinite-length block:

```
OUTPUT 719;"PROG:EXPL:DEF PROG1,#0";
OUTPUT 719;"10 PRINT “”HELLO WORLD"""&CHR$(10);
OUTPUT 719;"20 END"&CHR$(10) END
```

For a definite-length block:

```
OUTPUT 719;"PROG:EXPL:DEF PROG1,#230";
OUTPUT 719;"10 PRINT “”HELLO WORLD"""&CHR$(10);
OUTPUT 719;"20 END"&CHR$(10);
```

Query Syntax: `PROGram:EXPLicit:DEFine?`

Return Format: `definite-length block`

Attribute Summary:

```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command transfers a program between the analyzer and the controller. This allows programs developed on a controller to be loaded into the analyzer. The first parameter specifies the analyzer’s program buffer (the destination), which becomes the active program buffer. The second parameter is the Instrument BASIC program.

A program may be transferred to the analyzer with either the definite-length or the indefinite-length block syntax. The *simplest* way to load an Instrument BASIC program 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 GPIB END message, EOI set true). See example above.

When the analyzer returns the program to your controller, it always uses the definite-length block syntax. See ‘’Block Data” in the *GPIB Programmer’s Guide* for more information.

PROGram[:SELected]:DEFine command/query

Loads an Instrument BASIC program from an external controller into the active program buffer.

Command Syntax: `PROGram[:SELected]:DEFine <program>`

<program> ::= <block>

Example Statements: For an indefinite-length block:

```
OUTPUT 719;"PROG:DEF #0";
OUTPUT 719;"10 PRINT “”HELLO WORLD"""&CHR$(10);
OUTPUT 719;"20 END"&CHR$(10) END
```

For a definite-length block:

```
OUTPUT 719;"PROG:DEF #230";
OUTPUT 719;"10 PRINT “”HELLO WORLD"""&CHR$(10);
OUTPUT 719;"20 END"&CHR$(10);
```

Query Syntax: `PROGram[:SELected]:DEFine?`

Return Format: `definite-length block`

Attribute Summary:

```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command transfers a program between the analyzer and the controller. This allows programs developed on a controller to be loaded into the analyzer.

Use the PROG[:SEL]:NAME to select the active program buffer.

A program may be transferred to the analyzer with either the definite-length or the indefinite-length block syntax. The *simplest* way to load an Instrument BASIC program 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 GPIB END message, EOI set true). See example above.

When the analyzer returns the program to your controller, it always uses the definite-length block syntax. See ‘‘Block Data” in the *GPIB Programmer’s Guide* for more information.

PROGram[:SELected]:DELete:ALL — command

Deletes all Instrument BASIC programs stored in the analyzer.

Command Syntax: `PROGram[:SELected]:DELete:ALL`

Example Statements:

```
OUTPUT 719;":PROGRAM:DEL:ALL"
OUTPUT 719;"program:sel:delete:all"
```

Attribute Summary:

```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

In addition to deleting the active program, this command deletes *all* of the resident Instrument BASIC programs. Program variables—both those in COM and those not in COM are deleted as well.

PROGram[:SELected]:DELete[:SELected] command

Deletes the active Instrument BASIC program.

Command Syntax: `PROGram[:SELected]:DELete[:SELected]`

Example Statements:

```
OUTPUT 719;"Prog:Selected:Del:Selected"
OUTPUT 719;"PROG:DEL"
```

Attribute Summary:

```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  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.

PROGram[:SELected]:MALLocate command/query

Allocates memory space for Instrument BASIC programs.

Command Syntax: `PROGram[:SELected]:MALLocate {<number>|<bound>|DEFault}`

<number> ::= a real number (NRf data)
limits: 1200:(see discussion)

<bound> ::= MAX|MIN

Example Statements:

OUTPUT 719;"program:sel:mall 134987"
OUTPUT 719;"Prog:Mall 250982"

Query Syntax: `PROGram[:SELected]:MALLocate?`

Return Format: `Integer`

Attribute Summary:

```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Upper limit depends on the amount of memory available for IBASIC use. Largest number parsed is 500000.

PROG:MALL DEF causes the analyzer to resize the stack space to fit the current active program. In some cases, the analyzer may allocate more memory than the program needs.

Use MEM:FREE to determine current memory availability and usage.

Note If the message, ‘‘ERROR 2 Memory overflow’’ is displayed while the program is running, allocate more memory.

PROGram[:SELected]:NAME command/query

Selects an Instrument BASIC program.

Command Syntax: `PROGram[:SELected]:NAME <prog_list>`

<prog_list> ::= {PROGram1 }

Example Statements:
OUTPUT 719;"Program:Name PROG1"
OUTPUT 719;"PROGRAM:SEL:NAME PROGram1"

Query Syntax: `PROGram[:SELected]:NAME?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Use this command to designate an Instrument BASIC program buffer as the ‘‘active’’ program buffer.

For example, use this command to select a program buffer when you load an 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 Instrument BASIC program.

Command Syntax: `PROGram[:SELected]:NUMBer '<variable>', <block>`

<variable> ::= name of a numeric variable

When data is ASCII-encoded (FORM ASC):

<block> ::= <number>[,<number>] . . .

<number> ::= a real number (NRf data)
limits: -9.9e37:9.9e37

When data is binary-encoded (FORM REAL):

<block> ::= #<byte>[<length>]<number>[,<number>] . . .

<byte> ::= number of length bytes to follow (ASCII encoded)

<length> ::= number of data bytes to follow (ASCII encoded)

<number> ::= a real number (or 64-bit binary floating point)

Example Statements: For ASCII blocks:

```
OUTPUT 719;"FORM ASC"
OUTPUT 719;"PROG:NUMB 'A', 12.345"
```

For real, indefinite-length blocks:

```
ASSIGN @Dev TO 719
OUTPUT @Dev;"FORM REAL"
ASSIGN @Dev;FORMAT OFF
OUTPUT @Dev;"PROG:NUMB 'A', #0";12.345 END
ASSIGN @Dev;FORMAT ON
```

For real, definite-length blocks:

```
ASSIGN @Dev TO 719
OUTPUT @Dev;"FORM REAL"
ASSIGN @Dev;FORMAT OFF
OUTPUT @Dev;"PROG:NUMB 'A', #18";12.345
ASSIGN @Dev;FORMAT ON
```

Query Syntax: `PROGram[:SELected]:NUMBer?`

Return Format: `definite-length block`

Attribute Summary:

```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

Set or query the value of numeric variables from the currently selected IBASIC program. These variables can be either arrays or single values. If the specified variable is not defined in the current program, the error "Illegal variable name" (-283) is generated. The second parameter in a set command is a list of values for the variable or array to be assigned. This list may either be ASCII numbers separated by commas or a definite or indefinite block. The response is generated in the current format (specified by the FORMat[:DATA] command).

When an array is loaded with this command, values in the <block> parameter are loaded into the 1st through nth elements of the array (where *n* is number of values in the block).

Related Commands:

- To specify the active program, use PROG:NAME.
- To load string variables, use PROG:STR.

PROGram[:SELected]:STATe command/query

Selects the state of the active Instrument BASIC program.

Command Syntax: `PROGram[:SELected]:STATe STOP|PAUSe|RUN|CONTinue`

Example Statements:

```
OUTPUT 719;":program:stat STOP"
OUTPUT 719;"Program:Sel:Stat PAUSE"
```

Query Syntax: `PROGram[:SELected]:STATe?`

Return Format: `CHAR`

Attribute Summary:

```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This command is used to run, pause, stop or continue the active 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 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 719;"PROG:SELECTED:STR 'A$', 'Done'"
OUTPUT 719;"prog:string 'Message$','Measuring'"
```

Query Syntax: `PROGram[:SELected]:STRing?`

Return Format: `<string>`

Attribute Summary:

```
Option:  1C2 Agilent Technologies Instrument BASIC
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

This command sets or queries the contents of a string variable in the active Instrument BASIC program. Use the PROG:NAME to designate the active program.

Use the PROG[:SEL]:NUMB command to load or query numeric variables.

ROUTe:RECeiver command/query

Specifies receiver type.

Command Syntax: `ROUTe:RECeiver INPut|RF1|RF2|IF|COMBine|EXTernal`

Example Statements:
OUTPUT 719;"rout:receiver INPUT"
OUTPUT 719;"rout:receiver COMBINE"
OUTPUT 719;":Route:Rec EXT"

Query Syntax: `ROUTe:RECeiver?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  (see COMBine discussion)
Synchronization Required:  no
Preset State:  89410: INP, 89441: RF2
SCPI Compliance:  instrument-specific
```

Description:

INPut specifies the configuration in which the signal to be measured is connected to the front-panel connector labeled Channel 1. This is the default selection for the 89410A. It is valid only for the 89410A.

RF2 specifies the configuration in which the signal to be measured is connected to the input connector of the RF Section and the spectrum of interest is between 2 MHz and 2.650 GHz for the 89441A. This is the default selection for the 89441A. It is valid only for the 89441A.

RF1 specifies the configuration in which the signal to be measured is connected to the input connector of the RF Section and the spectrum of interest is between 0 Hz and 10 MHz. This is similar to the IF configuration but offers the advantage of not changing the input cable connections. For best 75Ω impedance matching and calibration of measurement results, use IF. This selection is valid only for the 89441A.

IF specifies the configuration in which the signal to be measured is connected to the Channel 1 connector of the IF Section and the spectrum of interest is between 0 Hz and 10 MHz. This is similar to the RF1 configuration but offers the advantage of better matching for 75Ω impedances (as opposed to using a minimum loss pad on the RF Section) and better calibration of measurement results. It's disadvantage is that it requires reconfiguring the input cabling if you were previously making measurements above 10 MHz. This selection is valid only for the 89441A.

COMBine specifies the configuration in which data from both input channels are combined as I + jQ (Ch1 + j*Ch2). This feature is part of option AYA (vector analysis/digital demod) and requires the presence of option AY7 (the second 10 MHz input channel). For more information on I + jQ, see online help (press the [**Help**] key, press **1** to show the index, use the knob or arrow keys to move through the list to **input section (Ch1+jCh2)**, then press **4** to jump to the discussion).

EXTernal specifies the configuration in which the input comes from an external downconverter such as the Agilent 89411A. This configuration has more commands under [SENS:]FREQ:EXT.

SCReen:CONTents **command/query**

Specifies what is displayed on the analyzer screen.

Command Syntax: `SCReen:CONTents TRACe|MSTate|IState|MMEMory|MEMory|OPTions|TCAPture`

Example Statements:
OUTPUT 719;"SCREEN:CONT TCAPTURE"
OUTPUT 719;"screen:cont TRACE"

Query Syntax: `SCReen:CONTents?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  TRAC
SCPI Compliance:  instrument-specific
```

Description:

TRACe displays the standard measurement trace/grid.

MSTate displays the measurement state display. This shows the instrument mode, frequency parameters, RBW/window settings, etc.

IState displays the instrument state display. This shows the range/input, trigger, and source settings.

MMEMory displays the current disk (mass memory) catalog.

MEMory displays the active memory configuration.

OPTions displays the options configuration.

TCAPture displays the time capture statistics.

[SENSe:]AVERage:COUNt — command/query

Specifies the number of traces to be averaged or the weighting factor for exponential averaging.

Command Syntax: `[SENSe:]AVERage:COUNt <number>|<step>|<bound>`

<number> ::= an integer (NRf data)
limits: 1:99999

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":AVERAGE:COUN 100"
OUTPUT 719;"sens:average:coun 20"

Query Syntax: `[SENSe:]AVERage:COUNt?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  10
SCPI Compliance:  confirmed 1992.0
```

Description:

Averaging overview:

- Averaging is turned on with the AVER ON command.
- AVER:IRES turns fast averaging on and off.
- AVER:IRES:RATE specifies the fast averaging update rate.
- AVER:TYPE specifies the average type: RMS (power), complex (time), or max (peak hold).
- AVER:TCON specifies termination control types which are normal, exponential, or repeat.
  - For normal averaging (AVER:TCON NORM), AVER:COUN specifies the number of measurement results that are averaged together. When the specified number of measurements have been taken and averaged together, the analyzer quits taking data and sets the NPO flag (No Pending Operations) to 1.
  - If repeat averaging is on (AVER:TCON REP), the averaging process restarts immediately after the current average is completed.
  - For exponential averaging (AVER:TCON EXP), AVER:COUN specifies a weighting factor which determines how the results of previous (averaged) measurement data (AVG_{i-1}) is combined with the current measurement data (X_i). The data is combined point-by-point as described by:

$$AVG_i = \frac{1}{n} X_i + \frac{n-1}{n} AVG_{i-1}$$

where n is the weighting factor specified by AVER:COUN. Note that for small values of n, the new data is a significant part of the average; for large values of n, new data has much less weight in the calculated average.

Note For exponential averaging, the averaging process does not stop after n measurements. The NPO flag is set to 1 after n measurements.

[SENSe:]AVERage:COUNt:INTermediate? query

Returns the current average count.

Query Syntax: `[SENSe:]AVERage:COUNt:INTermediate?`

Example Statements:
OUTPUT 719;":Aver:Count:Int?"
OUTPUT 719;"SENSE:AVER:COUN:INTERMEDIATE?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command returns the number of averages that have been completed. The number of averages that the analyzer performs is determined by [SENSE:]AVERage:COUNt.

[SENSe:]AVERage:IRESult:RATE command/query

Sets the value of the display update rate for fast averaging.

Command Syntax: `[SENSe:]AVERage:IRESult:RATE <number>|<step>|<bound>`

<number> ::= an integer (NRf data
limits: 1:99999

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":AVERAGE:IRES:RATE 100"
OUTPUT 719;"sense:aver:iresult:rate 300"

Query Syntax: `[SENSe:]AVERage:IRESult:RATE?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  10
SCPI Compliance:  instrument-specific
```

Description:

Fast averaging allows the numeric processing to run faster by reducing the time spent updating the display. The user selects a number that specifies how often the display is updated and turns fast averaging on. If 5 is entered, the display is updated once every 5 averages.

- Fast averaging is turned on and off with the AVER:IRES command.
- Averaging is turned on and off with the AVER command.

The larger the update rate, the faster the average gets to the specified count, N.

[SENSe:]AVERage:IRESult[:STATe] command/query

Turns the fast average on and off.

Command Syntax: `[SENSe:]AVERage:IRESult[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;"sens:average:ires:stat OFF"
OUTPUT 719;"Average:Ires ON"

Query Syntax: `[SENSe:]AVERage:IRESult[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Description:

Fast averaging allows the numeric processing to run faster by reducing the time spent updating the display. The display is updated once every X number of averages, where X is a user-selected value.

- The fast average update rate is set with AVER:IRES:RATE.
- Averaging is turned on with AVER ON.

[SENSe:]AVERage[:STATe] command/query

Turns averaging on and off.

Command Syntax: `[SENSe:]AVERage[:STATe] OFF|0|ON|1`

Example Statements:
```
OUTPUT 719;":AVERAGE OFF"
OUTPUT 719;"sens:aver:state ON"
```

Query Syntax: `[SENSe:]AVERage[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

When averaging is off, each trace represents the results of a single measurement. When averaging is on, each trace represents the combined results of several measurements.

- AVER:TYPE specifies how results are combined.
- AVER:TCON (terminal control) specifies how the averaging process terminates.
- AVER:COUN specifies the number of measurements to be averaged together (for normal averaging) or a weighting factor (for exponential averaging).
- AVER:IRES turns fast averaging on and off.
- AVER:IRES:RATE specifies the fast averaging update rate.

To determine when averaged measurement data is available, use *OPC (operation complete) as described in chapter 3 of the *GPIB Programmer's Guide* under "Synchronization."

- When averaging is off, the NPO flag (No Pending Operation) is set when the current measurement scan is complete.
- When averaging is on, the NPO flag is set when the number of averages requested has been completed. Even when the type is exponential and the number of averages is used as a weighting factor (the averaging process never stops), the NPO flag *does* eventually get set.

If the measurement is restarted with the *ABOR command, the NPO flag is not set until the requested number of averages is taken.

Note Trigger conditions must be met for each measurement; even when averaging is on.

[SENSe:]AVERage:TCONtrol **command/query**

Terminal Control specifies the action of the AVERage subsystem when AVERage:COUNt measurement results are generated.

Command Syntax: `[SENSe:]AVERage:TCONtrol EXPonential|NORMal|REPeat`

Example Statements:
OUTPUT 719;"Sens:Average:Tcon EXPONENTIAL"
OUTPUT 719;"AVER:TCONTROL REPEAT"

Query Syntax: `[SENSe:]AVERage:TCONtrol?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  EXP
SCPI Compliance:  confirmed
```

Description:

The parameters have the following meanings:

NORMal: Additional averages continue to be accumulated according to the average type selected.

REPeat: Clear average data and counter and restart the average process.

EXPonential: Continue the average with an exponential weighting applied to old values. For complex types, the additional averages are weighted as follows:

$$AVG_i = \frac{1}{n} X_i + \frac{n-1}{n} AVG_{i-1}$$

For RMS averaging the weighting is:

$$AVG_i = \sqrt{\frac{1}{n} |X_i|^2 + \frac{n-1}{n} |AVG_{i-1}|^2}$$

where n is the weighting factor specified with AVER:COUNt, AVG_i is the most recent average value, X_i is the most recent measurement data, and AVG_{i-1} is the previous average value

[SENSe:]AVERage:TYPE command/query

Selects the averaging type.

Command Syntax: `[SENSe:]AVERage:TYPE MAX|RMS|COMPlex`

Example Statements:
OUTPUT 719;":Aver:Type MAX"
OUTPUT 719;"SENS:AVERAGE:TYPE RMS"

Query Syntax: `[SENSe:]AVERage:TYPE?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  RMS
SCPI Compliance:  confirmed
```

Description:

Averaging is turned on with AVER ON. AVER:TYPE may be specified *before* averaging is turned on, however. The three averaging types are RMS (power), COMPlex (time), and MAX (peak hold, a display function).

RMS (root mean squared) is a power average which reduces the variance of each point's *magnitude* and smooths the noise floor. This averaging type does not lower the noise floor as complex averaging can.

COMPlex or *time* averaging is available only when the instrument mode is vector (INST VECT). Complex averaging reduces the variance of each point's *real and imaginary* parts which lower's and smoothes the noise floor.

MAX is the same as peak hold, which displays the maximum value of each displayed point over the course of several measurements (the measurement count is specified with AVER:COUN).

When you press softkeys in the front-panel menus to select average type, you also select the type of termination control used to calculate average information. The following table shows the command strings that correpsond to the average type softkeys. To display the SCPI command strings that correspond to front-panel keys, activate the SCPI command echo by pressing [**Local/Setup**], [SCPI cmd echo **on**] or sending "`SYST:GPIB:ECHO 1`". Subsequent SCPI command strings are displayed in the upper-lefthand corner of the display.

Front-Panel Average Types and their Corresponding GPIB Command Strings

Front-Panel Selection	Corresponding Command String
rms (video)	AVER:TYPE RMS;TCON NORM
rms (video) exponential	AVER:TYPE RMS;TCON EXP
time	AVER:TYPE COMP;TCON NORM
time exponential	AVER:TYPE COMP;TCON EXP
continuous peak hold	AVER:TYPE MAX

[SENSe:]BANDwidth:MODE:ARBitrary command/query

Selects the resolution bandwidth type.

Command Syntax: `[SENSe:]BANDwidth:MODE:ARBitrary OFF|0|ON|1`

Example Statements:
OUTPUT 719;":band:mode:arb ON"
OUTPUT 719;"Sens:Bandwidth:Mode:Arbitrary ON"

Query Syntax: `[SENSe:]BANDwidth:MODE:ARBitrary?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (off)
SCPI Compliance:  instrument-specific
```

Description:

When BAND:MODE:ARB is OFF, only values in the 1-3-10 sequence are valid entries. When it is ON, any value (within limits) is valid.

Related Commands:

- BAND specifies a resolution bandwidth value. It doesn't disable span tracking, so if span changes so does the RBW. It *does* change BAND:AUTO to ON.
- BAND:AUTO controls span tracking (on/off). It must be off if you want to specify an RBW that doesn't change.
- BAND:AUTO:OFFS specifies whether the RBW/span ratio tracked is user-determined.

[SENSe:]BANDwidth[:RESolution] command/query

Specifies the resolution bandwidth.

Command Syntax: `[SENSe:]BANDwidth[:RESolution] <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (depends on configuration)
<unit>	::=	HZ
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"SENS:BANDWIDTH:RES 300 KHZ"
OUTPUT 719;"bandwidth 100 khz"
```

Query Syntax: `[SENSe:]BANDwidth[:RESolution]?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  89410: 100 kHz, 89441: 300 kHz
SCPI Compliance:  confirmed
```

Description:

The resolution bandwidth value is limited by a number of factors. In general, the minimum bandwidth is limited by the maximum time record and the maximum bandwidth is limited by the minimum time record. The time record length is a function of the amount of memory installed and allocated to measurement memory (as opposed to memory used for IBASIC programming or used as a RAM disk).

[SENSe:]BANDwidth[:RESolution]:AUTO command/query

Specifies whether the resolution bandwidth tracks span.

Command Syntax: `[SENSe:]BANDwidth[:RESolution]:AUTO OFF|0|ON|1`

Example Statements:
OUTPUT 719;"SENS:BAND:RESOLUTION:AUTO ON"
OUTPUT 719;"bandwidth:auto OFF"

Query Syntax: `[SENSe:]BANDwidth[:RESolution]:AUTO?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1 (ON)
SCPI Compliance:  confirmed
```

Description:

At preset and power-on, resolution bandwidth (RBW) tracks span changes to maintain an internally-defined RBW/span ratio. (The resulting RBW also depends on the state of BAND:MODE:ARB.)

Related Commands:

- BAND specifies a resolution bandwidth value. It doesn’t disable span tracking, so if span changes so does the RBW. It *does* change BAND:AUTO to ON.
- BAND:AUTO controls span tracking (on/off). It must be off if you want to specify an RBW that doesn’t change.
- BAND:MODE:ARB controls whether the rbw mode is arb (on) or 1-3-10 (off).
- BAND:AUTO:OFFS specifies whether the RBW/span ratio tracked is user-determined.

[SENSe:]BANDwidth[:RESolution]:AUTO:OFFSet — command/query

Specifies whether the RBW/span ratio tracked when BAND:AUTO is ON is user-determined.

Command Syntax: `[SENSe:]BANDwidth[:RESolution]:AUTO:OFFSet OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Band:Auto:Offs ON"
OUTPUT 719;"SENSE:BAND:RES:AUTO:OFFS OFF"

Query Syntax: `[SENSe:]BANDwidth[:RESolution]:AUTO:OFFSet?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (off)
SCPI Compliance:  instrument-specific
```

Description:

With BAND:AUTO ON and BAND:AUTO:OFFS OFF, the RBW tracks span changes to maintain an internally-defined RBW/span ratio. (The resulting RBW also depends on the state of BAND:MODE:ARB.)

If BAND:AUTO is ON, BAND:AUTO:OFFS is automatically turned on when you enter an RBW value (with BAND). Subsequent changes in span will cause the RBW to track it such that it maintains the new RBW/span ratio.

The BAND:AUTO:OFFS command has no effect if BAND:AUTO is OFF.

[SENSe:]CORRection[1 | 2]:EDELay[:TIME] command/query

Adjusts the delay, in small increments, for the specified input channel.

Command Syntax: `[SENSe:]CORRection[1|2]:EDELay[:TIME] <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	-100 nanoseconds to +100 nanoseconds
<unit>	::=	S
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"sense:corr:edel:time 0"
OUTPUT 719;"Corr2:Edelay 0"
```

Query Syntax: `[SENSe:]CORRection[1|2]:EDELay[:TIME]?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

By default, the delay for an input channel is zero. Use this command to adjust an input channel's delay to compensate for external devices (such as cables or attenuators). CORR1 selects input channel 1; CORR2 selects input channel 2. The delay that you specify is not used until you send sense:correction[1|2]:external:state ON.

For additional details, see online help for the [**System Utility**] [more cal setup] [external calibration] softkey.

[SENSe:]CORRection[1 | 2]:EXTernal[:STATe] command/query

Enables or disables the use of user-defined calibration coefficients.

Command Syntax: `[SENSe:]CORRection[1|2]:EXTernal[:STATe] OFF|0|ON|1`

Example Statements:
OUTPUT 719;":CORR:EXT ON"
OUTPUT 719;"sense:corr:external:stat OFF"

Query Syntax: `[SENSe:]CORRection[1|2]:EXTernal[:STATe]?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Description:

You can adjust the gain and delay calibration coefficients for each input channel. By default, the gain of an input channel is one, the delay is zero seconds. You can adjust these values using the sense:correction:loss:magnitude and sense:correction:edelay commands.

The values that you specify are not used until you send sense:correction:external ON. To return to the default gain and delay values, send sense:correction:external OFF.

For additional details, see online help for the [**System Utility**] [more cal setup] [external calibration] softkey.

[SENSe:]CORRection[1|2]:FILTer:XTIMe:STATe command/query

Enables or disables time-domain corrections.

Command Syntax: `[SENSe:]CORRection[1|2]:FILTer:XTIMe:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Sens:Correction2:Filt:Xtime:Stat ON"
OUTPUT 719;"CORR2:FILTER:XTIM:STATE OFF"

Query Syntax: `[SENSe:]CORRection[1|2]:FILTer:XTIMe:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  Off
SCPI Compliance:  instrument-specific
```

Description:

Time calibrations compensate the time data for the magnitude and phase characteristics of each input channel. For the greatest accuracy, time calibrations should be used whenever demod is enabled, or accurate time data is desired. It should be turned off whenever measurement speed is a high priority.

Accuracy of frequency domain results is not affected by this command.

[SENSe:]CORRection[1 | 2]:IMPedance[:INPut][:MAGNitude] command/query

Sets the dBm reference impedance when the 1 MΩ input termination is selected.

Command Syntax: `[SENSe:]CORRection[1|2]:IMPedance <number>[<unit>]`

<number> ::= a real number (NRf data)
limits: .001:10E9

<unit> ::= OHM

Example Statements:

OUTPUT 719;":Corr1:Impedance 600"
OUTPUT 719;"SENS:CORRECTION2:IMP 150"

Query Syntax: `[SENSe:]CORRection[1|2]:IMPedance[:INPut][:MAGNitude]?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  50
SCPI Compliance:  instrument-specific
```

Description:

This command changes the reference impedance used to calculate dBm when the selected impedance is 1 MΩ (INP:IMP 1e6). Each input channel may be given a value independent of the other (assuming channel 2, which is optional, is installed).

Related Commands:

- The input impedance is set to 1 MΩ with the command: “INP[1|2]:IMP 1e6"

Note The input signal is terminated with 1 MΩ when this command is effective.

[SENSe:]CORRection[1|2]:LOSS[:INPut]:MAGNitude command/query

Adjusts the gain for the specified input channel.

Command Syntax: `[SENSe:]CORRection[1|2]:LOSS[:INPut]:MAGNitude <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	0.000001 to 1000000
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements: OUTPUT 719;"SENSE:CORR:LOSS:INPUT:MAGN .001"

Query Syntax: `[SENSe:]CORRection[1|2]:LOSS[:INPut]:MAGNitude?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

By default, the gain for an input channel is one. Use this command to adjust an input channel's gain to compensate for external devices (such as cables or attenuators). CORR1 selects input channel 1; CORR2 selects input channel 2. The gain that you specify is not used until you send sense:correction:external:state ON.

For additional details, see online help for the [**System Utility**] [more cal setup] [external calibration] softkey.

[SENSe:]CORRection[1 | 2]:OFFS **command/query**

Adjusts the DC offset, in small increments, for the specified input channel.

Command Syntax: `[SENSe:]CORRection[1|2]:OFFS <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	-20 volts to +20 volts
<unit>	::=	V
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":corr:offs 0"
OUTPUT 719;"Sense:Corr:Offs 0"
```

Query Syntax: `[SENSe:]CORRection[1|2]:OFFS?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

By default, the offset for an input channel is zero. Use this command to adjust an input channel's DC offset to compensate for external devices. The offset is compensated for mathematically. With DC coupling, the input range must still be adjusted to handle both the AC and DC components of the signal. The offset that you specify is only applied in the IF receiver modes and only after you send sense:correction[1|2]:external:state ON. Also, for a receiver mode of IF(0-10MHz), the time data must be baseband.

For additional details, see online help for the [**System Utility**] [more cal setup] [external calibration] softkey.

[SENSe:]DATA command/query

Uploads or downloads time-capture data between the analyzer and the controller.

Command Syntax: `[SENSe:]DATA TCAP1|TCAP2,<DATA>`

```
<DATA>  ::=  <DEF_BLOCK>
        ::=  <NRf>,<NRf>,<NRf>, . . .
```

Example Statements:

```
OUTPUT 719;":data tcap1, USER2"
OUTPUT 719;"Sense:Data Tcap2, USER2"
```

Query Syntax: `[SENSe:]DATA?`

Return Format: `DEF_USER`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

In the example statements, USER2 is a definite-length data block as defined in Chapter 4 of the *GPIB Programmer's User Guide*. The TCAP1 parameter designates the channel-1, time-capture buffer and TCAP2 designates the channel-2, time-capture buffer.

The analyzer can't load indefinite-length blocks; the data must be definite-length blocks or ASCII data. The data is stored internally in 32-bit integers but the data transferred with this command is floating-point. The floating-point numbers are scaled and converted to integers with the time-capture range value specified with the [SENSE:]TCAP[1|2]:RANG values. Using the time-capture range to scale the data yields the best dynamic range.

The query form of this command requires the use of a parameter: TCAP1|TCAP2 as in

DATA? TCAP1

Related Commands:

- To query the number of measurement points in the time-capture buffer, use [SENSe:]DATA:HEAD:POINts?.

Time capture data may also be transferred between the analyzer and a controller with the LAN file-transfer utility, FTP (option UG7).

[SENSe:]DATA:HEADer:POINts? query

Returns the number of points for the time-capture buffer specified.

Query Syntax: `[SENSe:]DATA:HEADer:POINts? TCAP1|TCAP2`

Example Statements:
OUTPUT 719;"SENS:DATA:HEADER:POIN? TCAP2"
OUTPUT 719;"data:head:poin? TCAP1"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query is used to determine the number of data points in the specified time-capture buffer. This information may be required to transfer the contents of the buffer from the analyzer to the controller.

Related Commands:

- To transfer time-capture data, use [SENSe:]DATA.

[SENSe:]DDEMod:ADAPt command/query

Enables adaptive equalization on/off.

Command Syntax: `[SENSe:]DDEMod:ADAPt OFF|0|ON|1`

Example Statements:
OUTPUT 719;"sense:ddem:adapt ON"
OUTPUT 719;"Ddem:Adap OFF"

Query Syntax: `[SENSe:]DDEMod:ADAPt?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

ON The equaliztion filter coefficients are updated with each measurement

OFF The equalization filter coefficients do not change

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:CLOCk command/query

Specifies the clock offset in symbols.

Command Syntax: `[SENSE:]DDEMod:CLOCk <number>[unit]|<step>|<bound>`

<number>	::=	a real number (NRf data) limits: depends on configuration
<unit>	::=	SYM
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":DDEMOD:CLOC 0"
OUTPUT 719;"sense:ddem:cloc 0"
```

Query Syntax: `[SENSe:]DDEMod:CLOCk?`

Return Format: `Real`

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This commands corresponds to *clock adjust* as discussed in the online Help.

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:CNVRg command/query

Specifies the convergence rate.

Command Syntax: `[SENSe:]DDEMod:CNVRg <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:1

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements: OUTPUT 719;"Sense:Ddem:Cnvrg 0"
OUTPUT 719;"DDEM:CNVR 0"

Query Syntax: `[SENSe:]DDEMod:CNVRg?`

Return Format: `Real`

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This commands specifies the rate at which the LMS equalizer converges. Larger values converge faster, but may be unstable and result in higher residual errors. The value can be changed while the equalizer is adapting.

[SENSe:]DDEMod:CRATe **command/query**

Sets the chip rate for Wideband CDMA measurements.

Command Syntax: `[SENSe:]DDEMod:CRATe {<number>[<unit>]}`

<number> ::= a real number (NRf data)
limits: 4086000:8212000 (16424000 for the ch1 + j*ch2 receiver)

<unit> ::= [HZ]

Example Statements: OUTPUT 719;":ddemod:crat 4096000"
OUTPUT 719;"Sense:Ddem:Crat 4096000"

Query Syntax: `[SENSe:]DDEMod:CRATe?`

Return Format: `Real`

Attribute Summary:

```
Option:  B73 (Digital Wideband CDMA analysis) or
B79 (Digital ARIB 1.0-1.2 Wideband CDMA analysis or
080 (Digital 3GPP Wideband CDMA analysis)

Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

This command sets the chip rate for W-CDMA measurements. The default chip rate is 4.096 MHz (Opt. B73, B79) or 3.84 MHz (Opt. 080). Sending this command as a query returns the current chip rate, in Hz.

The maximum chip rate that you can select is limited by the maximum W-CDMA span parameter (MEM:MALL:MEAS:MAXS).

[SENSe:]DDEMod:DVBQam:NSTate command/query

Specifies the number of states in the DVBQAM modulation format.

Command Syntax: `[SENSe:]DDEMod:DVBQam:NSTate <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 16:64

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"SENSE:DDEM:DVBQ:NST 16"
OUTPUT 719;"ddem:dvbq:nst 32"

Query Syntax: `[SENSe:]DDEMod:DVBQam:NSTate?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYA and AYH
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

Use this command to specify the number of states for the DVBQAM video modulation format. Valid numbers are 16, 32, or 64. For example, sense:ddem:dvbq:nst16 selects 16 DVBQAM.

To select the DVBQAM video modulation format, use DDEM:FORMAT DVBQAM. For additional information, see the “Option AYH” chapter in the *Operator’s Guide*.

[SENSe:]DDEMod:EDGE command/query

Selects the EDGE (Enhanced Data rates for GSM Evolution) demodulation format.

Command Syntax: `[SENSe:]DDEMod:EDGE NOFL`

Example Statements:
OUTPUT 719;"SENSE:DDEM:EDGE NOFL"
OUTPUT 719;"ddem:edge nofl"

Query Syntax: `[SENSe:]DDEMod:EDGE?`

Return Format: `Real`

Attribute Summary:
```
Options: AYA (Vector modulation Analysis and Adaptive Equaliza-
tion) and B7A (Enhanced Data rates for GSM Evolution (EDGE))
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This commands selects the EDGE demodulation format. You must set the instrument mode to digital demodulation (send INST DDEM) before selecting the EDGE demodulation format.

The only EDGE demodulation format that you can select is NOFL.

Tip

Selecting the EDGE demodulation format does not automatically set the appropriate filters or symbol rate. Make sure you set the IQ-measured filter to EDGE(winRC) (DDEM:FILT:MEAS EDGM), the IQ-reference filter to EDGE (DDEM:FILT:REF EDGE) and the symbol rate to 270.833kHz (DDEM:SRAT 270833.33 HZ).
To quickly set these and other EDGE demodulation parameters, send DDEM:PRES EDGE.

[SENSe:]DDEMod:EQFLen — command/query

Changes the length of the analyzer's adaptive equalization filter.

Command Syntax: `[SENSe:]DDEMod:EQFLen {<number>[<unit>]}|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 3:99 (odd values only)

<unit> ::= [SYM]

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

OUTPUT 719;"SENSE:DDEM:EQFLEN 0"
OUTPUT 719;"ddem:eqfl 0"

Query Syntax: `[SENSe:]DDEMod:EQFLen?`

Return Format: `Real`

Attribute Summary:

```
Option: AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This commands changes the overall length of the adaptive equalization filter. The length of the filter is entered in symbols. The length of the filter in taps is the number of symbols multiplied by the number of points per symbol.

[SENSe:]DDEMod:EQRE command

Resets the filter coefficients for the analyzer's equalization filter.

Command Syntax: `[SENSe:]DDEMod:EQRE`

Example Statements:
OUTPUT 719;":Ddemod:Eqre"
OUTPUT 719;"SENSE:DDEM:EQRE"

Attribute Summary:
```
Option: AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

This command resets the filter coefficients of the equalization filter to that of a unit impulse response.

[SENSe:]DDEMod:EQUalize **command/query**

Turns equalization on/off for Digital and Video demodulation.

Command Syntax: `[SENSe:]DDEMod:EQUalize OFF|0|ON|1`

Example Statements:
OUTPUT 719;"sense:ddem:equalize ON"
OUTPUT 719;"Ddem:Equ OFF"

Query Syntax: `[SENSe:]DDEMod:EQUalize?`

Return Format: `Integer`

Attribute Summary:
```
Option: AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

ON Applies the equaliztion filter to the IQ Measured data

OFF The equalization filter is available, but not used.

For more information on using equalization in digital and video demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:FILTer:ALPHa command/query

Specifies the alpha of the Nyquist filters and BT for Gaussian filters.

Command Syntax: `[SENSe:]DDEMod:FILTer:ALPHa <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0.05 to 100

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":DDEMOD:FILT:ALPHA 0"
OUTPUT 719;"sens:ddem:filter:alph 0"

Query Syntax: `[SENSe:]DDEMod:FILTer:ALPHa?`

Return Format: `Real`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command corresponds to the *alpha / BT* softkey. It sets the value of the α and bandwidth-time variables for both the measurement filter and the reference filter when the instrument mode is Digital or Video demodulation.

To set the filter alpha for the Wideband CDMA instrument mode, use [SENSe:]DDEMod:WCDMa:FILTer:ALPHa.

For more information on Digital and Video demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:FILTer:MEASurement command/query

Specifies the type of filter used in digital or video demodulation measurements.

Command Syntax: `[SENSe:]DDEMod:FILTer:MEASurement <param>`

<param> ::= OFF|0|RECTangular|RCOSine|RRCosine|GAUSsian|PHEQualize|USER|EDGM|LPASs

Example Statements:

OUTPUT 719;"Sense:Ddem:Filt:Measurement USER"
OUTPUT 719;"DDEM:FILTER:MEAS RRCOSINE"

Query Syntax: `[SENSe:]DDEMod:FILTer:MEASurement?`

Return Format: CHAR

Attribute Summary:

```
Option:  AYA (vector modulation analysis)or
B7A (Enhanced Data rates for GSM Evolution (EDGE))
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

OFF|0 specifies no filtering.

RECT specifies the use of a rect filter.

RCOS specifies the use of a *raised-cosine* filter. The value of the alpha variable is specified with the DDEM:FILT:ALPH command.

RRC specifies the use of a *root raised-cosine* filter. The value of the alpha variable is specified with the DDEM:FILT:ALPH command.

GAUS specifies the use of a *Gaussian* filter. The value of the BT variable is specified with the DDEM:FILT:ALPH command.

LPAS specifies the use of a low-pass filter. This is a Gaussian filter with BT = 0.5 and T = 1/(4.0 * symbol_rate).

USER specifies the use of a user-defined filter. These are defined solely by data in the data registers (alpha and BT have no effect). Data is loaded into the data registers with the MMEM:LOAD:TRAC command. The data register used for the filter is specified with the DDEM:FILT:MEAS:USER:FEED command.

EDGM specifies the use of the windowed raised-cosine filter. This is an EDGE demodulation specific measurement filter, not affected by alpha.

The *reference* filter type is specified with the DDEM:FILT:REF command.

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:FILTer:MEASurement:USER:FEED command/query

Specifies which data register is used to define the user-defined digital demod measurement filter.

Command Syntax: `[SENSe:]DDEMod:FILTer:MEASurement:USER:FEED D1|D2|D3|D4|D5|D6`

Example Statements:
OUTPUT 719;"Ddem:Filter:Meas:User:Feed D2"
OUTPUT 719;"SENS:DDEMOD:FILT:MEAS:USER:FEED D5"

Query Syntax: `[SENSe:]DDEMod:FILTer:MEASurement:USER:FEED?`

Return Format: `STRING`

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Before the data register is selected, the user-defined data should be loaded into it and the user-defined filter type selected.

Related Commands:

- To specify that the measurement filter is user-defined, use DDEM:FILT:MEAS USER.
- To load the filter definition data into a data register, use MMEM:LOAD:TRAC.

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:FILTer:REFerence command/query

Specifies the type of filter used as the digital demod reference.

Command Syntax: `[SENSe:]DDEMod:FILTer:REFerence <param>     <param>   ::= RECTangular|RCOSine|RRCosine|CHEByshev|GAUSsian|EDGE|USER`

Example Statements:
OUTPUT 719;":ddem:filter:ref RECT"
OUTPUT 719;"Sense:Ddem:Filt:Reference RRCOSINE"

Query Syntax: `[SENSe:]DDEMod:FILTer:REFerence?`

Return Format: `CHAR`

Attribute Summary:

```
Option:  AYA (vector modulation analysis)or
         B7A (Enhanced Data rates for GSM Evolution (EDGE))

Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

RECT specifies the use of a rectangular filter (the impulse response is the rect function).

RCOS specifies the use of a *raised-cosine* filter. The value of the alpha variable is specified with the DDEM:FILT:ALPH command.

RRC specifies the use of a *root raised-cosine* filter. The value of the alpha variable is specified with the DDEM:FILT:ALPH command.

GAUS specifies the use of a **Gaussian** filter. The value of the BT variable is specified with the DDEM:FILT:ALPH command.

EDGE specifies the use of an EDGE filter. When the measurement filter is set to OFF, it is the EDGE TX (transmit) filter. When the measurement filter is set to EDGE (winRC), it is the convolution of the EDGE TX and measurement filters. This selection requires Option B7A.

USER specifies the use of a user-defined filter. These are defined solely by data in the data registers (alpha and BT have no effect). Data is loaded into the data registers with the MMEM:LOAD:TRAC command. The data register used for the filter is specified with the DDEM:FILT:REF:USER:FEED command.

The *measurement* filter type is specified with the DDEM:FILT:MEAS command.

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:FILTer:REFerence:USER:FEED command/query

Specifies which data register is used as the user-defined digital demod reference filter.

Command Syntax: `[SENSe:]DDEMod:FILTer:REFerence:USER:FEED D1|D2|D3|D4|D5|D6`

Example Statements:
OUTPUT 719;"SENSE:DDEM:FILTER:REF:USER:FEED ‘D2’"
OUTPUT 719;"ddem:filter:ref:user:feed ‘D5’"

Query Syntax: `[SENSe:]DDEMod:FILTer:REFerence:USER:FEED?`

Return Format: `STRING`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Before the data register is selected, the user-defined data should be loaded into it and the user-defined filter type selected.

Related Commands:

- To specify that the reference filter is user-defined, use DDEM:FILT:REF USER.
- To load the filter definition data into a data register, use MMEM:LOAD:TRAC.

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator’s Guide*.

[SENSe:]DDEMod:FORMat — command/query

Specifies the demodulation format for digital modulation or video modulation.

Command Syntax: `[SENSe:]DDEMod:FORMat QPSK|PSK|QAM|MSK|FSK|DVBQam|VSB|EDGE`

Example Statements:

```
OUTPUT 719;"SENS:DDEMOD:FORM PSK"
OUTPUT 719;"ddem:format QAM"
```

Query Syntax: `[SENSe:]DDEMod:FORMat?`

Return Format: `CHAR`

Attribute Summary:

```
Option:  AYA and, for some, AYH or B7A (see below)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command corresponds to the softkey selections under the *demod format* menu. You must set the instrument mode to digital demodulation (send INST DDEM) to select QPSK, PSK, QAM, MSK, FSK, and EDGE. You must set the instrument mode to video demodulation (send INST VDEM) to select DVBQ, VSB or 16|32|64|256 QAM.

QPSK selects quadrature phase-shift keying (requires option AYA).
PSK selects phase-shift keying (requires option AYA).
QAM selects quadrature amplitude modulation (requires option AYA).
MSK selects minimum-shift keying (requires option AYA).
FSK selects frequency-shift keying (requires option AYA).
DVBQ selects digital video broadcast QAM (requires options AYA and AYH)
VSB selects vestidual side-band (requires options AYA and AYH).
EDGE selects Enhanced Data rates for GSM Evolution (requires option B7A).

Command Combinations:

- To specify normal or pi/4 DQPSK, use DDEM:QPSK:FORM DIFF or DPI4.
- To specify BPSK or 8 PSK, use DDEM:FORM PSK;PSK:NST 2 | 8.
- To specify different Digital Demod 16|32 QAM formats, use DDEM:QAM:NST 16 | 32.
- To specify MSK type 1 or type 2, use DDEM:MSK:FORM TYP1 or TYP2
- To specify FSK 2 or FSK 4, use DDEM:FSK:NST 2 | 4.
- To specify 8 | 16 VSB, use DDEM:VSG:NST 8 | 16.
- To specify DVB QAM 16 | 32 | 64, use DDEM:DVBQ:NST 16 | 32 | 64.
- To specify different Video Demod 16|32|64|256 QAM formats, use DDEM:QAM:NST 16|32|64|256.
- You can also specify the EDGE format with DDEM:EDGE:FORM NOFL.

For more information on digital demodulation and on video demodulation, see the *Operator's Guide*. See also DDEM:QPSK:FORM, DDEM:PSK:NSTate, DDEM:QAM:NSTate, DDEM:FSK:NSTate, DDEM:DVBQ:NSTate, DDEM:EDGE:FORM, and DDEM:VSB:NSTate.

[SENSe:]DDEMod:FREQuency:MIRRor command/query

Turns frequency mirroring on or off.

Command Syntax: `[SENSe:]DDEMod:FREQuency:MIRRor OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Ddem:Frequency:Mirr OFF"
OUTPUT 719;"SENS:DDEMOD:FREQ:MIRROR OFF"

Query Syntax: `[SENSe:]DDEMod:FREQuency:MIRRor?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYH (video modulation analysis)
         B73 (Wideband CDMA Analysis)
Synchronization Required:  no
Preset State:  +0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

Sending ddemod:freq:mirror ON turns mirroring on. When mirroring is on, the analyzer “flips” or “mirrors” the displayed spectrum around the analyzer’s center frequency.

[SENSe:]DDEMod:FSK:NSTate **command/query**

Specifies the number of states in the FSK modulation format.

Command Syntax: `[SENSe:]DDEMod:FSK:NSTate <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 2:4

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements: OUTPUT 719;"sens:ddem:fsk:nst 3"
OUTPUT 719;"Ddemod:Fsk:Nst 3"

Query Syntax: `[SENSe:]DDEMod:FSK:NSTate?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

The valid values for this command are 2 or 4.

Related Commands:

- To select the FSK modulation format, use DDEM:FORMAT FSK.

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:LCODe command/query

Sets the long code for Wideband CDMA measurements.

Command Syntax: `[SENSe:]DDEMod:LCODe <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 1:512

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

OUTPUT 719;":DDEMOD:LCOD 1"
OUTPUT 719;"sense:ddem:lcod 3"

Query Syntax: `[SENSe:]DDEMod:LCODe?`

Return Format: `Integer`

Attribute Summary:

```
Option:  B73 (Digital Wideband CDMA analysis (W-CDMA))or
         B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
         080 (Digital 3GPP W-CDMA analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

When DDEM:LCOD:MODE CODE is selected, this command sets the long code that the analyzer uses to demodulate W-CDMA signals.

The range of valid long-codes depend on the W-CDMA format (selected with the [SENSe]:DDEM:SEL command). The following table shows the W-CDMA formats, associated valid long-code values and required analyzer option:

W-CDMA Format	DDEM:SEL Commad	Valid Long Codes	Required Option
trial 1998	TFOR	1 to 128	B73
ARIB 1.0-1.2	A1FOR	1 to 512	B79
3GPP	G3FOR	0 to 128	080

[SENSe:]DDEMod:LCODe:GROup command/query

Sets the long code group for Wideband CDMA measurements.

Command Syntax: `[SENSe:]DDEMod:LCODe:GROup <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 1:32

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"Sense:Ddem:Lcode:Gro 1"
OUTPUT 719;"DDEM:LCODE:GRO 1"

Query Syntax: `[SENSe:]DDEMod:LCODe:GROup?`

Return Format: `Integer`

Attribute Summary:
```
Option:  B73 (Digital Wideband CDMA analysis (W-CDMA))or
         B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
         080 (Digital 3GPP W-CDMA analysis)

Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

When DDEM:LCOD:MODE GROUP is selected, this command sets the long code group that the analyzer searches through to determine which long code to use to demodulate W-CDMA signals.

The range of valid long-code groups depend on the W-CDMA format (selected with the [SENSe]:DDEM:SEL command). The following table shows the W-CDMA formats, associated valid long-code group values and required analyzer option:

W-CDMA Format	DDEM:SEL Commad	Valid Long Code Groups	Required Option
trial 1998	TFOR	2 to 5	B73
ARIB 1.0-1.2	A1FOR	1 to 32	B79
3GPP	G3FOR	0 to 63	080

[SENSe:]DDEMod:LCODe:MODE **command/query**

Determines how the analyzer detects the long code for W-CDMA measurements.

Command Syntax: `[SENSe:]DDEMod:LCODe:MODE CODE|GROup`

Example Statements:
OUTPUT 719;":ddemod:lcod:mode GROUP"
OUTPUT 719;"Sense:Ddem:Lcode:Mode CODE"

Query Syntax: `[SENSe:]DDEMod:LCODe:MODE?`

Return Format: `CHAR`

Attribute Summary:

```
Option:  B73 (Digital Wideband CDMA analysis (W-CDMA))or
         B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
         080 (Digital 3GPP W-CDMA analysis)

Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command determines how the analyzer detects the long code for W-CDMA measurements.

Use CODE if you know the long code, in which case you must also send DDEM:LCODEe to specify the long code.

Use GROUP if you don’t know the long code, but you do know the long-code group used to generate the W-CDMA signal. In this case, you must also send DDEM:LCODe:GROup to specify the long-code group.

CODE provides the fastest measurement speed.

[SENSe:]DDEMod:MLENgth command/query

Sets the main length, in slots, for W-CDMA measurements.

Command Syntax: `[SENSe:]DDEMod:MLENGTH <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number	::=	a real number (NRf data)
limits	::=	8 to 20
<unit>	::=	[S|SLOT]
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;":ddem:mlen 8"
OUTPUT 719;"Sens:Ddem:Mlength 10"

Query Syntax: `[SENSe:]DDEMod:MLENgth?`

Return Format: `Real`

Attribute Summary:

```
Option:  B73 (Digital Wideband CDMA analysis (W-CDMA))or
B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
080 (Digital 3GPP W-CDMA analysis)

Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command determines the number of slots that the analyzer displays for W-CDMA measurements. You must enter an integer number of slots. If you use units of seconds, the analyzer rounds your entry up as necessary to obtain an integer number of slots.

[SENSe:]DDEMod:MSK:FORMat command/query

Specifies the form of MSK (minimum-shift key) modulation format.

Command Syntax: `[SENSe:]DDEMod:MSK:FORMat TYPe1|TYPe2`

Example Statements:
OUTPUT 719;":Ddemod:Msk:Format TYPE1"
OUTPUT 719;"SENS:DDEM:MSK:FORM TYPE1"

Query Syntax: `[SENSe:]DDEMod:MSK:FORMat?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:NORMalize command/query

Turns digital demod normalization on/off.

Command Syntax: `[SENSe:]DDEMod:NORMalize OFF|0|ON|1`

Example Statements:
OUTPUT 719;"sense:ddem:norm OFF"
OUTPUT 719;"Ddemod:Norm OFF"

Query Syntax: `[SENSe:]DDEMod:NORMalize?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYA (vector modulation analysis) and
         B7A (Enhanced Data rates for GSM gsm Evolution (EDGE)) or
         B73 (Digital Wideband CDMA analysis (W-CDMA)) or
         B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
         080 (Digital 3GPP W-CDMA analysis)

Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

ON performs a minimum square error calculation of the all constellation states and sets the system gain such that the outer states have a magnitude of 1. For W-CDMA code-domain power displays, ON normalizes code-domain power relative to the total signal power in the code domain.

OFF means you can interpret the measurement data in volts.

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:PRATe command/query

Specifies the number of points per symbol for vector modulation analysis.

Command Syntax: `[SENSe:]DDEMod:PRATe <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: (depends on configuration)

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":DDEMOD:PRAT 2"
OUTPUT 719;"sens:ddemod:prat 10"

Query Syntax: `[SENSe:]DDEMod:PRATe?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

The acceptable argument values for this command are 1, 2, 3, 4, 10, and 20. The maximum number of points between symbols is 20 but the analyzer decreases this value if the `result-length` × `PRATE` is greater than the maximum number of time points.

For more information on digital demodulation, see online help and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:PRESet **command**

Presets the digital demodulation configuration to be one of the standard configurations listed below.

Command Syntax: `[SENSe:]DDEMod:PRESet <standard>`

<standard> ::=NADC|PDC|GSM|PHP|DECT|CDPD|TETRa|CDMB|CDMM |ATV8|ATV16|DVB16|DVB32|DVB64|APCO|EDGE|WCDM|WCD5|WCD10|WCD20|BLU

Example Statements:

```
OUTPUT 719;":DDEMOD:PRES ATV16, 2.66409e+06"
OUTPUT 719;"sens:ddemod:pres TETRA, 8.70895e+06"
```

Attribute Summary:

```
Option:  AYA (vector modulation analysis) and
         B7A (Enhanced Data rates for GSM gsm Evolution (EDGE)) or
         B73 (Digital Wideband CDMA analysis (W-CDMA)) or
         B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
         080 (Digital 3GPP W-CDMA analysis)

Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command corresponds to the softkey selections under the *standard setups* menu. Note that CDMB selects the CDMA Base standard setup; CDMM selects the CDMA Mobile standard setup.

You must set the instrument mode to video demodulation (send INST:VDEM) to select ATV8, ATV16, DVB16, DVB32, and DVB64 (Option AYA).

You must set the instrument mode to Wideband CDMA (send INST:WCDMa) to select WCD5, WCD10, or WCD20 (Option B73, B79). WCD5 is the only preset for Option 080 (Digital 3GPP W-CDMA analysis).

Note WCDM is available in both the Wideband CDMA and Digital Demodulation instrument modes. For the Wideband CDMA instrument mode, WCDM is identical to WCD5.

You must set the instrument mode to digital demodulation (send INST DDEM) to select all other standard setups.

For more information on digital or video demodulation or on Wideband CDMA, see online help (press the [**Help**] key on the front panel, then press the softkey of interest). Or see the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:PSK:NSTate command/query

Specifies the number of states in the PSK modulation format.

Command Syntax: `[SENSe:]DDEMod:PSK:NSTate <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 2:8

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;":ddem:psk:nstate 2"
OUTPUT 719;"Sens:Ddemod:Psk:Nst 8"
```

Query Syntax: `[SENSe:]DDEMod:PSK:NSTate?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

Valid values for this command are 2 or 8.

Related Commands:

- To select the PSK modulation format, use DDEM:FORMAT PSK.

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:QAM:NSTate — command/query

Specifies the number of states in the QAM modulation format.

Command Syntax: `[SENSe:]DDEMod:QAM:NSTate <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 16:256

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

OUTPUT 719;"SENSE:DDEM:QAM:NST 16"
OUTPUT 719;"ddem:qam:nst 32"

Query Syntax: `[SENSe:]DDEMod:QAM:NSTate?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYA and, for 64 or 256 QAM, option AYH (see below)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

Use this command to select the number of states for the QAM modulation format. Valid values are 16 or 32. If you have option AYH, you can also specify 64 or 256.

To select the QAM modulation format, use DDEM:FORMAT QAM.

For additional information on digital demodulation or video demodulation, see the *Operator's Guide*.

[SENSe:]DDEMod:QPSK:FORMat command/query

Specifies the form of QPSK modulation format.

Command Syntax: `[SENSe:]DDEMod:QPSK:FORMat NORMal|OFFSet|DIFFerential|DPI4`

Example Statements:
OUTPUT 719;"Sense:Ddem:Qpsk:Format NORMAL"
OUTPUT 719;"DDEM:QPSK:FORM OFFSET"

Query Syntax: `[SENSe:]DDEMod:QPSK:FORMat?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:SEARch:PULSe:STATe **command/query**

Turns pulse search on or off.

Command Syntax: `[SENSe:]DDEMod:SEARch:PULSe:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;":ddem:search:puls:state OFF"
OUTPUT 719;"Sens:Ddem:Search:Puls:State ON"

Query Syntax: `[SENSe:]DDEMod:SEARch:PULSe:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This corresponds to *pulse search* as discussed in online help. A pulse is defined as an off-on-off carrier transition.

For more information on digital demodulation, see online help and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:SEARch:SYNC:OFFSet — command/query

Specifies the offset for sync search in digital demodulation.

Command Syntax:

```
[SENSe:]DDEMod:SEARch:SYNC:OFFSet <param>
```

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: depends on configuration
<unit>	::=	SYM
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":DDEM:SEAR:SYNC:OFFS 14"
OUTPUT 719;"sense:ddem:sear:sync:offs 60"
```

Query Syntax:

```
[SENSe:]DDEMod:SEARch:SYNC:OFFSet?
```

Return Format:

```
Integer
```

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command takes integer numbers of symbols. If a real number is used, the fractional part of the number is ignored. If no units are specified, SYM is assumed.

For more information on digital demodulation, see online help and the concepts discussion in the *Operator's Guide*.

[SENSe:]DDEMod:SEARch:SYNC:PATTern command/query

Specifies the sync pattern for digital demodulation.

Command Syntax: `[SENSe:]DDEMod:SEARch:SYNC:PATTern <STRING>`

Example Statements:
OUTPUT 719;"SENS:DDEM:SEAR:SYNC:PATT ‘1010100100011101111001001010’"
OUTPUT 719;"ddemod:sear:sync:patt ‘11001010110100011’"

Query Syntax: `[SENSe:]DDEMod:SEARch:SYNC:PATTern?`

Return Format: `STRING`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

The analyzer accepts only the characters one (1) or zero (0) in these strings. All other characters are ignored.

For more information on digital demodulation, see online help and the concepts discussion in the *Operator’s Guide*.

[SENSe:]DDEMod:SEARch:SYNC:STATe command/query

Turns sync search on or off.

Command Syntax: `[SENSe:]DDEMod:SEARch:SYNC:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"SENS:DDEM:SEARCH:SYNC:STATE OFF"
OUTPUT 719;"ddem:sear:sync:stat ON"

Query Syntax: `[SENSe:]DDEMod:SEARch:SYNC:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

For more information on digital demodulation, see online help and the concepts discussion in the *Operator's Guide*.

Related Commands:

- To specify the sync pattern, use DDEM:SEAR:SYNC:PATT.
- To specify the sync offset, use DDEM:SEAR:SYNC:OFFS.

[SENSe:]DDEMod:SEARch:TIME command/query

Specifies the amount of data that is demodulated.

Command Syntax: `[SENSe:]DDEMod:SEARch:TIME <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: depends on configuration
<unit>	::=	S|SYM
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"sens:ddem:search:time +6.70781893E-3S"
OUTPUT 719;"Ddem:Sear:Time 150sym"
```

Query Syntax: `[SENSe:]DDEMod:SEARch:TIME?`

Return Format: `Real`

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

Determines the amount of data that is demodulated when pulse search or sync search is on. You can specify search length in number of symbols or units of time. The amount of data displayed is specified with DDEM:TIME (result length).

DDEM:SEAR:TIME corresponds to *search length* as discussed in the online help.

[SENSe:]DDEMod:SELect — command/query

Selects base-station W-CDMA measurements when the instrument mode is Wideband CDMA.

Command Syntax: `[SENSe:]DDEMod:SELect BASE|A1For|TFOR|SOR`

Example Statements:
OUTPUT 719;":Ddemod:Sel TFOR"
OUTPUT 719;"SENS:DDEMOD:SEL TFOR"

Query Syntax: `[SENSe:]DDEMod:SELect?`

Return Format: `CHAR`

Attribute Summary:

```
Option: B73 (Digital Wideband CDMA analysis (W-CDMA))or
        B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
        080 (Digital 3GPP W-CDMA analysis)

Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

Selects the type of base-station measurement used for W-CDMA measurements.

TFOR and **BASE** configure the analyzer to measure W-CDMA signals that conform to the Japanese Experimental System (also called trial 1998 forward link). BASE is provided for backward compatibility (Requires option B73).

A1FOR configures the analyzer to measure W-CDMA signals that conform to the ARIB 1.0-1.2 standard (Requires option B79).

G3FOR configures the analyzer to measure W-CDMA signals that conform to the 3GPP standard (Requires option 080).

[SENSe:]DDEMod:SRATe command/query

Specifies the digital modulation symbol rate.

Command Syntax: `[SENSe:]DDEMod:SRATe <param>     <param>   ::= <number>[<unit>]|<step>|<bound>     <number>  ::= a real number (NRf data)`
`limits:  depends on configuration     <unit>    ::= HZ     <step> ::= UP|DOWN     <bound>   ::= MAX|MIN`

Example Statements: OUTPUT 719;":ddem:srate 2.43e4"
OUTPUT 719;"Sens:Ddem:Srate 270833 HZ"

Query Syntax: `[SENSe:]DDEMod:SRATe?`

Return Format: `Real`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command corresponds to the *symbol rate* softkey.

For more information on digital demodulation, see online help (press the [**Help**] key on the front panel, then press the softkey of interest) and the concepts discussion in the *Operator's Guide*.

Note In the digital demodulation instrument mode, the maximum span is limited by the MaxSpan/SymbolRate ratio. For a given symbol rate, this ratio limits the maximum span to conserve memory usage. For all demod types except FSK, this ratio is 15.625 and cannot be changed. For the FSK demod type the ratio may be changed to values between 16 and 100, but larger values use more memory. See MEM:MALL:MEAS:MSSR.

[SENSe:]DDEMod:TIME — command/query

Specifies how much demodulated data is displayed.

Command Syntax: `[SENSe:]DDEMod:TIME <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: depends on configuration
<unit>	::=	SYM
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"SENS:DDEMOD:TIME 1.56e2"
OUTPUT 719;"ddem:time 2e3"
```

Query Syntax: `[SENSe:]DDEMod:TIME?`

Return Format: `Real`

Attribute Summary:

```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This parameter corresponds to *result length*, as discussed in the online help. DDEM:SEAR:TIME should be larger than DDEM:TIME. If you set the search length less than the result length, the analyzer displays a warning message and sets the result length equal to the search length.

To specify how much demodulated data is displayed for Wideband CDMA measurements, use DDEMod:MLENgth.

[SENSe:]DDEMod:TIME:CCHannel command/query

Sets the code channel for Wideband CDMA analysis.

Command Syntax: `[SENSe:]DDEMod:TIME:CCHannel {<number>[<unit>]}|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:2047

<unit> ::= [CODE]

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"sense:ddem:time:cchannel 0"
OUTPUT 719;"Ddem:Time:Cch 0"

Query Syntax: `[SENSe:]DDEMod:TIME:CCHannel?`

Return Format: `Integer`

Attribute Summary:

```
Option:  B73 (Digital Wideband CDMA analysis (W-CDMA))or
         B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
         080 (Digital 3GPP W-CDMA analysis)

Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

When the instrument mode is Wideband CDMA, all time-domain measurement data (except composite time-domain data) and symbol table/error summary information is for a single code layer and a single code channel.

This command and the DDEMod:TIME:CLAYer command set the code channel and code layer. For example, if you want the symbol table to show data for channel 3 in code layer 16000 sym/s, send “DDEMod:TIME:CCHannel 3" to select code channel 3; send ”DDEM:TIME:CLAY 16000" to select code layer 16000 sym/s.

NOTE Option 080 use code layer 16000 sym/s.
Options B73 and B79 use code layer 15000 sym/s.

[SENSe:]DDEMod:TIME:CLAYer command/query

Sets the code layer for Wideband CDMA analysis.

Command Syntax: `[SENSe:]DDEMod:TIME:CLAYer <number>|<bound>`

<number> ::= a real number (NRf data)
limits: 8000:4096000

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":DDEM:TIME:CLAY 8000"
OUTPUT 719;"sense:ddem:time:clayer 2048000"

Query Syntax: `[SENSe:]DDEMod:TIME:CLAYer?`

Return Format: `Integer`

Attribute Summary:

```
Option:  B73 (Digital Wideband CDMA analysis (W-CDMA))or
         B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
         080 (Digital 3GPP W-CDMA analysis)

Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

When the instrument mode is Wideband CDMA, all time-domain measurement data (except composite time-domain data) and symbol table/error summary information is for a single code layer and a single code channel.

This command and the DDEMod:TIME:CCHannel command set the code channel and code layer. For example, if you want the symbol table to show data for channel 3 in code layer 16000 sym/s, send “DDEMod:TIME:CCHannel 3" to select code channel 3; send ”DDEM:TIME:CLAY 16000" to select code layer 16000 sym/s.

NOTE Option 080 use code layer 16000 sym/s.
Options B73 and B79 use code layer 15000 sym/s.

The available code layers are determined by the demod format (set with the SENSe:DDEM:SEL command) and the chip rate (set with the SENSe:DDEM:CRATe command).

[SENSe:]DDEMod:TIME:GATE:DELay command/query

Sets the gate delay when time gating is on for Wideband CDMA measurements.

Command Syntax: `[SENSe:]DDEMod:TIME:GATE:DELay {<number>[<unit>]}|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:15 slots

<unit> ::= [S|SLOT]

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"Sens:Ddemod:Time:Gate:Delay 0"
OUTPUT 719;"DDEM:TIME:GATE:DEL 0"

Query Syntax: `[SENSe:]DDEMod:TIME:GATE:DELay?`

Return Format: `Real`

Attribute Summary:
```
Option:  B73 (Wideband CDMA analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

This command sets the gate delay (in whole slots) when time gating is used in the Wideband CDMA instrument mode. For details about time gating, see SENSe:]DDEMod:TIME:GATE:STATe.

[SENSe:]DDEMod:TIME:GATE[:SPAN] command/query

Sets the gate length (span) when time gating is on for Wideband CDMA measurements.

Command Syntax: `[SENSe:]DDEMod:TIME:GATE[:SPAN] {<number>[<unit>]}|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 1:16 slots

<unit> ::= [S|SLOT]

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":ddemod:time:gate 1"
OUTPUT 719;"Sens:Ddem:Time:Gate:Span 4"

Query Syntax: `[SENSe:]DDEMod:TIME:GATE[:SPAN]?`

Return Format: `Real`

Attribute Summary:
```
Option:  B73 (Wideband CDMA analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

This command sets the gate length (in whole slots) when time gating is used in the Wideband CDMA instrument mode. For details about time gating, see SENSe:]DDEMod:TIME:GATE:STATe.

[SENSe:]DDEMod:TIME:GATE:STATe command/query

Turns time gating on or off for Wideband CDMA measurements.

Command Syntax: `[SENSe:]DDEMod:TIME:GATE:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"SENS:DDEM:TIME:GATE:STATE OFF"
OUTPUT 719;"ddem:time:gate:stat ON"

Query Syntax: `[SENSe:]DDEMod:TIME:GATE:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Option:  B73 (Wideband CDMA analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description

When the instrument mode is Wideband CDMA, this command turns time gating on or off. When time gating is on, the analyzer uses the values of DDEMod:TIME:GATE[:SPAN] and DDEMod:TIME:GATE:DELay for the gate length and gate delay.

Gate length sets the width of the gate (in whole slots); gate delay determines the delay, or offset, of the gate (in whole slots). For example, a gate length of 3 and gate delay of 1 displays results for three slots, starting at slot two— in other words, slots 2-4.

For further details about time gating, select the Wideband CDMA instrument mode on the analzyer and see online help for the [gate on/off] softkey (under the [**Time**] hardkey).

[SENSe:]DDEMod:VSB:NSTate command/query

Specifies the number of states in the VSB modulation format.

Command Syntax: `[SENSe:]DDEMod:VSB:NSTate <number>|<step>|<bound>`

<number>	::=	a real number (NRf data) limits: 8:16
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;"SENSE:DDEM:VSB:NST 8"
OUTPUT 719;"ddem:VSB:nst 16"

Query Syntax: `[SENSe:]DDEMod:VSB:NSTate?`

Return Format: `Integer`

Attribute Summary:

```
Option:  AYA and AYH
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

Use this command to select the number of states for the VSB modulation format. Valid values are 8 and 16. To select the VSB modulation format, use DDEM:FORMAT VSB.

For additional information on video demodulation, see the *Operator's Guide*.

[SENSe:]DDEMod:WCDMa:FILTer:ALPHa command/query

Sets the alpha of the root-cosine filter used in W-CDMA measurements.

Command Syntax: `[SENSe:]DDEMod:WCDMa:FILTer:ALPHa <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: .05:100

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":Ddemod:Wcdm:Filt:Alpha 0"
OUTPUT 719;"SENS:DDEMOD:WCDM:FILT:ALPHA 0"

Query Syntax: `[SENSe:]DDEMod:WCDMa:FILTer:ALPHa?`

Return Format: `Real`

Attribute Summary:

```
Option:  B73 (Digital Wideband CDMA analysis (W-CDMA))or
         B79 (Digital ARIB 1.0-1.2 W-CDMA analysis) or
         080 (Digital 3GPP W-CDMA analysis)


Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command sets the alpha of the raised root-cosine filter used in W-CDMA measurements. For details, see the “Wideband CDMA Concepts” chapter in the Operator’s Guide.

[SENSe:]DEMod[1 | 2] command/query

Selects the demodulation result.

Command Syntax: `[SENSe:]DEMod[1|2] OFF|0|AM|PM|FM|BASeband`

Example Statements:
OUTPUT 719;"sens:demod2 BASEBAND"
OUTPUT 719;"Dem AM"

Query Syntax: `[SENSe:]DEMod[1|2]?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Description:

OFF|0 turns demodulation off.

AM turns on AM demodulation.

PM turns on PM demodulation.

FM turns on FM demodulation.

BASeband may be specified for input channel 2 when channel 1 is in demodulation. When this is specified for channel 2, the LO is set to 0 Hz for channel 2. This allows comparison of baseband data on channel 2 with channel 1 demod (baseband) results.

[SENSe:]DEMod[1|2]:CARRier:AUTO command/query

Turns automatic carrier compensation on and off.

Command Syntax: `[SENSe:]DEMod[1|2]:CARRier:AUTO OFF|0|ON|1`

Example Statements:
OUTPUT 719;":DEM2:CARRIER:AUTO OFF"
OUTPUT 719;"sense:dem:carr:auto OFF"

Query Syntax: `[SENSe:]DEMod[1|2]:CARRier:AUTO?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  +1 (ON)
SCPI Compliance:  instrument-specific
```

Description:

This feature determines how the analyzer locks to the carrier frequency. Has no effect on AM demodulation.

ON uses an algorithm to lock to the carrier frequency. The center frequency of the analyzer should be set to or near the frequency of the carrier (the modulation format must not be suppressed-carrier).

OFF assumes the carrier to be at the analyzer’s center frequency. The accuracy of the setting affects phase accuracy in PM demodulation and dc offsets in FM demodulation. This method is useful when the carrier is too small to detect, when the carrier frequency is hopping within the span, or when the analyzer is locked to an external reference which is locked to the carrier. The accuracy required for setting the center frequency on the carrier varies with the span setting, but keeping it within 0.5% of the carrier frequency is recommended for PM.

[SENSe:]DEMod[1|2]:CARRier:AUTO:PM command/query

Specifies the type of automatic carrier compensation for PM demod.

Command Syntax: `[SENSe:]DEMod[1|2]:CARRier:AUTO:PM PAFReq|PHASe`

Example Statements:
OUTPUT 719;"Sens:Demod:Carr:Auto:Pm PAFREQ"
OUTPUT 719;"DEM2:CARRIER:AUTO:PM PAFREQ"

Query Syntax: `[SENSe:]DEMod[1|2]:CARRier:AUTO:PM?`

Return Format: `CHAR`

Attribute Summary:
```
Option:  AYA (vector modulation analysis)
Synchronization Required:  no
Preset State:  PAFR
SCPI Compliance:  instrument-specific
```

Description:

This setting has meaning only when PM demodulation is active and the DEM:CARR:AUTO is ON. It determines how the analyzer detects the PM carrier frequency.

PHASe detects phase offset between the analyzer's local oscillator and the carrier and produces a zero-mean phase record.

PAFReq detects phase and frequency differences. Corrects for phase offset and removes phase ramping due to frequency difference between the analyzer's local oscillator and the carrier.

[SENSe:]DEMod[1|2]:CARRier:FREQ? query

Query the results of the demod carrier display.

Query Syntax: `[SENSe:]DEMod[1|2]:CARRier:FREQ?`

Example Statements:

OUTPUT 719;":demod2:carr:freq?"
OUTPUT 719;"Sens:Dem:Carrier:Freq?"

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The instrument mode must be demod and DISP:WIND:TRAC:DCARR must be ON for there to be data available for this query.

[SENSe:]DETector[:FUNCtion] command/query

Selects the detector method used.

Command Syntax: `[SENSe:]DETector[:FUNCtion] SIGNal|SAMPle|POSitive`

Example Statements:
OUTPUT 719;"SENSE:DET:FUNC SAMPLE"
OUTPUT 719;"detector SIGNAL"

Query Syntax: `[SENSe:]DETector[:FUNCtion]?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  POS
SCPI Compliance:  confirmed
```

Description:

The detector function operates only in scaler instrument mode and only when there are more data points than display points. Then it is used to select which data points are displayed.

SIGNal peak detection tests whether the signal rises and falls within the cell represented by a given data point. When the signal both rises and falls, the odd-numbered data point indicates the maximum value and the even-numbered data point indicates the minimum value encountered during its cell. This algorithm does a good job of combining noise and discrete spectral components, but does not give a true randomness of noise.

SAMPle detection digitizes the instantaneous value of the signal at the end of each cell. It is the best way to indicate the randomness of noise.

POSitive detection insures that all sinusoids maximum value encountered in each cell is displayed. This mode does not give a good representation of random noise either, because it captures the crests of the noise.

[SENSe:]FEED — command/query

Specifies input to be either time capture or receiver channel input.

Command Syntax: `[SENSe:]FEED 'INPut'|'TCAPure'`

Example Statements:
OUTPUT 719;":feed 'INPUT'"
OUTPUT 719;"Sense:Feed 'TCAPTURE'"

Query Syntax: `[SENSe:]FEED?`

Return Format: `STRING`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  INP
SCPI Compliance:  instrument-specific
```

Description:

When time capture is selected, the analyzer is configured to capture time data and store it in memory. Measurements are made on the captured data in memory rather than on the data coming in over the instrument channels (as happens when FEED is INP).

The analyzer cannot capture time data when the current instrument mode is Scalar.

[SENSe:]FREQuency:BASeband command/query

Specifies either zoomed or baseband operation for vector measurements.

Command Syntax: `[SENSe:]FREQuency:BASeband OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Freq:Baseband ON"
OUTPUT 719;"SENS:FREQ:BASEBAND OFF"

Query Syntax: `[SENSe:]FREQuency:BASeband?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +0
SCPI Compliance:  instrument-specific
```

Description:

This command has no effect if INST:SEL is ADEM, DDEM, or SCAL or if ROUT:REC is RF2, COMB, or EXT.

[SENSe:]FREQuency:CENTer — command/query

Specifies the center frequency for the current measurement.

Command Syntax: `[SENSe:]FREQuency:CENTer <param>`

```
<param>  ::= <number>[<unit>]|<step>|<bound>
<number> ::= a real number (NRf data)
             limits: (see table in description)
<unit>   ::= HZ
<step>   ::= UP|DOWN
<bound>  ::= MAX|MIN
```

Example Statements:

```
OUTPUT 719;"Sense:Freq:Cent 2e6"
OUTPUT 719;"FREQUENCY:CENTER 2.34MHZ"
```

Query Syntax: `[SENSe:]FREQuency:CENTer?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  (see table in description)
SCPI Compliance:  confirmed
```

Description:

Center frequency (SENS:FREQ:CENT) and frequency span (SENS:FREQ:SPAN) work together to define the band of frequencies the instrument analyzes. The current value of one parameter is held constant when you change the value of the other. When either center frequency or span is changed, start and stop frequency values change.

Center Frequency Limits and Preset Values

	89410 or 89441 IF	89441 RF
value limits	0:10 MHz	2 MHz:2.650 GHz
preset values	5 MHz	1.328 GHz

IF indicates ROUT:REC IF or RF1 is active.
RF indicates ROUT:REC RF2 is active.

Step size value (used with `UP|DOWN`) is automatically calculated when FREQ:STEP:AUTO is ON and is user-defined when AUTO is OFF. User-definable step size values are specified with FREQ:STEP. Refer to online Help for more information.

Display scaling (lin/log) does not affect this command.

Center frequency cannot be set when ROUT:REC is COMB (I+jQ).

[SENSe:]FREQuency:CENTer:TRACk command/query

Activates signal tracking for either input channel or turns it off.

Command Syntax: `[SENSe:]FREQuency:CENTer:TRACk INP1|INP2|OFF|0`

Example Statements:
OUTPUT 719;"Sense:Freq:Cent:Track INP1"
OUTPUT 719;"FREQ:CENTER:TRAC OFF"

Query Syntax: `[SENSe:]FREQuency:CENTer:TRACk?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  OFF
SCPI Compliance:  instrument-specific
```

Description:

When signal tracking is enabled, the analyzer adjusts the value of the center frequency to keep the largest signal centered in the current frequency span. Selecting either INP1 or INP2 turns on signal tracking.

INP1 turns on signal tracking and uses channel 1 input signal to determine center frequency.

INP2 turns on signal tracking and uses the signal on channel 2 to determine center frequency. INP2 should not be selected unless the (optional) second channel is installed.

Note Tracking is not possible when: averaging is on, manual sweep is active, analog or digital demod is active, capture playback is active, or when ROUT:REC is RF2 or COMB.

[SENSe:]FREQuency:EXTernal:BANDwidth command/query

Specifies the bandwidth of the external downconverter signal.

Command Syntax: `[SENSe:]FREQuency:EXTernal:BANDwidth <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (depends on configuration)
<unit>	::=	HZ
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":SENSE:FREQ:EXT:BANDWIDTH 3 MHZ"
OUTPUT 719;"freq:ext:band 9E6"
```

Query Syntax: `[SENSe:]FREQuency:EXTernal:BANDwidth?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  +3.0E+6
SCPI Compliance:  instrument-specific
```

Description:

This command is effective only when ROUTe:RECeiver is EXT as when using a downconverter such as the Agilent 89411A. See Help Text (under the Help hardkey) for more information.

[SENSe:]FREQuency:EXTernal:CENTer command/query

Specifies the center frequency of the external downconverter signal.

Command Syntax: `[SENSe:]FREQuency:EXTernal:CENTer <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (depends on configuration)
<unit>	::=	HZ
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;":Freq:External:Cent 5.6 MHZ"
OUTPUT 719;"SENSE:FREQUENCY:EXT:CENTER 5E6"

Query Syntax: `[SENSe:]FREQuency:EXTernal:CENTer?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +5.6E+6
SCPI Compliance:  instrument-specific
```

Description:

This command is effective only when ROUTe:RECeiver is EXT as when using a downconverter such as the Agilent 89411A. See Help Text (under the Help hardkey) for more information.

[SENSe:]FREQuency:EXTernal:COMMunicate command/query

Enables or disables the remote control of an external RF or microwave analyzer.

Command Syntax: `[SENSe:]FREQuency:EXTernal:COMMunicate OFF|0|ON|1`

Example Statements:
OUTPUT 719;"sens:frequency:ext:comm ON"
OUTPUT 719;"Frequency:Ext:Communicate OFF"

Query Syntax: `[SENSe:]FREQuency:EXTernal:COMMunicate?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by preset
SCPI Compliance:  instrument-specific
```

Description:

To achieve remote control of the analyzer used as a downconverter front end, the GPIB peripheral address must be set to the address of the external analyzer with FREQ:EXT:COMM:ADDR.

[SENSe:]FREQuency:EXTernal:COMMunicate:ADDRess command/query

Specifies the address of the external analyzer used as a downconverter.

Command Syntax: `[SENSe:]FREQuency:EXTernal:COMMunicate:ADDress <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:30
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;":FREQ:EXT:COMMUNICATE:ADDR 8"
OUTPUT 719;"sense:freq:ext:communicate:addr 15"

Query Syntax: `[SENSe:]FREQuency:EXTernal:COMMunicate:ADDRess?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command is useful only when ROUT:REC is EXT and FREQ:EXT:COMM is ON.

[SENSe:]FREQuency:EXTernal:MAXimum command/query

Specifies the the maximum frequency to which an external receiver can be tuned.

Command Syntax: `[SENSe:]FREQuency:EXTernal:MAXimum <param>`

<param> ::= <number>[<unit>]|<step>|<bound>

<number> ::= a real number (NRf data)
limits: (depends on configuration)

<unit> ::= HZ

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

OUTPUT 719;"sens:freq:ext:max 15 GHZ"
OUTPUT 719;":Frequency:Ext:Max 1.5E10"

Query Syntax: `[SENSe:]FREQuency:EXTernal:MAXimum?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  +2.2E+10
SCPI Compliance:  instrument-specific
```

Description:

This command is effective only when ROUTe:RECeiver is EXT as when using a downconverter such as the Agilent 89411A. See Help Text (under the Help hardkey) for more information.

[SENSe:]FREQuency:EXTernal:MINimum command/query

Specifies the the minimum frequency to which an external receiver can be tuned.

Command Syntax: `[SENSe:]FREQuency:EXTernal:MINimum <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (depends on configuration)
<unit>	::=	HZ
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;"sens:freq:ext:min 100 KHZ"
OUTPUT 719;":Frequency:Ext:Min 1E6"

Query Syntax: `[SENSe:]FREQuency:EXTernal:MINimum?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0E0
SCPI Compliance:  instrument-specific
```

Description:

This command is effective only when ROUTe:RECeiver is EXT as when using a downconverter such as the Agilent 89411A. See Help Text (under the Help hardkey) for more information.

[SENSe:]FREQuency:EXTernal:MIRRor command/query

Lets you reverse the spectrum when using the external or ch1+j*ch2 receiver.

Command Syntax: `[SENSe:]FREQuency:EXTernal:MIRRor OFF|0|ON|1`

Example Statements:
```
OUTPUT 719;"Sense:Freq:Ext:Mirror OFF"
OUTPUT 719;"FREQ:EXTERNAL:MIRR ON"
```

Query Syntax: `[SENSe:]FREQuency:EXTernal:MIRRor?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

This command is effective only when ROUTe:RECeiver is EXT as when using a downconverter such as the Agilent 89411A, or when ROUTe:RECeiver is COMBine (COMBine selects the ch1+j*ch2 receiver). See Help Text (under the Help hardkey) for more information.

[SENSe:]FREQuency:MANual — command/query

This command specifies the frequency when manual sweep is active.

Command Syntax: `[SENSe:]FREQuency:MANual <param>`

```
<param>   ::= <number>[<unit>]|<step>|<bound>
<number>  ::= a real number (NRf data)
              limits: (see table in description)
<unit>    ::= HZ
<step>    ::= UP|DOWN
<bound>   ::= MAX|MIN
```

Example Statements:

OUTPUT 719;"Sens:Frequency:Man 5.00e3"
OUTPUT 719;"FREQ:MANUAL 5685325HZ"

Query Syntax: `[SENSe:]FREQuency:MANual?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  (see table in description)
SCPI Compliance:  confirmed
```

Description:

Related Commands:

- Manual sweep is available only when instrument mode is scalar (INST[:SEL] SCAL).
- Manual sweep is activated with the [SENS:]SWE:MODE MAN command.
- The step size for UP|DOWN is defined with the FREQ:STEP command if FREQ:STEP:AUTO is OFF. When FREQ:STEP:AUTO is ON, the step size tracks that of center frequency.
- The manual frequency value is constrained to be within the start and stop frequency settings (FREQ:STAR and FREQ:STOP).

Manual Frequency Limits and Preset Values

	89410 or 89441 IF	89441 RF
value limits	0:10 MHz	2 MHz:2.650 GHz
preset values	5 MHz	1.328 GHz

IF indicates ROUT:REC IF or RF1 is active.
RF indicates ROUT:REC RF2 is active.

[SENSe:]FREQuency:SPAN command/query

Specifies the frequency span to be measured.

Command Syntax: `[SENSe:]FREQuency:SPAN <param>`

```
<param>   ::= <number>[<unit>]|<step>|<bound>
<number>  ::= a real number (NRf data)
              limits: (see table in description)
<unit>    ::= HZ
<step>    ::= UP|DOWN
<bound>   ::= MAX|MIN
```

Example Statements:

OUTPUT 719;"frequency:span 2e6"
OUTPUT 719;"Sens:Frequency:Span 5.5 MHz"

Query Syntax: `[SENSe:]FREQuency:SPAN?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  (see table in description)
SCPI Compliance:  confirmed
```

Description:

SENS:FREQ:SPAN and SENS:FREQ:CENT work together to define the band of frequencies to analyze.

Frequency Span Limits and Preset Values

	89410 or 89441 IF	89441 RF Scalar	89441 RF Vector	89410/41 I + jQ
value limits	1.0 Hz:10 MHz	1.0 Hz:2.648 GHz	1.0 Hz:7 MHz†	2.0 Hz:20 MHz
preset values	10 MHz	2.648 GHz	7 MHz	20 MHz

† Maximum frequency span is 8 MHz for Agilent 89441A analyzers that have options AYA and AYH (see FREQ:SPAN:WIDE for details).
IF indicates ROUT:REC IF or RF1 is active.
RF indicates ROUT:REC RF2 is active.
I+jQ indicates ROUT:REC COMB is active.

Step size (used with `UP|DOWN`) is *not* user-definable for FREQ:SPAN.

Note In the digital demodulation instrument mode, the maximum span is limited by the MaxSpan/SymbolRate ratio. For a given symbol rate, this ratio limits the maximum span to conserve memory usage. For all demod types except FSK, this ratio is 15.625 and cannot be changed. For the FSK demod type the ratio may be changed to values between 16 and 100, but larger values use more memory. See MEM:MALL:MEAS:MSSR.

[SENSe:]FREQuency:SPAN:FULL command

Sets the analyzer to the widest frequency span possible.

Command Syntax: `[SENSe:]FREQuency:SPAN:FULL`

Example Statements:

```
OUTPUT 719;":freq:span:full"
OUTPUT 719;"Sense:Freq:Span:Full"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

Sets the maximum frequency span possible, depending on instrument mode.

For the 89410A:

- When ROUT:REC is not COMB, full span is 10 MHz.
- When ROUT:REC is COMB, full span is 20 MHz.

For the 89441A:

- When ROUT:REC is RF1 or IF, full span is 10 MHz.
- When ROUT:REC is COMB, full span is 20 MHz.
- When ROUT:REC is RF2 and instrument mode is *vector* or *analog demodulation*, full span is 7 MHz. If you have an Agilent 89441A with option AYH, you can extend full span to 8 MHz (see FREQ:SPAN:WIDE for details).
- When ROUT:REC is RF2 and instrument mode is *scalar*, full span is:
  - **89441A** 2.648 GHz

[SENSe:]FREQuency:SPAN:PCHirp command/query

Determines the period of the periodic chirp source type.

Command Syntax: `[SENSe:]FREQuency:SPAN:PCHirp EXACt|NEARest`

Example Statements:
OUTPUT 719;"SENS:FREQUENCY:SPAN:PCH NEAREST"
OUTPUT 719;"frequency:span:pchirp NEAREST"

Query Syntax: `[SENSe:]FREQuency:SPAN:PCHirp?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This feature is active only when the source type is periodic chirp, RBW mode is arb, and RBW coupling is auto. See help text for further information.

[SENSe:]FREQuency:SPAN:WIDE command/query

Applicable only to Agilent 89441A analyzers that have option AYH; extends the maximum frequency span to 8 MHz.

Command Syntax: `[SENSe:]FREQuency:SPAN:WIDE OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Freq:Span:Wide OFF"
OUTPUT 719;"SENS:FREQUENCY:SPAN:WIDE ON"

Query Syntax: `[SENSe:]FREQuency:SPAN:WIDE?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AYA (Vector Modulation Analysis) and AYH (Digital Video
Modulation Analysis)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This GPIB command is applicable only to Agilent 89441A analyzers that have option AYH. This GPIB command corresponds to the [RF (2-1800 MHz) wide] softkey and determines the maximum frequency span available when ROUT:REC RF2 is selected.

When ROUT:REC RF2 is selected, sending ON extends the maximum frequency span to 8 MHz. Sending OFF (default selection) returns the maximum frequency span to 7 MHz.

There are several details you need to know when using the 8 MHz frequency span. For details, see the *Operator's Guide* and see online help for the [**Instrument Mode**] [receiver] [RF (2-1800 MHz)] softkey.

[SENSe:]FREQuency:STARt command/query

Defines the start (lowest) frequency for the measurement band.

Command Syntax: `[SENSe:]FREQuency:STARt <param>`

```
<param>   ::= <number>[<unit>]|<step>|<bound>
<number>  ::= a real number (NRf data)
              limits: (see table in description)
<unit>    ::= HZ
<step>    ::= UP|DOWN
<bound>   ::= MAX|MIN
```

Example Statements:

```
OUTPUT 719;"SENS:FREQUENCY:STAR 50000"
OUTPUT 719;"frequency:star 2e+4"
```

Query Syntax: `[SENSe:]FREQuency:STARt?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  (see table in description)
SCPI Compliance:  confirmed
```

Description:

Start frequency (SENS:FREQ:STAR) and stop frequency (SENS:FREQ:STOP) work together to define the band of frequencies the instrument analyzes. SENS:FREQ:STAR defines the band's lower limit, SENS:FREQ:STOP defines its upper limit.

Start Frequency Limits and Preset Values

	89410 or 89441 IF	89441 RF	
		Scalar	Vector
value limits	–5 MHz:10 MHz	–1.324 GHz:2.65 GHz	–1.5 MHz:2.65 GHz
preset values	0 Hz	2 MHz	1.328 GHz

IF indicates ROUT:REC IF or RF1 is active.
RF indicates ROUT:REC RF2 is active.

If ROUT:REC is not COMB, start and stop values are independent except STARt + 1.0 HZ ≤ STOP.

When ROUT:REC is COMB, only span and stop may be specified and START = –STOP. The stop range is 1 Hz:10 MHz and the preset value is 10 MHz.

Step size value (used with `UP|DOWN`) is automatically calculated when FREQ:STEP:AUTO is ON and is user-defined when AUTO is OFF. User-definable step size values are specified with FREQ:STEP. Refer to online Help for more information.

[SENSe:]FREQuency:STEP:AUTO command/query

Specifies whether step size is automatic or not.

Command Syntax: `[SENSe:]FREQuency:STEP:AUTO OFF|0|ON|1`

Example Statements:
OUTPUT 719;"sense:freq:step:auto ON"
OUTPUT 719;"Freq:Step:Auto ON"

Query Syntax: `[SENSe:]FREQuency:STEP:AUTO?`

Return Format: `Integer, Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  ON
SCPI Compliance:  instrument-specific
```

Description:

When step size is automatic (AUTO ON), the step size is determined by the analyzer for center frequency, start frequency, stop frequency, source-sine frequency, and manual-sweep frequency. Frequency *span* is *always* automatic. See online Help for more information.

User-defined step size is entered with the FREQ:STEP[:INCR] command. Steps to increment or decrement a value are accomplished with the UP or DOWN parameters.

[SENSe:]FREQuency:STEP[:INCRement] command/query

Specifies the step size to be used for changing frequency parameters.

Command Syntax: `[SENSe:]FREQuency:STEP[:INCRement] <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (see table in description)
<unit>	::=	HZ
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;":FREQUENCY:STEP 0"
OUTPUT 719;"sens:frequency:step:increment 0"

Query Syntax: `[SENSe:]FREQuency:STEP[:INCRement]?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  (see table)
SCPI Compliance:  instrument-specific
```

Description:

Step size value (used with `UP|DOWN`) is automatically calculated when FREQ:STEP:AUTO is ON and is user-defined when AUTO is OFF. User-definable step size values are specified with FREQ:STEP. Refer to online Help for more information.

A user-defined step size value may be used with the following commands:

- SENS:FREQ:CENT
- SENS:FREQ:STAR
- SENS:FREQ:STOP
- SOUR:FREQ[:CW]
- FREQ:MAN

Step size for frequency span is not user-definable.

Step Size Limits and Preset Values

	89410 or 89441 IF	89441 RF
value limits	.001 Hz:10 MHz	.001 Hz:2.650 GHz
preset values	25 kHz	6.63 MHz

IF indicates ROUT:REC IF or RF1 is active.
RF indicates ROUT:REC RF2 is active.

[SENSe:]FREQuency:STOP — command/query

Specifies the stop (highest) frequency in the measurement band.

Command Syntax: `[SENSe:]FREQuency:STOP <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (see table in description)
<unit>	::=	HZ
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"SENS:FREQUENCY:STOP 50000"
OUTPUT 719;"frequency:stop 2e+5"
```

Query Syntax: `[SENSe:]FREQuency:STOP?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  (see table in description)
SCPI Compliance:  confirmed
```

Description:

Start frequency (SENS:FREQ:STAR) and stop frequency (SENS:FREQ:STOP) work together to define the band of frequencies the instrument analyzes. SENS:FREQ:STAR defines the band's lower limit, SENS:FREQ:STOP defines its upper limit.

Stop Frequency Limits and Preset Values

	89410 or 89441 IF	89441 RF	
		Scalar	Vector
value limits	.5 Hz:15 MHz	2 MHz:3.98 GHz	2 MHz:2.6535 GHz
preset values	10 MHz	2.650 GHz	1.3315 GHz

IF indicates ROUT:REC IF or RF1 is active.
RF indicates ROUT:REC RF2 is active.

If ROUT:REC is not COMB, start and stop values are independent except STARt + 1.0 HZ ≤ STOP.

When ROUT:REC is COMB, only span and stop may be specified and START = –STOP. The stop range is 1 Hz:10 MHz and the preset value is 10 MHz.

Step size value (used with `UP|DOWN`) is automatically calculated when FREQ:STEP:AUTO is ON and is user-defined when AUTO is OFF. User-definable step size values are specified with FREQ:STEP. Refer to online Help for more information.

[SENSe:]SWEep[1|2]:MODE command/query

Selects between automatic (linear) and manual sweep.

Command Syntax: `[SENSe:]SWEep[1|2]:MODE AUTO|MANual`

Example Statements:
OUTPUT 719;"Sens:Swe2:Mode AUTO"
OUTPUT 719;"SWE2:MODE MANUAL"

Query Syntax: `[SENSe:]SWEep[1|2]:MODE?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  AUTO
SCPI Compliance:  confirmed
```

Description:

Manual sweep is used to make a measurement at a single frequency. It allows the user to manually set the frequency of the local oscillator normally used to sweep a measurement.

Related Commands:

- The MANual selection is valid *only* when the instrument type is scalar (INST[:SEL] SCAL).
- The manual frequency value is specified with the [SENS:]FREQ:MAN command.

[SENSe:]SWEep[1 | 2]:OVERlap **command/query**

Specifies how much (consecutive) time blocks are allowed to overlap when averaging is on.

Command Syntax: `[SENSe:]SWEep[1|2]:OVERlap <number>[<unit>]|<step>|<bound>`

<number>	::=	a real number (NRF data) limits: 0:99.99
<unit>	::=	PCT
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;"SENSE:SWE:OVER 30PCT"
OUTPUT 719;"sweep:over 52.5"

Query Syntax: `[SENSe:]SWEep[1|2]:OVERlap?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0.00%
SCPI Compliance:  instrument-specific
```

Description:

Averaging overlapped data blocks allows more accurate results to be acquired in less time. Units are assumed to be percent.

This command specifies the maximum possible overlap. It corresponds to overlap: averaging on.

The numbers in the sweep node which normally specify an input channel for the command to affect, has no channel-specific affect for this command; the specified overlap is the same for all installed input channels.

[SENSe:]SWEep[1 | 2]:POINts **command/query**

Specifies the number of alias-protected frequency points.

Command Syntax: `[SENSe:]SWEep:POINts <number>|<step>|<bound>`

<number> ::= 51|101|201|401|801|1601|3201

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;":Sweep:Poin 101"
OUTPUT 719;"SENS:SWEEP:POIN 801"
```

Query Syntax: `[SENSe:]SWEep[1|2]:POINts?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  401
SCPI Compliance:  confirmed
```

Description:

Changing the number of frequency points affects the display resolution as follows.

$$\text{display resolution} = \frac{span}{number\ of\ points - 1}$$

[SENSe:]SWEep[1 | 2]:TIME:DELay command/query

Specifies the delay between the trigger and the beginning of the time record.

Command Syntax: `[SENSe:]SWEep[1|2]:TIME:DELay <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (see discussion)
<unit>	::=	S
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
```
OUTPUT 719;"Sens:Sweep1:Time:Delay 5ms"
OUTPUT 719;"SWE1:TIME:DEL (SWE1:TIME:GATE:DEL?)"
```

Query Syntax: `[SENSe:]SWEep[1|2]:TIME:DELay?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 S
SCPI Compliance:  instrument-specific
```

Description:

The sweep node numbers (1|2) specify which input (receiver) channel is affected by the command. If no number is specified, the command is applied to channel 1. Note that channel 2 is optional (AY7) and may not be installed in the analyzer. To determine whether the second channel is installed, send *OPT? and see if AY7 (second channel) is returned.

Limits:

Pretrigger "delay" is the amount of time *before* the trigger occurs that you want data; the trigger occurs after the point in time that you want to begin collecting data. This is specified by entering a negative number and is limited by the size of the time capture RAM available for the measurement.

Without the time capture RAM (option AY9), there are 64K samples (65,536) available. When a second channel is installed and enabled, the pretrigger delay limit is half what it is when only one channel is enabled; 32K per channel. With deep capture RAM, option AY9, there are 1M samples (1,048,576) available; .5M per channel if the optional second channel is enabled. The amount of time represented by these sample sizes depends on the time each sample represents, which varies with span (Δt).

Post-trigger delay is the amount of time *after* the trigger occurs that you want to begin the time record. This is limited by the range of a counter that can count to 2G (2,147,483,648) samples. This parameter is not affected by the presence of the capture RAM or second-channel options. Again, the time represented by this count depends on the time each sample represents, which varies with span (Δt).

In scalar mode, the analyzer picks the span used for the measurement. This affects the resolution in that Δt becomes a function of measurement parameters other than the user-specified span.

[SENSe:]SWEep[1|2]:TIME:GATE:DELay command/query

Specifies when the time gate begins relative to the beginning of the main time record.

Command Syntax:

```
[SENSe:]SWEep[1|2]:TIME:GATE:DELay <param>
```

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

Example Statements:

```
OUTPUT 719;":swe:time:gate:del 15 us"
OUTPUT 719;"Sense:Swe:Time:Gate:Del 40e-3"
```

Query Syntax:

```
[SENSe:]SWEep[1|2]:TIME:GATE:DELay?
```

Return Format: Real

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0 S
SCPI Compliance:  instrument-specific
```

Description:

The sweep node numbers (1|2) specify which input (receiver) channel is affected by the command. If no number is specified, the command is applied to channel 1. Note that channel 2 is optional (AY7) and may not be installed in the analyzer. To determine whether the second channel is installed, send *OPT? and see if AY7 (second channel) is returned.

Note Time operations do not function when the instrument mode is *scalar*. When the measurement is zoomed, a local oscillator signal is mixed with the measurement data and time data is not baseband. Time gating is not available in digital demod.

Limits: The time gate is defined by the *gate delay*, which defines its starting time relative the the beginning of the main time record, and *gate length*, which defines when it stops relative to its start. Both start and stop values must be within the main time record, also called *main time*. So the lower limit of the gate delay is zero and the upper limit depends on the gate length defined with SWE:TIME:GATE[:SPAN] as follows:

maximum gate delay = main length – gate length

The minimum gate delay is 0.

Related Commands:

- To set the gate delay step size, use SWE:TIME:GATE:DEL:STEP.
- To turn the gate on, use SWE:TIME:GATE:STATE.
- To set the gate time, use SWE:TIME.

[SENSe:]SWEep[1|2]:TIME:GATE:DELay:STEP[:INCRement] command/query

Specifies the step size used to increment or decrement the time gate delay.

Command Syntax: `[SENSe:]SWEep[1|2]:TIME:GATE:DELay:STEP[:INCRement] <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (see discussion)
<unit>	::=	S
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":swe:time:gate:del:step 0"
OUTPUT 719;"Sens:Swe:Time:Gate:Delay:Step:Incr 0"
```

Query Syntax: `[SENSe:]SWEep[1|2]:TIME:GATE:DELay:STEP[:INCRement]?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  (see discussion)
SCPI Compliance:  instrument-specific
```

Description:

The numbers in the sweep node which normally specify an input channel for the command to affect, has no channel-specific affect for this command; the step value is the same for all installed input channels.

Note Time operations do not function when the instrument mode is *scalar*. When the measurement is zoomed, a local oscillator signal is mixed with the measurement data and time data is not baseband. Time gating is not available in digital demod.

Limits:

The minimum step size is 0.

The maximum limit is *main time length – gate length*.

The resolution is Δt.

Preset Values:

When ROUT:REC is INP, IF, RF1, or COMB, the preset value is 39.0625 ns

When ROUT:REC is RF2, the preset value is 111.607143 ns

When ROUT:REC is EXT, the preset value is 260.41667 ns.

[SENSe:]SWEep[1 | 2]:TIME:GATE[:SPAN] command/query

Specifies the gate time length, in seconds.

Command Syntax:

```
[SENSe:]SWEep:TIME:GATE[:SPAN] <param>
```

<param> ::= <number>[<unit>]|<step>|<bound>

<number> ::= a real number (NRf data)
limits: (see discussion)

<unit> ::= S

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:

```
OUTPUT 719;"SENSE:SWE2:TIME:GATE:SPAN 5E-6"
OUTPUT 719;"sweep:time:gate 74 ms"
```

Query Syntax:

```
[SENSe:]SWEep[1|2]:TIME:GATE[:SPAN]?
```

Return Format: Real

Attribute Summary:

```
Synchronization Required:  no
Preset State:  12.734375 us
SCPI Compliance:  instrument-specific
```

Description:

The numbers in the sweep node which normally specify an input channel for the command to affect, has no channel-specific affect for this command; the gate length is the same for all installed input channels.

Note Time operations do not function when the instrument mode is *scalar*. When the measurement is zoomed, a local oscillator signal is mixed with the measurement data and time data is not baseband. Time gating is not available in digital demod.

The preset value depends on the number of frequency points in use. If the maximum number of frequency points is set below 401, the preset value for gate length may be smaller than 3.828125 µs (the preset value for 1 MHz RBW with the default window type, flattop).

Limits: The lower limit depends on window type and whether the measurement is zoomed as shown in the following table; value differences are due to finer time resolution in baseband. The upper limit is the current main time record length.

Gate length lower limits

Window Type	Baseband	Zoom
uniform	351.5625 ns	390.625 n
Hanning	507.8125 ns	546.875 ns
gaussian top	742.1875 ns	781.250 ns
flat top	1.2890625 us	1.328125 us

[SENSe:]SWEep[1|2]:TIME:GATE:STATe — command/query

Turns time gating on or off.

Command Syntax: `[SENSe:]SWEep[1|2]:TIME:GATE:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"SENSE:SWE2:TIME:GATE:STAT OFF"
OUTPUT 719;"sweep2:time:gate:stat ON"

Query Syntax: `[SENSe:]SWEep[1|2]:TIME:GATE:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

When the gate is ON, frequency analysis is performed on the gate time record.

The numbers in the sweep node which normally specify an input channel for the command to affect, has no channel-specific affect for this command; the gate state is the same for all installed input channels.

Note Time operations do not function when the instrument mode is *scalar*. When the measurement is zoomed, a local oscillator signal is mixed with the measurement data and time data is not baseband. Time gating is not available in digital demod.

[SENSe:]SWEep[1 | 2]:TIME:OVERlap command/query

Specifies how much (consecutive) time blocks are allowed to overlap when averaging is off.

Command Syntax: `[SENSe:]SWEep[1|2]:TIME:OVERlap {<number>[<unit>]}|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:99.99

<unit> ::= [PCT]

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":Swe2:Time:Over 40.2607"
OUTPUT 719;"SENSE:SWE:TIME:OVERLAP 90.4304"

Query Syntax: `[SENSe:]SWEep[1|2]:TIME:OVERlap?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  98%
SCPI Compliance:  instrument-specific
```

Description:

The amount of overlap can be used to control the amount of information displayed in a waterfall or spectrogram display. It can also be used to control the time capture playback speed.

This command specifies the maximum possible overlap. It corresponds to overlap: averaging off.

The numbers in the sweep node which normally specify an input channel for the command to affect, has no channel-specific affect for this command; the specified overlap is the same for all installed input channels.

Related Commands:

- To specify amount of overlap processing used when averaging is on, use [SENS:]SWE:OVER.

[SENSe:]SWEep[1 | 2]:TIME:RESolution:AUTO **command/query**

Automatically reduces span as time record length is increased past the point where the existing span accommodates the requested time record length.

Command Syntax: `[SENSe:]SWEep[1|2]:TIME:RESolution:AUTO OFF|0|ON|1`

Example Statements:
OUTPUT 719;"sens:sweep:time:res:auto OFF"
OUTPUT 719;"Swe:Time:Res:Auto OFF"

Query Syntax: `[SENSe:]SWEep[1|2]:TIME:RESolution:AUTO?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

The numbers in the sweep node which normally specify an input channel for the command to affect, has no channel-specific affect for this command; if AUTO is ON, it affects all installed input channels.

Note Time operations do not function when the instrument mode is *scalar*. When the measurement is zoomed, a local oscillator signal is mixed with the measurement data and time data is not baseband. Time gating is not available in digital demod.

Span determines time sample spacing and the number of frequency points determines the maximum record size (meaning points, not seconds). As the main time record length increases, more and more time samples are required. When the number of samples exceeds the maximum time record size, either the sample rate may remain the same and the main length be limited or the sample rate may be reduced, allowing main length to grow.

If AUTO is OFF, the sample rate (and thus span) is not changed to accommodate long main time lengths.

If AUTO is ON, the sample rate (and thus span) is adjusted to acquire the most time points possible without exceeding the maximum time points allowed by the number of frequency points set with SWEep:POINts.

[SENSe:]SWEep[1 | 2]:TIME[:SPAN] command/query

Specifies the time record length, also known as main time length.

Command Syntax: `[SENSe:]SWEep[1|2]:TIME[:SPAN] <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (see description)
<unit>	::=	S
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;":SWEEP2:TIME (SWE2:TIME:gate?)"
OUTPUT 719;"sens:sweep:time:span 4e-2"

Query Syntax: `[SENSe:]SWEep[1|2]:TIME[:SPAN]?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  38.203125 us
SCPI Compliance:  instrument-specific
```

Description:

The numbers in the sweep node which normally specify an input channel for the command to affect, has no channel-specific affect for this command; the step value is the same for all installed input channels.

Note Time operations do not function when the instrument mode is *scalar*. When the measurement is zoomed, a local oscillator signal is mixed with the measurement data and time data is not baseband. Time gating is not available in digital demod.

Limits: The size of the time record is a function of the number of points in the time record and the amount of time each point represents (inverse of the effective sample rate). The effective sample rate is a function of span. The limits, in terms of time, vary with the span, window type, and number of frequency points. Usually, you will pick a span and then pick either record length or RBW. Changing either parameter changes the other. So the limits depend on the current span. The *maximum* time record length occurs at maximum frequency points (3201), minimum span (1 Hz), and minimum resolution bandwidth (700 µHz; minimum RBW occurs in arbitrary RBW mode). The time record length at these settings is 3.2 ks. The *minimum* time record length occurs at the maximum span (10 MHz) and maximum RBW (3 MHz). The time record length at these settings is 351.5625 ns for the uniform window, and 1.2890625 µs for flattop window. The number of frequency points is not a consideration for the minimum time record length because of a limit on the maximum RBW/span ratio.

If SWE:TIME:AUTO is ON, increasing the time record length decreases the span automatically when span becomes the limiting factor. The preset value is OFF.

[SENSe:]TCAPture[1 | 2]:ABORt **command**

Stops a time capture that is in progress.

Command Syntax: `[SENSe:]TCAPture[1|2]:ABORt`

Example Statements:
OUTPUT 719;":TCAPTURE2:ABOR"
OUTPUT 719;"sense:tcap:abor"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command aborts filling the time capture buffer for both channel inputs, regardless of the channel specified in the TCAP node.

If the FEED is not TCAP, this command has no effect.

[SENSe:]TCAPture[1|2]:DIRection — command/query

Specifies the time capture playback direction.

Command Syntax: `[SENSe:]TCAPture[1|2]:DIRection FORWard|REVerse`

Example Statements:

OUTPUT 719;"Sense:Tcap:Direction REVERSE"
OUTPUT 719;"TCAP:DIR REVERSE"

Query Syntax: `[SENSe:]TCAPture[1|2]:DIRection?`

Return Format: `CHAR`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

[SENSe:]TCAPture[1 | 2][:IMMediate] command

Begins a time capture.

Command Syntax: `[SENSe:]TCAPture[1|2][:IMMediate]`

Example Statements:
OUTPUT 719;":tcapture2"
OUTPUT 719;"Sens:Tcapture:Imm"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This command initiates filling the time capture buffer for both channel inputs, regardless of the channel specified in the TCAP node. This command also changes the input feed to TCAPture.

To start analysis of the captured data, send ABORt.

Related Commands:

- To stop time catpure, use TCAP:ABORt.
- To specify the receiver input, use FEED INP.
- To display the entire time capture buffer for channel 1, use CALC:FEED ‘TCAP 1’
- To store time capture data to disk, use MMEM:STORE:TCAP.

[SENSe:]TCAPture[1|2]:LENGth — command/query

Sets the length of the time capture record.

Command Syntax: `[SENSe:]TCAPture[1|2]:LENGth <param>`

```
<param>   ::=  <number>[<unit>]|<step>|<bound>
<number>  ::=  a real number (NRf data)
                limits: (see discussion)
<unit>    ::=  S|POINTS|RECORDS
<step>    ::=  UP|DOWN
<bound>   ::=  MAX|MIN
```

Example Statements:

```
OUTPUT 719;"SENS:TCAPTURE:LENG 0"
OUTPUT 719;"tcapture:leng 0"
```

Query Syntax: `[SENSe:]TCAPture[1|2]:LENGth?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  +3.2768E+4 (POINTS)
SCPI Compliance:  instrument-specific
```

Description:

Option AY9 adds 1 Megasample RAM for time capture.

Limits: The limits are defined by a number of things. The first limit is set by the amount of installed time capture memory. Without the optional time capture RAM, the capacity is 32 KSamples in two channels. (If the analyzer has only one channel or channel 2 is off, the capacity is 64 KSamples.) With the optional time capture RAM, the capacity is 512 KSamples in two channels. (If the analyzer has only one channel or channel 2 is off, the capacity is 1024 KSamples or 1 MSample.)

Each sample takes 8 bytes, 4 real and 4 imaginary. If the data is moved out of the analyzer, baseband data is real-only (4 bytes) and zoomed data is real/imaginary (4/4 bytes). If data is captured and transferred out of the analyzer, be aware that the capture was done in /2 spans. When the captured data is analyzed in the analyzer, this limitation is eliminated by resampling to get arbitrary (other-than- /2) spans.

To determine how much time is available (limits in terms of seconds) you must also consider the span and resolution bandwidth settings. The relationship between these values is as follows: $L = N \times \Delta t$, where L is record length, N is the number of samples, and the time between samples is $\Delta t = \frac{1}{(span \times 2.56)}$. We also know that $N = \frac{k \times 2.56 \times span}{RBW}$ where k is window bandwidth, and RBW is resolution bandwidth.

Substituting for N and Δt we get $L = \frac{k}{RBW}$ as long as N is not larger than the installed memory (standard = 32 KSamples, option = 1 MSample). See concepts for values of k.

[SENSe:]TCAPture[1|2]:POSition? query

Returns the current position in the time capture buffer.

Query Syntax: `[SENSe:]TCAPture[1|2]:POSition?`

Example Statements:

OUTPUT 719;":Tcap2:Position?"
OUTPUT 719;"SENS:TCAPTURE:POS?"

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

[SENSe:]TCAPture[1|2]:RANGe command/query

Specifies range value for time capture data.

Command Syntax: `[SENSe:]TCAPture[1|2]:RANGe <number>|<bound>`

<number> ::= a real number (NRf data)
limits: 1.0E-20:1.0E20

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"sens:tcapture2:rang 4.01739e+19"
OUTPUT 719;"Tcapture2:Rang 9.85167e+19"

Query Syntax: `[SENSe:]TCAPture[1|2]:RANGe?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Description:

The range must be specified before time data is loaded into the capture buffers from a controller. This value (in Volts) is used as a reference to convert the data to integer format. To get full 32-bit accuracy, the value sent should be the same as the largest data value to be loaded.

Note Any data values larger than the range value will be clipped.

Related Commands:

- To load data into the analyzer, use [SENS:]DATA.
- To define the data format for the transfer, use FORM:DATA.

[SENSe:]TCAPture[1 | 2]:STARt **command/query**

Specifies the position in the time capture buffer to begin analysis.

Command Syntax: `[SENSe:]TCAPture[1|2]:STARt <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (depends on configuration)
<unit>	::=	S|POINTS|RECORDS
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":TCAP2:START 0"
OUTPUT 719;"sens:tcapture:star 0"
```

Query Syntax: `[SENSe:]TCAPture[1|2]:STARt?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0 (seconds)
SCPI Compliance:  instrument-specific
```

Description:

Related Commands:

- To specify the end of the time capture data on which to performa analysis, use TCAP:STOP.
- To begin filling the time capture buffer(s), use TCAP.
- To specify the size of the time capture in points, use TCAP:LENG.

[SENSe:]TCAPture[1 | 2]:STOP — command/query

Specifies the position in the waveform capture buffer to end analysis.

Command Syntax: `[SENSe:TCAPture[1|2]:STOP <param>`

<param>	::=	<number>[unit]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (limits depend on configuration)
<unit>	::=	S|POINTS|RECORDS
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;"Sens:Tcapture:Stop 0"
OUTPUT 719;"TCAPTURE2:STOP 0"

Query Syntax: `[SENSe:]TCAPture[1|2]:STOP?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0
SCPI Compliance:  instrument-specific
```

Description:

After filling the time capture buffer, this parameter is set to the end of the buffer.

Related Commands:

- To specify the start of the time capture data on which to performa analysis, use TCAP:STAR.
- To begin filling the time capture buffer(s), use TCAP.
- To specify the size of the time capture in points, use TCAP:LENG.

[SENSe:]VOLTage[1 | 2][:DC]:RANGe:AUTO command/query

Turns autoranging on or off.

Command Syntax: `[SENSe:]VOLTage[1|2][:DC]:RANGe:AUTO OFF|0|ON|1|ONCE`

Example Statements:
OUTPUT 719;":volt2:range:auto OFF"
OUTPUT 719;"Sense:Volt:Dc:Range:Auto OFF"

Query Syntax: `[SENSe:]VOLTage[1|2][:DC]:RANGe:AUTO?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  confirmed
```

Description:

Related commands:

- To set the autorange direction, use SENS:VOLT:RANG:AUTO:DIR.
- To execute one autorange, send SENS:VOLT:RANG:AUTO ONCE. The analyzer then samples the input signal value and selects a fixed range setting (autoranging is turned off).

The voltage node numbers (1|2) specify which input (receiver) channel is affected by the command. In a single-channel 89441A, the channel specifier does nothing. If no number is specified, the command is applied to channel 1. Note that channel 2 is optional (AY7) and may not be installed in the analyzer. To determine whether the second channel is installed, press the Measurement Data hardkey and see if the softkeys have ch1/ch2 designations.

Note When the analyzer is configured such that it measures signals from a downconverter (like the RF section or the Agilent 89411A; when ROUTe:RECeiver is RF1, RF2, or EXT), autoranging is not available. It is available when the measured signal is connected directly to the IF input.

[SENSe:]VOLTage[1 | 2][:DC]:RANGe:AUTO:DIRection **command/query**

Sets the direction in which the range setting may be changed when autoranging is active.

Command Syntax: `[SENSe:]VOLTage[1|2][:DC]:RANGe:AUTO:DIRection UP|EITHer`

Example Statements:
OUTPUT 719;"SENSE:VOLT:DC:RANGE:AUTO:DIRECTION EITHER"
OUTPUT 719;"volt2:rang:auto:dir EITHER"

Query Syntax: `[SENSe:]VOLTage[1|2][:DC]:RANGe:AUTO:DIRection?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  EITH
SCPI Compliance:  confirmed
```

Description:

By default, autoranging is off.

UP sets the direction of autoranging value changes to be larger (up) but not smaller (down). This mode is better than EITHer for setting the range to measure transient signals because it avoids the measurement interruptions caused by frequent autoranging but may not have the optimum range setting for best signal-to-noise ratio.

EITH sets the direction of autoranging value changes to be either up or down. This mode tracks the signal amplitude, if it changes slowly, and optimizes the range for best signal-to-noise ratio. If the signal of interest is transient, the EITHer setting may spend too much time selecting a new range instead of making measurements.

Note When the analyzer is configured such that it measures signals from a downconverter (like the RF section or the Agilent 89411A; when ROUTe:RECeiver is RF1, RF2, or EXT), autoranging is not available. It is available when the measured signal is connected directly to the IF input labeled Channel 1.

Related commands:

- To execute one autorange, send SENS:VOLT:RANG:AUTO ONCE.

The voltage node numbers (1|2) specify which input (receiver) channel is affected by the command. In a single-channel 89441A, the channel specifier does nothing. If no number is specified, the command is applied to channel 1. Note that channel 2 is optional (AY7) and may not be installed in the analyzer. To determine whether the second channel is installed, press the Measurement Data hardkey and see if the softkeys have ch1/ch2 designations.

[SENSe:]VOLTage[1 | 2][:DC]:RANGe:UNIT:VOLTage command/query

Sets or queries the input range unit.

Command Syntax: `[SENSe:]VOLTage[1|2][:DC]:RANGe:UNIT:VOLTage <param>`

<param> ::= dBm|dBV|dBVrms|dBVpk|V|Vpk|Vrms|W|Wrms

Example Statements:
OUTPUT 719;":Voltage:Rang:Unit:Voltage W"
OUTPUT 719;"SENS:VOLTAGE:DC:RANG:UNIT:VOLT dBVpk"

Query Syntax: `[SENSe:]VOLTage[1|2][:DC]:RANGe:UNIT:VOLTage?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  DBM
SCPI Compliance:  confirmed
```

[SENSe:]VOLTage[1 | 2][:DC]:RANGe[:UPPer] command/query

Specifies the upper limit of the analyzer input's sensitivity range.

Command Syntax: `[SENSe:]VOLTage[1|2][:DC]:RANGe[:UPPer] <param>`

<param> ::= <number>[<unit>]|<step>|<bound>

<number> ::= a real number (NRf data)
limits: (see table)

<unit> ::= dBm|dBV|dBVrms|dBVpk|V|Vpk|Vrms|W|Wrms

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"sense:volt:dc:range:upp 0"
OUTPUT 719;"Voltage:Rang 0"

Query Syntax: `[SENSe:]VOLTage[1|2][:DC]:RANGe[:UPPer]?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 dBm
SCPI Compliance:  confirmed
```

Description:

The range setting determines the maximum signal level that can be applied to the analyzer's input connector without exceeding the largest value the analog-to-digital converter is configured to process. Executing this command turns autoranging OFF for the selected channel. The specified value is rounded up to the next allowed value.

Range Value Limits (50Ω)

	89410 or 89441 IF	89441 RF
value limits	–30 dBm: +24 dBm	–50 dBm: +25 dBm

IF indicates ROUT:REC IF or RF1 is active. RF indicates ROUT:REC RF2 is active.

Note By default, the vertical scale reference level tracks the range setting.

Related commands:

- To set the reference level, use DISP:WIND:TRAC:Y:RLEV.
- To turn autoranging on, use VOLT:RANGE:AUTO ON.
- To specify autoranging direction, use VOLT:RANGE:AUTO:DIR.

The voltage node numbers (1|2) specify which input (receiver) channel is affected by the command. In a single-channel 89441A, the channel specifier does nothing. If no number is specified, the command is applied to channel 1. Note that channel 2 is optional (AY7) and may not be installed in the analyzer. To determine whether the second channel is installed, press the Measurement Data hardkey and see if the softkeys have ch1/ch2 designations.

[SENSe:]VOLTage[1 | 2]:PROTection:CLEar command

Resets the analyzer's input-protection relay.

Command Syntax: `[SENSe:]VOLTage[1|2]:PROTection:CLEar`

Example Statements:
OUTPUT 719;":VOLT:PROTECTION:CLE"
OUTPUT 719;"sense:volt:prot:clear"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The input-protection relay is tripped (opened) when the signal at the input connector is significantly above the maximum input range. Bits 4 and 5 of the Questionable Voltage condition register indicate a tripped condition for input channels 1 and 2, respectively.

If the analyzer has two channels, this command resets both.

[SENSe:]WINDow:GATE command/query

Selects the FFT window type for gated data.

Command Syntax: `[SENSe:]WINDow:GATE UNIForm|FLATtop|HANNing|GTOP`

Example Statements:
OUTPUT 719;"Sens:Window:Gate FLATTOP"
OUTPUT 719;"WIND:GATE UNIFORM"

Query Syntax: `[SENSe:]WINDow:GATE?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  FLAT
SCPI Compliance:  instrument-specific
```

Description:

UNIF specifies the uniform window for the active input channels. The window's rectangular shape does not attenuate any portion of the time record. The uniform window is generally used with self-windowing functions such as burst and periodic chirps.

FLAT specifies the flat top window for the active input channels. The flat top window offers greater amplitude accuracy but lower frequency resolution than the Hanning window. It is generally used when a component's amplitude must be measured accurately, such as when using a fixed sine stimulus.

HANN specifies the Hanning window for the active input channels. The Hanning window offers higher frequency resolution but lower amplitude accuracy than the flat top window. It is the most commonly used window and is usually applied in random noise measurements.

GTOP specifies the gaussian top window for the active input channels. The GTOP window offers less amplitude accuracy, slightly higher frequency resolution, and much lower sidelobes than the Hanning window. It is generally used when a component's amplitude must be measured accurately, such as when using a fixed sine stimulus.

[SENSe:]WINDow:GATE:COUPling command/query

Specifies whether or not the gated-time window type tracks the main-time window type.

Command Syntax: `[SENSe:]WINDow:GATE:COUPling OFF|0|ON|1`

Example Statements:
OUTPUT 719;":wind:gate:coup ON"
OUTPUT 719;"Sens:Window:Gate:Coupling ON"

Query Syntax: `[SENSe:]WINDow:GATE:COUPling?`

Return Format: `CHAR`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  1 (ON)
SCPI Compliance:  instrument-specific
```

[SENSe:]WINDow[:TYPE] command/query

Selects the FFT window type.

Command Syntax: `[SENSe:]WINDow[:TYPE] UNIForm|FLATtop|HANNing|GTOP`

Example Statements:
OUTPUT 719;"SENS:WIND:TYPE UNIFORM"
OUTPUT 719;"wind GTOP"

Query Syntax: `[SENSe:]WINDow[:TYPE]?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  FLAT
SCPI Compliance:  instrument-specific
```

Description:

UNIF specifies the uniform window for the active input channels. The window's rectangular shape does not attenuate any portion of the time record. The uniform window is generally used with self-windowing functions such as burst and periodic chirps.

FLAT specifies the flat top window for the active input channels. The flat top window offers greater amplitude accuracy but lower frequency resolution than the Hanning window. It is generally used when a component's amplitude must be measured accurately, such as when using a fixed sine stimulus.

HANN specifies the Hanning window for the active input channels. The Hanning window offers higher frequency resolution but lower amplitude accuracy than the flat top window. It is the most commonly used window and is usually applied in random noise measurements.

GTOP specifies the gaussian top window for the active input channels. The GTOP window offers less amplitude accuracy, slightly higher frequency resolution, and much lower sidelobes than the Hanning window. It is generally used when a component's amplitude must be measured accurately, such as when using a fixed sine stimulus.

SOURce:FREQuency[:CW] command/query

Specifies the frequency for the fixed sine source type.

Command Syntax: `SOURce:FREQuency[:CW] <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 0:10e+6 (see restriction in description)
<unit>	::=	HZ
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"SOUR:FREQ 5 MHz"
OUTPUT 719;"Sour:Freq 1e6"
```

Query Syntax: `SOURce:FREQuency[:CW]?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  1 MHz (see restriction in description)
SCPI Compliance:  instrument-specific
```

Description:

For the 89441A, the source fixed-sine frequency may be specified only when ROUT:REC is IF, RF1, or COMB (I+jQ).

Related Commands

- The fixed-sine (or CW) source type is specified with the SOUR:FUNC SIN command.
- The source signal level is specified with the SOUR:VOLT command.
- Entries are accepted when the source is OFF or the source type is *not* fixed sine.

SOURce:FREQuency:OFFSet command/query

Specifies the offset frequency (from IF center) for the fixed sine source type.

Command Syntax: `SOURce:FREQuency:OFFSet <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -3.5 MHz:3.5 MHz
<unit>	::=	HZ
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"sour:freq:offset 0"
OUTPUT 719;"Sour:Frequency:Offs 0"
```

Query Syntax: `SOURce:FREQuency:OFFSet?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0 Hz
SCPI Compliance:  confirmed
```

Description:

In RF operation (when ROUT:REC is RF2), the source center frequency tracks the receiver frequency during the sweep. The offset frequency allows the source to track the receiver sweep with a constant frequency difference.

- This command is not available for the 89410A.
- For the 89441A, this command is available only when ROUT:REC is RF2.

SOURce:FUNCtion[:SHAPe] command/query

Selects the source output signal type.

Command Syntax: `SOURce:FUNCtion[:SHAPe] SINusoid|USER|RANDom|PCHirp`

Example Statements:
OUTPUT 719;":SOUR:FUNCTION USER"
OUTPUT 719;"sour:function:shap SINUSOID"

Query Syntax: `SOURce:FUNCtion[:SHAPe]?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  SIN
SCPI Compliance:  confirmed
```

Description:

After preset and when power is first turned on, the source is off.

- The OUTP 1 command may be used to turn it on.
- The amplitude value is specified with the SOUR:VOLT command.

Arguments used with the SOUR:FUNC command are as follows:

SIN selects fixed sine as the source output type. The signal is a continuous sine wave output at the center frequency of the current measurement or sub-measurement. The frequency of this signal is specified with the SOUR:FREQ command.

RAND selects random noise (with a Gaussian distribution) as the source output type. This source type's peak-to-rms ratio is not as low as the chirp signal, but the energy at any of the tones is not concentrated at a particular point in time, making it useful for measuring some dispersive networks.

- Random noise yields a fast, linear estimate of the system under test. Because it is not periodic in the time record, random noise requires windowing; usually the Hanning window.
- The bandwidth of the random noise is set so that most of the energy in the source signal is within the measured span.

PCH selects periodic chirp as the source output type. The signal has a low peak-to-rms ratio, useful for network measurements.

USER selects a user-defined source signal stored in one of the six data registers, D1–D6. The particular data register is selected with the SOUR:FUNC:USER:FEED command. The data must be in SDF format. Data for arbitrary source signals may have been time-domain traces that were saved earlier, created with commercial software such as PC Matlab, or defined with a BASIC program.

SOURce:FUNCtion:USER:FEED command/query

Selects one of six data registers containing user-defined source signals.

Command Syntax: `SOURce:FUNCtion:USER:FEED <string>`

<string> ::= ‘D1’|’D2’|’D3’|’D4’|’D5’|’D6’

Example Statements:
OUTPUT 719;"SOUR:FUNCTION:USER:FEED ‘D3’"
OUTPUT 719;"source:func:user:feed ‘D1’"

Query Syntax: `SOURce:FUNCtion:USER:FEED?`

Return Format: `STRING`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  D1
SCPI Compliance:  instrument-specific
```

Description:

This user-defined signal is used as the source output when the selected source type is USER (SOUR:FUNC USER).

SOURce:IFINput:STATe **command/query**

Selects a stimulus for the analyzer's RF source.

Command Syntax: `SOURce:IFINput:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"Sour:Ifinput:Stat ON"
OUTPUT 719;"SOURCE:IFIN:STAT ON"

Query Syntax: `SOURce:IFINput:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Option:  AY8 (Internal RF Source)
Synchronization Required:  no
Preset State:  OFF
SCPI Compliance:  instrument-specific
```

Description:

This command is applicable only in Agilent 89441A analyzers that have the optional RF source. Send SOURCE:IFINPUT:STATE:ON to drive the analyzer's RF source with an external signal. Send SOURCE:IFINPUT:STATE:OFF (default) to drive the RF source with the source signal from the analyzer's IF Section.

For additional details, see online help for the [ext IF input on/off] softkey. To display online help for this softkey, press [Help], press 1 to display the online help index, then page through the index and select this topic: ''Source, external signals, using as input to source''.

SOURce:RF

command/query

Switches the source output range between 0–10 MHz and RF (GHz)./

Command Syntax: `SOURce:RF OFF|0|ON|1`

Example Statements:
OUTPUT 719;":source:rf OFF"
OUTPUT 719;"Source:Rf ON"

Query Syntax: `SOURce:RF?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  ON
SCPI Compliance:  instrument-specific
```

Description:

This command is available only for the 89441A and only if the optional RF source (AY8) is installed and ROUT:REC is RF2.

ON specifies the RF source (2–2646 MHz for the 89441A).

OFF specifies the 0–10 MHz source.

SOURce:USER:REPeat **command/query**

Specifies whether the arbitrary and chirp source signals occur once (data record played once) or repeatedly.

Command Syntax: `SOURce:USER:REPeat OFF|0|ON|1`

Example Statements:
OUTPUT 719;"SOUR:USER:REP ON"
OUTPUT 719;"source:user:rep ON"

Query Syntax: `SOURce:USER:REPeat?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  1
SCPI Compliance:  instrument-specific
```

Description:

This function (repeating source) is available only when the source type is periodic chirp or arbitrary, the trigger type is not free run, and the source is on. See help text for information.

When repeat is on, the analyzer outputs data to the source connector continuously, without interruption.

When repeat is off, the source behavior is affected by trigger mode. In free run trigger, source output is continuous, just as it is with repeat on. For any other trigger mode, the source begins its output only when a trigger occurs and shuts off after all the data in the register has been output. This happens each time a trigger occurs.

Related Commands

- To specify the source type, use SOUR:FUNC.
- To specify the trigger type, use TRIG:SOUR.
- To turn the source on/off, use OUTP[:STAT].

SOURce:VOLTage[:LEVel][:IMMediate][:AMPLitude] command/query

Specifies the source output level.

Command Syntax: `SOURce:VOLTage[:LEVel][:IMMediate][:AMPLitude] <param>`

<param> ::= <number>[<unit>]|<step>|<bound>

<number> ::= a real number (NRf data)
limits: (see table)

<unit> ::= dBm|dBV|dBVrms|dBVpk|V|Vpk|Vrms|W|Wrms

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;"SOUR:VOLTAGE:LEV:IMM:AMPLITUDE -10DBM"
OUTPUT 719;"sour:volt .1V"

Query Syntax: `SOURce:VOLTage[:LEVel][:IMMediate][:AMPLitude]?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  -10 dBm (100 mV)
SCPI Compliance:  confirmed
```

Description:

Specifies a source output level that applies to all waveform types. For random noise or periodic chirp, the level is the total wideband level (the summation of the waveform measured at full span). For smaller spans, not all of the energy appears in the measurement because some of the waveform's power is outside the selected span. If the level is specified for random noise in Vrms, when the source type is changed, the analyzer maintains the rms level for the new type. The same is true for Vpk. Because the ratio of peak/rms varies for different signal types, the peak value changes when the analyzer maintains the rms value. If a new rms value would require a peak value outside the limits of the source voltage, the level is set to the limit. Limits are ±5 Vpk and include offset as well as level. If no units are entered with the number, dBm are assumed.

Source Output Level Limits

	89410 or 89441 SOUR:RF 0	89441 SOUR:RF 1
value limits	–110 dBm:+24 dBm 0 V_{pk}:5.013 V_{pk}	–40 dBm:+23 dBm (see following text)

For the Agilent 89441A, you can set the source output level as high as +23 dBm. However, the output level for the Agilent 89441A is specified only to +13 dBm. Source performance is not specified beyond +13 dBm. For additional details, see the specifications shipped with the Agilent 89441A *Installation & Verification Guide*.

Notes

- It is good practice to specify the source level *before* the source is turned on. When the source is turned on, the output voltage level is whatever value was active when it was turned off. The active value may be large enough to damage sensitive devices. At turn-on and preset, the level is –10 dBm (100 mV).
- Changing instrument mode does not turn the source off nor change the source output value.

SOURce:VOLTage[:LEVel][:IMMediate]:OFFSet command/query

Specifies the source output dc offset in volts.

Command Syntax: `SOURce:VOLTage[:LEVel][:IMMediate]:OFFSet <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: -3.42:+3.42
<unit>	::=	V
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:
OUTPUT 719;":Source:Volt:Offset 0"
OUTPUT 719;"SOUR:VOLT:LEVEL:IMM:OFFSET 0"

Query Syntax: `SOURce:VOLTage[:LEVel][:IMMediate]:OFFSet?`

Return Format: `Real`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0V
SCPI Compliance:  confirmed
```

Description:

The maximum value for the source output is ±5 Vpk. This value is a sum of the dc offset and peak value of the source amplitude. So the limits of the source offset are dependent on the source level.

Step size (used with UP|DOWN) varies in a 1, 2, 5 pattern starting with 100 uVpk.

SOURce:VOLTage[:LEVel]:UNIT:VOLTage command/query

Sets or queries the source level unit.

Command Syntax: `SOURce:VOLTage[:LEVel]:UNIT:VOLTage <param>`

<param> ::= dBm|dBV|dBVrms|dBVpk|V|Vpk|Vrms|W|Wrms

Example Statements:
OUTPUT 719;"sour:volt:level:unit:voltage Wrms"
OUTPUT 719;"Sour:Volt:Unit:Volt Vrms"

Query Syntax: `SOURce:VOLTage[:LEVel]:UNIT:VOLTage?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  DBM
SCPI Compliance:  confirmed
```

SOURce:VOLTage:PROTection:CLEar command

Clears the source output protection circuitry.

Command Syntax: `SOURce:VOLTage:PROTection:CLEar`

Example Statements:

```
OUTPUT 719;":SOURCE:VOLT:PROT:CLEAR"
OUTPUT 719;"sour:voltage:prot:cle"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The source-protection relay opens when approximately ±13 volts or greater is applied to the SOURCE BNC connector. A message on the screen tells you when the relay has been tripped.

To detect a source-tripped condition, set the status registers to report a source trip in the questionable voltage register set.

STATus:DEVice:CONDition? query

Reads and clears the Device State Condition register.

Query Syntax: `STATus:DEVice:CONDition?`

Example Statements:
```
OUTPUT 719;"Status:Dev:Condition?"
OUTPUT 719;"STAT:DEV:CONDITION?"
```

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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'' for a definition of bits in the register set. See ''General Status Register Model'' 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>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements: OUTPUT 719;"STAT:DEVICE:ENAB 1024"
OUTPUT 719;"stat:device:enab 768"

Query Syntax: `STATus:DEVice:ENABle?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ''Device State Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' 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 719;":stat:device?"
OUTPUT 719;"Stat:Dev:Event?"
```

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  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" for a definition of bits in the register set. See "General Status Register Model" 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>`

<number> ::= a real number (NRf data)
limtis: 0:32767

Example Statements:
OUTPUT 719;":STAT:DEV:NRTRANSITION 67"
OUTPUT 719;"stat:device:ptr 4098"

Query Syntax: `STATus:DEVice:NTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ‘‘Device Status Register Set’’ for a definition of bits in the register set. See ‘‘General Status Register Model’’ 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>`

<number> ::= a real number (NRf data)
limtis: 0:32767

Example Statements:
OUTPUT 719;":STAT:DEV:PRTRANSITION 513"
OUTPUT 719;"stat:device:ptr 1025"

Query Syntax: `STATus:DEVice:PTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ''Device State Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' 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 719;"STAT:OPERATION:COND?"
OUTPUT 719;"stat:operation:cond?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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'' for a definition of bits in the register set. See ''General Status Register Model'' 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 719;":Status:Oper:Enab 9488"
OUTPUT 719;"STATUS:OPER:ENABLE 32231"
```

Query Syntax: `STATus:OPERation:ENABle?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  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'' for a definition of bits in the register set. See ''General Status Register Model'' 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 719;"stat:oper:event?"
OUTPUT 719;"Stat:Operation?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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'' for a definition of bits in the register set. See ''General Status Register Model'' 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>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;":Stat:Operation:Ntr 5"
OUTPUT 719;"STAT:OPERATION:NTR 53"

Query Syntax: `STATus:OPERation:NTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ''Operational Status Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' 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>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;"status:oper:ptr 5"
OUTPUT 719;"Status:Oper:Ptransition 55"

Query Syntax: `STATus:OPERation:PTRansition?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ''Operational Status Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' 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 719;":STAT:PRES"
OUTPUT 719;"status:pres"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

STAT:PRESet effects on the Questionable Voltage and Questionable Frequency register set as followss:

- 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 brings 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:PRESet 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 719;"Status:Ques:Cond?"
OUTPUT 719;"STATUS:QUES:CONDITION?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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'' for a definition of bits in the register set. See ''General Status Register Model'' 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:QUEStional:ENABle <number>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;":stat:ques:enable 28055"
OUTPUT 719;"Stat:Questionable:Enab 2478"

Query Syntax: `STATus:QUEStionable:ENABle?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ''Questionable Status Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' 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 719;"STAT:QUESTIONABLE:EVEN?"
OUTPUT 719;"status:ques?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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’’ for a definition of bits in the register set. See ‘‘General Status Register Model’’ for information about the role of event registers in register sets.

STATus:QUEStionable:FREQuency:CONDition? query

Reads and clears the Questionable Frequency Event register.

Query Syntax: `STATus:QUEStionable:FREQuency:CONDition?`

Example Statements:

```
OUTPUT 719;":Stat:Questionable:Freq:Condition?"
OUTPUT 719;"STAT:QUES:FREQUENCY:COND?"
```

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Note The Questionable Frequency Condition register is automatically cleared after it is read by this query.

See ''Questionable Frequency Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of event registers in register sets.

STATus:QUEStionable:FREQuency:ENABle — command/query

Sets and queries bits in the Questionable Frequency Enable register.

Command Syntax: `STATus:QUEStionable:FREQuency:ENABle <number>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:

```
OUTPUT 719;":Stat:Ques:Frequency:Enab 27"
OUTPUT 719;"STATUS:QUES:FREQ:ENABLE 168"
```

Query Syntax: `STATus:QUEStionable:FREQuency:ENABle?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Frequency 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 bit setting is *not* modified by the *RST command.

See ''Questionable Frequency Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of enable registers in register sets.

STATus:QUEStionable:FREQuency[:EVENt]? query

Reads and clears the Questionable Frequency Event register.

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

Example Statements:
OUTPUT 719;"status:ques:freq:event?"
OUTPUT 719;"Stat:Questionable:Freq?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 Frequency Event register. (The decimal weight of a bit is 2^n, where n is the bit number.)

Note The Questionable Frequency Event register is automatically cleared after it is read by this query.

See ''Questionable Frequency Register Set'' definition of bits in the register set. See ''General Status Register Model'' for information about the role of event registers in register sets.

STATus:QUEStionable:FREQuency:NTRansition command/query

Sets and queries bits in the Questionable Frequency Negative Transition register.

Command Syntax: `STATus:QUEStionable:FREQuency:NTRansition <number>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;":Status Ques:Frequency:Ntr 978"
OUTPUT 719;"STAT:QUESTIONABLE:FREQ:NTRANSITION 63"

Query Syntax: `STATus:QUEStionable:FREQuency:NTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Frequency 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 bit setting is *not* modified by the *RST command.

See ''Questionable Frequency Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of negative transition registers in register sets.

STATus:QUEStionable:FREQuency:PTRansition — command/query

Sets bits in the Questionable Frequency Positive Transition register.

Command Syntax: `STATus:QUEStionable:FREQuency:PTRansition <number>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;"Stat:Questionable:Freq:Ptr 512"
OUTPUT 719;"STATUS:QUES:FREQUENCY:PTR 677"

Query Syntax: `STATus:QUEStionable:FREQuency:PTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Frequency 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 bit setting is *not* modified by the *RST command.

See ''Questionable Frequency Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of positive transition registers in register sets.

STATus:QUEStionable:MODulation:CONDition? query

Reads and clears the Questionable Modulation Event register.

Query Syntax: `STATus:QUEStionable:MODulation:CONDition?`

Example Statements:

```
OUTPUT 719;":STAT:QUESTIONABLE:MOD:CONDITION?"
OUTPUT 719;"stat:ques:modulation:cond?"
```

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Note The Questionable Modulation Condition register is automatically cleared after it is read by this query.

See ‘‘Questionable Modulation Register Set’’ for a definition of bits in the register set. See ‘‘General Status Register Model’’ for information about the role of event registers in register sets.

STATus:QUEStionable:MODulation:ENABle — command/query

Sets and queries bits in the Questionable Modulation Enable register.

Command Syntax: `STATus:QUEStionable:MODulation:ENABle <number>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;":Stat:Ques:Modulation:Enab 1"
OUTPUT 719;"STATUS:QUES:MOD:ENABLE 3"

Query Syntax: `STATus:QUEStionable:MODulation:ENABle?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Modulation 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 bit setting is *not* modified by the *RST command.

See ''Questionable Modulation Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of enable registers in register sets.

STATus:QUEStionable:MODulation[:EVENt]? query

Reads and clears the Questionable Modulation Event register.

Query Syntax: `STATus:QUEStionable:MODulation[:EVENt]?`

Example Statements:
OUTPUT 719;"Status:Ques:Mod:Event?"
OUTPUT 719;"STAT:QUESTIONABLE:MOD?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 Modulation Event register. (The decimal weight of a bit is 2^n, where n is the bit number.)

Note The Questionable Modulation Event register is automatically cleared after it is read by this query.

See ‘‘Questionable Modulation Register Set’’ definition of bits in the register set. See ‘‘General Status Register Model’’ for information about the role of event registers in register sets.

STATus:QUEStionable:MODulation:NTRansition command/query

Sets and queries bits in the Questionable Modulation Negative Transition register.

Command Syntax: `STATus:QUEStionable:MODulation:NTRansition <number>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;":Status Ques:Modulation:Ntr 3"
OUTPUT 719;"STAT:QUESTIONABLE:MOD:NTRANSITION 2"

Query Syntax: `STATus:QUEStionable:MODulation:NTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Modulation 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 bit setting is *not* modified by the *RST command.

See ''Questionable Modulation Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of negative transition registers in register sets.

STATus:QUEStionable:MODulation:PTRansition command/query

Sets and queries bits in the Questionable Modulation Positive Transition register.

Command Syntax: `STATus:QUEStionable:MODulation:PTRansition <number>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements: OUTPUT 719;"Stat:Questionable:Mod:Ptr 1"
OUTPUT 719;"STATUS:QUES:MODULATION:PTR 3"

Query Syntax: `STATus:QUEStionable:MODulation:PTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

To set a single bit in the Questionable Modulation 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 bit setting is *not* modified by the *RST command.

See ''Questionable Modulation Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' 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>`

<number> ::= a real number (NRf)
limits: 0:32767

Example Statements:
OUTPUT 719;":Stat:Questionable:Ntr 5"
OUTPUT 719;"STATUS:QUES:NTR 54"

Query Syntax: `STATus:QUEStionable:NTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See "Questionable Status Register Set" for a definition of bits in the register set. See "General Status Register Model" 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>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;"status:ques:ptransition 5"
OUTPUT 719;"Stat:Ques:Ptransition 53"

Query Syntax: `STATus:QUEStionable:PTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ''Questionable Status Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of positive transition registers in register sets.

STATus:QUEStionable:VOLTage:CONDition? query

Reads and clears the Questionable Voltage Event register.

Query Syntax: `STATus:QUEStionable:VOLTage:CONDition?`

Example Statements:
OUTPUT 719;":stat:questionable:volt:condition?"
OUTPUT 719;"Stat:Ques:Voltage:Cond?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Note The Questionable Voltage Condition register is automatically cleared after it is read by this query.

See ‘‘Questionable Voltage Register Set’’ for a definition of bits in the register set. See ‘‘General Status Register Model’’ for information about the role of event 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>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;"Stat:Questionable:Volt:Enable 5"
OUTPUT 719;"STAT:QUES:VOLTAGE:ENAB 23"

Query Syntax: `STATus:QUEStionable:VOLTage:ENABle?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ''Questionable Voltage Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' 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 719;"STATUS:QUES:VOLT:EVENT?"
OUTPUT 719;"stat:questionable:volt?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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'' for a definition of bits in the register set. See ''General Status Register Model'' 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:QEUStionable:VOLTage:NTRansition <number>`

<number> ::= a real number (NRf)
limits: 0:32767

Example Statements:
OUTPUT 719;"STAT:QUESTIONABLE:VOLT:NTRANSITION 7"
OUTPUT 719;"stat:ques:voltage:ntr 37"

Query Syntax: `STATus:QUEStionable:VOLTage:NTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ''Questionable Voltage Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' 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:QEUStionable:VOLTage:PTRansition <number>`

<number> ::= a real number (NRf)
limits: 0:32767

Example Statements:
OUTPUT 719;"STAT:QUESTIONABLE:VOLT:PTRANSITION 4"
OUTPUT 719;"stat:ques:voltage:ptr 33"

Query Syntax: `STATus:QUEStionable:VOLTage:PTRansition?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 bit setting is *not* modified by the *RST command.

See ''Questionable Voltage Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of positive transition registers in register sets.

STATus:USER:ENABle — command/query

Sets and queries bits in the User Defined Enable register.

Command Syntax: `STATus:USER:ENABle <number>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:

```
OUTPUT 719;"status:user:enable 127"
OUTPUT 719;"Stat:User:Enable 511"
```

Query Syntax: `STATus:USER:ENABle?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

To set a single bit in the User Defined 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 bit setting is *not* modified by the *RST command.

See ''User Defined Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of enable registers in register sets.

STATus:USER[:EVENt]? query

Reads and clears the User Defined Event register.

Query Syntax: `STATus:USER[:EVENt]?`

Example Statements:
OUTPUT 719;":Stat:User?"
OUTPUT 719;"STAT:USER:EVEN?"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  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 Defined Event register. (The decimal weight of a bit is 2^n, where n is the bit number.)

Note The User Defined Event register is automatically cleared after it is read by this query.

See ''User Defined Register Set'' for a definition of bits in the register set. See ''General Status Register Model'' for information about the role of event registers in register sets.

STATus:USER:PULSe command

Sets bits in the User Defined Event register.

Command Syntax: `STATus:USER:PULSe <number>`

<number> ::= a real number (NRf data)
limits: 0:32767

Example Statements:
OUTPUT 719;"Stat:User:Puls 4096"
OUTPUT 719;"STAT:USER:PULS 32"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Each bit in the User Defined 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'' for more information.

SYSTem:BEEPer:STATe — command/query

Turns the analyzer's beeper on or off.

Command Syntax: `SYSTem:BEEPer:STATe OFF|0|ON|1`

Example Statements:
OUTPUT 719;"syst:beeper:stat ON"
OUTPUT 719;"System:Beep:Stat OFF"

Query Syntax: `SYSTem:BEEPer:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The beeper (audible alarm) sounds when error messages, warning messages, or other (informative) messages are displayed.

The beeper state is saved in non-volatile RAM; it is not affected by preset.

SYSTem:COMMunicate[1|2]:GPIB:ADDRess command/query

Sets the GP-IB address of the analyzer.

Command Syntax: `SYSTem:COMMunicate[1|2]:GPIB:ADDRess <number>|<step>|<bound>`

<number> ::= a real number (NRf data)
limits: 0:33

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":SYSTEM:COMM2:GPIB:ADDR 24"
OUTPUT 719;"syst:communicate:gpib:address 4"

Query Syntax: `SYSTem:COMMunicate[1|2]:GPIB:ADDRess?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

The address is set to 19 at the factory.

The [1|2] in the communicate node has no effect for this command.

There is another board in the backpanel that has 20 MB of RAM, 2 LAN connectors, and an GPIB connector on it. The GPIB connector on this board is called the *System Interconnect* and is used exclusively to control other instruments that serve as a downconverter. When the analyzer instructs another instrument (connected to the System Interconnect) to talk, it also addresses itself to listen and the address used is 1 plus the value of the address set for the main (first) GPIB connector with this command.

For example, sending SYST:COMM:GPIB:ADDR 17 sets the System Inteconnect address to 18.

Related Commands:

- To set the address of the external analyzer, use [SENS:]FREQ:EXT:COMM:ADDR.
- To enable/disable remote control of an external analyzer, use [SENS:]FREQ:EXT:COMM.

SYSTem:COMMunicate[1|2]:LAN:EADDress? query

Queries the instrument ethernet address.

Query Syntax: `SYSTem:COMMunicate[1|2]:LAN:EADDress?`

Example Statements:
OUTPUT 719;"Syst:Comm2:Lan:Eadd?"
OUTPUT 719;"SYSTEM:COMM:LAN:EADDRESS?"

Return Format: `STRING`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This parameter represents the Ethernet Station Address assigned to the particular network interface being used in the instrument.

Note This number is unique for each hardware interface and is assigned by the factory. This value can only be set by qualified factory personnel.

The [1|2] in the communicate node has no effect for this command.

SYSTem:COMMunicate[1 | 2]:LAN:IPADdress command/query

Sets the device Internet Protocol address.

Command Syntax: `SYSTem:COMMunicate[1|2]:LAN:IPADdress <param>`

<param> ::= (dotted-decimal IP address in STRING format)

Example Statements: OUTPUT 719;"System:Comm:Lan:Ipaddress ‘15.1.221.235’"

Query Syntax: `SYSTem:COMMunicate[1|2]:LAN:IPADdress?`

Return Format: `STRING`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This parameter determines the device Internet Protocol address used for communication between network nodes. The IP address is entered and displayed in dotted-decimal notation. For example, 15.1.221.235. The assignment of an IP address is done by your local network administrator.

Notes

Changing this parmeter requires cycling power on the analyzer.
Changing this paramater while open TCP/IP connections exist will cause the open connections to hang.

The [1|2] in the communicate node has no effect for this command.

SYSTem:COMMunicate[1|2]:LAN:PORT **command/query**

Specifies which of the two LAN connectors to use.

Command Syntax: `SYSTem:COMMunicate[1|2]:LAN:PORT BNC|AUI`

Example Statements:
OUTPUT 719;":syst:communicate:lan:port BNC"
OUTPUT 719;"System:Comm:Lan:Port BNC"

Query Syntax: `SYSTem:COMMunicate[1|2]:LAN:PORT?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

Only one port should have a network connected to it. When the BNC (thinLAN) port is selected, any MAUs plugged into the AUI port should be removed.

Note If you are controlling the analyzer via the LAN interface, executing this command terminates the control link.

The [1|2] in the communicate node has no effect for this command.

SYSTem:COMMunicate[1|2]:LAN:ROUTe:GATeway command/query

Sets the IP address for a LAN gateway.

Command Syntax: `SYSTem:COMMunicate[1|2]:LAN:ROUTe:GATeway <param>`

<param> ::= (dotted-decimal IP address in STRING format)

Example Statements: OUTPUT 719;"SYST:COMM:LAN:ROUT:GATEWAY ‘15.1.221.235’"

Query Syntax: `SYSTem:COMMunicate[1|2]:LAN:ROUTe:GATeway?`

Return Format: `STRING`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This parameter determines the IP address of a gateway used for routing packets to destinations which are not on the local subnet. The subnet mask determines how this decision is made. This parameter is entered in dotted decimal notation. For example, 15.1.221.235. This value should be obtained from your local network administrator.

Note A value of zero disables gateway routing.

The [1|2] in the communicate node has no effect for this command.

SYSTem:COMMunicate[1|2]:LAN:ROUTe:SMASk command/query

Sets the subnet mask.

Command Syntax: `SYSTem:COMMunicate[1|2]:LAN:ROUTe:SMASk <param>`

<param> ::= (dotted-decimal mask in STRING format)

Example Statements: OUTPUT 719;"Syst:Comm:Lan:Rout:Smask '255.255.248.0'"

Query Syntax: `SYSTem:COMMunicate[1|2]:LAN:ROUTe:SMASk?`

Return Format: `STRING`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This parameter is the subnet mask. The subnet mask is used to determine if a packet needs to be routed to a gateway rather than directly to the receiver. You can disable use of the gateway routing by setting the network mask to zero. This parameter is entered in dotted decimal notation. For example, 255.255.248.0. This value should be obtained from your local network administrator.

The [1|2] in the communicate node has no effect for this command.

SYSTem:COMMunicate[1|2]:LAN:STATe command/query

Turns networking on/off.

Command Syntax: `SYSTem:COMMunicate[1|2]:LAN:STATe OFF|0|ON|1`

Example Statements:

OUTPUT 719;"SYST:COMMUNICATE:LAN:STATE ON"
OUTPUT 719;"syst:comm:lan:stat OFF"

Query Syntax: `SYSTem:COMMunicate[1|2]:LAN:STATe?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

When networking is off more memory may be available for other uses. This memory reconfiguration requires that the analyzer power switch be turned off and back on.

The [1|2] in the communicate node has no effect for this command.

SYSTem:COMMunicate[1|2]:LAN:XWINdow:HOSTname command/query

Specifies the IP address of an external X11 display server.

Command Syntax: `SYSTem:COMMunicate:LAN:XWINdow:HOSTname <param>`

<param> ::= (dotted-decimal IP address in STRING format)

Example Statements: OUTPUT 719;":SYST:COMMUNICATE:LAN:XWIN:HOSTNAME ‘15.1.221.235’"

Query Syntax: `SYSTem:COMMunicate[1|2]:LAN:XWINdow:HOSTname?`

Return Format: `STRING`

Attribute Summary:

```
Option:  UG7 (advanced LAN)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

This command must be issued before activating the X11 Window feature. The “hostname” is the host’s IP address entered in dotted-decimal notation. For example, 15.1.221.235.

The [1|2] in the communicate node has no effect for this command.

SYSTem:COMMunicate[1 | 2]:LAN:XWINdow:RATE command/query

Specifies the maximum update rate for the X11 window display.

Command Syntax: `SYSTem:COMMunicate[1|2]:LAN:XWINdow:Rate <param>`

<param> ::= <number>[<unit>]|<step>|<bound>

<number> ::= a real number (NRf data)
limits: 0:60

<unit> ::= [HZ]

<step> ::= UP|DOWN

<bound> ::= MAX|MIN

Example Statements:
OUTPUT 719;":System:Comm:Lan:Xwindow:Rate 47"
OUTPUT 719;"SYSTEM:COMM:LAN:XWINDOW:RATE 44"

Query Syntax: `SYSTem:COMMunicate[1|2]:LAN:XWINdow:RATE?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

Setting the value to 0 places no limit on the rate while setting it to a value places an upper limit on the update rate. On some computer systems this setting is important to limit the amount of LAN traffic being sent to the display for either network bandwidth considerations or for loading on the destination computer system.

The [1|2] in the communicate node has no effect for this command.

SYSTem:COMMunicate[1|2]:LAN:XWINdow[:STATe] command/query

Turns the X11 Window display capability on/off.

Command Syntax: `SYSTem:COMMunicate[1|2]:LAN:XWINdow[:STATe] OFF|0|ON|1`

Example Statements:

OUTPUT 719;"system:comm:lan:xwindow:stat OFF"
OUTPUT 719;"System:Comm:Lan:Xwindow OFF"

Query Syntax: `SYSTem:COMMunicate[1|2]:LAN:XWINdow[:STATe]?`

Return Format: `Integer`

Attribute Summary:

```
Option:  UG7 (advanced LAN)
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  instrument-specific
```

Description:

The host X11 server must be configured to allow the analyzer permission to open a window for this command to complete successfully. This is done with the `xhost` command for UNIX systems.

Preqequisites:

- The LAN must be active (ON) for the XWindows feature to work. See SYST:COMM:LAN:STATe.
- Before setting STATE to ON, the user should set the Display IP address with the SYSTEM:COMMUNICATE:LAN:XWINDOW:HOSTNAME command.

After this command is issued, no analyzer activity takes place until the X11 window on the workstation is positioned and placed with the mouse.

The [1|2] in the communicate node has no effect for this command.

SYSTem:COMMunicate[1|2]:SERial:CONTrol:DTR command/query

Specifies hardware handshake type for the serial ports.

Command Syntax: `SYSTem:COMMunicate[1|2]:SERial:CONTrol:DTR OFF|IBFULL`

Example Statements:
OUTPUT 719;"Syste,:Comm2:Ser:Control:Dtr OFF"
OUTPUT 719;"SYSTEM:COMM:SER:CONTROL:DTR IBF"

Query Syntax: `SYSTem:COMMunicate[1|2]:SERial:CONTrol:DTR?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

(DTR is a serial interface signal signifying data-terminal-ready.) When this command is set to IBFull, hardware pacing is used for the serial ports; the analyzer unasserts DTR if it is unable to receive data and it does not send data if the CTS line (clear-to-send) is not asserted.

When DTR is OFF, the analyzer uses the CTS line for transmit pacing and it leaves the DTR line unasserted.

The 1|2 in the communicate node selects the particular serial port to be configured. If no number is specified, 1 is assumed.

Note The 89441A uses the SERIAL 2 port to control the downconverter portion of the analyzer. While the analyzer is configured as an 89441A, none of the serial communication parameters may be changed for the SERIAL 2 port.

SYSTem:COMMunicate[1 | 2]:SERial[:RECeive]:BAUD command/query

Specifies the serial communication baud rate.

Command Syntax: `SYSTem:COMMunicate[1|2]:SERial[:RECeive]:BAUD <param>`

<param>	::=	<number>|<step>|<bound>
<number>	::=	a real number (NRf data) limits: 75:19200
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;":SYST:COMMUNICATE:SER:BAUD 16335"
OUTPUT 719;"system:comm:serial:rec:baud 8681"
```

Query Syntax: `SYSTem:COMMunicate[1|2]:SERial[:RECeive]:BAUD?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

Any number within the limits is accepted and rounded to one of the following: 75, 150, 300, 600, 1200, 2400, 4800, 9600, 19200.

SYSTem:COMMunicate[1 | 2]:SERial[:RECeive]:PACE **command/query**

Specifies software handshake type for the serial ports.

Command Syntax: `SYSTem:COMMunicate[1|2]:SERial[:RECeive]:PACE XON|NONE`

Example Statements:
OUTPUT 719;"System:Comm2:Serial:Rec:Pace NONE"
OUTPUT 719;"SYSTEM:COMM:SERIAL:PACE XON"

Query Syntax: `SYSTem:COMMunicate[1|2]:SERial[:RECeive]:PACE?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

The 1|2 in the communicate node selects the particular serial port to be configured. If no number is specified, 1 is assumed.

Note The 89441A uses the SERIAL 2 port to control the downconverter portion of the analyzer. While the analyzer is configured as an 89441A, none of the serial communication parameters may be changed for the SERIAL 2 port.

SYSTem:COMMunicate[1 | 2]:SERial[:RECeive]:PARity[:TYPE] command/query

Sets the parity type for the two serial interface ports.

Command Syntax: `SYSTem:COMMunicate[1|2]:SERial[:RECeive]:PARity[:TYPE] <param>`

<param> ::= EVEN|ODD|ZERO|ONE|NONE

Example Statements:
OUTPUT 719;":syst:communicate:ser:parity EVEN"
OUTPUT 719;"Syst:Comm:Serial:Rec:Parity:Type NONE"

Query Syntax: `SYSTem:COMMunicate[1|2]:SERial[:RECeive]:PARity[:TYPE]?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

The 1|2 in the communicate node selects the particular serial port to be configured. If no number is specified, 1 is assumed.

Note The 89441A uses the SERIAL 2 port to control the downconverter portion of the analyzer. While the analyzer is configured as an 89441A, none of the serial communication parameters may be changed for the SERIAL 2 port.

SYSTem:DATE **command/query**

Specify the year/month/day (for the date stamp?)

Command Syntax: `SYSTem:DATE <year>,<month>,<day>`

<year> ::= a real number (NRf data)
limits: 1991:2090

<month> ::= a real number (NRf data)
limits: 1:12

<day> ::= a real number (NRf data)
limits: 1:31

Example Statements:

OUTPUT 719;"Syst:Date 1991, 11, 26"
OUTPUT 719;"SYST:DATE 1993, 4, 1"

Query Syntax: `SYSTem:DATE?`

Return Format: `Integer, Integer, Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This information affects the filename dates in the disk directory operations. The date information is saved with a trace and may be examined with the SDF (standard data format) utilities. Also, date information is part of a plot or print time stamp.

All numbers are rounded to the nearest integer. Days of the months are checked for limits for the given month, including the number of days in February of a leap year.

SYSTem:ERRor? query

Returns one error message from the analyzer's error queue.

Query Syntax: `SYSTem:ERRor?`

Example Statements:
OUTPUT 719;"SYST:ERROR?"
OUTPUT 719;"syst:error?"

Return Format: `Integer, STRING`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

The error queue temporarily stores up to 20 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 20 error messages are reported before any are read from the queue, the oldest (first) error messages are saved. The last error message indicates more error messages were received than the queue could hold. If there are no errors in the queue, the query returns +0,"No error". For more information about error messages, see "Error Messages" in the appendix.

Note The error queue is cleared when the analyzer is turned on and when it receives the *CLS command.

SYSTem:GPIB:ECHO **command/query**

Turns the echo (display) of GPIB command mnemonics on/off.

Command Syntax: `SYSTem:GPIB:ECHO OFF|0|ON|1`

Example Statements:
OUTPUT 719;":Syst:Gpib:Echo OFF"
OUTPUT 719;"SYST:GPIB:ECHO OFF"

Query Syntax: `SYSTem:GPIB:ECHO?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

Commands are displayed in the upper left-hand corner of the analyzer's display screen.

SYSTem:KEY **command/query**

Writes or queries front-panel key presses.

Command Syntax: `SYSTem:KEY <number>`

<number> ::= a real number (NRf data)
limits: 0:511

Example Statements:

```
OUTPUT 719;"SYST:KEY 111"
OUTPUT 719;"syst:key 21"
```

Query Syntax: `SYSTem:KEY?`

Return Format: `Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

Sending the command with a keycode simulates pressing of that front-panel key. See table for the front-panel keycodes.

The query returns the keycode for the last key pressed. If no keys have been pressed since the last query, a –1 is returned.

Keys are encoded as follows:

- 0–127 are front-panel keys.
- 128–255 are shifted front-panel keys (never returned on query)
- 256–511 are interpreted as RPG (rotary pulse generator; knob) movement. The direction and amount of movement is determined by <query value> – 384. So a query value of 386 represents 2 RPG ticks and a query value of 382 represents –2 RPG clicks.

*RST clears the queue of keys.

Front Panel Keycodes

Key Name	Number	Key Name	Number	Key Name	Number
Measurement Data	0	Preset	25	Back Space	50
Norm	1	Disk Utility	26	F1 (softkey)	111
Ref Lvl/Scale	2	Plot/Print	27	F2 (softkey)	112
A	3	BASIC	28	F3 (softkey)	113
B	4	System Utilitl	29	F4 (softkey)	114
C	5	Local/Setup	30	F5 (softkey)	115
D	6	Marker	31	F6 (softkey)	116
Data Format	7	Marker	32	F7 (softkey)	117
Display	8	Marker Search	33	F8 (softkey)	118
Math	9	Marker Function	34	Return	119
Frequency	10	Marker/Entry	35	Shift[1]	128
Time	11	7	36	Mkr Val	130
Range	12	8	37	(Shift) A[2]	131
Input	13	9	38	(Shift) B[2]	132
ResBw/Window	14	< up arrow >	39	(Shift) C[2]	133
Source	15	4	40	(Shift) D[2]	134
Sweep	16	5	41	Mkr Peak	159
Instrument Mode	17	6	42	Zero Offset	160
Average	18	< down arrow >	43	Mkr CF	161
Trigger	19	1	44	Mkr Ref	162
Pause/Single	20	2	45	Mkr Val	163
Meas Restart	21	3	46	(Shift) < up arrow >[3]	167
Help	22	0 (zero)	47	(Shift) < down arrow >[4]	171
View State	23	• (decimal point)	48		
Save/Recall	24	+/-	49		

[1] Add 128 to base values of shifted keys to get shifted values.
[2] Select multiple active traces; usually, selecting a trace deactivates all others.
[3] Previous page (for editing BASIC programs)
[4] Next page (for editing BASIC programs)

SYSTem:KLOCk command/query

Keyboard lockout; disables the keyboard.

Command Syntax: `SYSTem:KLOCk OFF|0|ON|1`

Example Statements:
OUTPUT 719;"syst:klock OFF"
OUTPUT 719;"Syst:Klock OFF"

Query Syntax: `SYSTem:KLOCk?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  +0
SCPI Compliance:  confirmed
```

Description:

This command allows your controller to disable the keyboard. This affects the front-panel keys as well as the DIN/QWERTY remote keyboard and provides local lockout capability during the running of Instrument BASIC programs.

The query returns 1 if the keyboard is disabled.

Note At *RST, the value is +0 (OFF).

SYSTem:PRESet **command**

Sets most of the analyzer parameters to their default (power-on) state.

Command Syntax: `SYSTem:PRESet`

Example Statements:
OUTPUT 719;":SYST:PRES"
OUTPUT 719;"system:pres"

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

The default value for each parameter is listed in the Attribute Summary section of the command description page in the command reference.

This command is similiar to *RST. For differences, see *RST.

If a file named AUTO_ST containing an instrument state exists on any drive, that state is loaded and becomes active at power-on.

The following are *not* affected by the SYST:PRES command (or *RST):

- The state of the Power-on Status Clear flag.
- The state of all enabled and transition registers.
- The GPIB input and output queues.
- The time and date (SYST:TIME and SYST:DATE).
- The GPIB address settings.
  - SYST:COMM:GPIB:ADDR
  - HCOP:PLOT:ADDR
  - HCOP:PRIN:ADDR
- The memory configuration.
- The GPIB controller capability setting.
- The default disk selection (MMEM:MSIS).
- External disk address.
- External receiver address.
- Contents of data registers.
- Contents of math function and constant registers.
- Contents of the RAM and NVRAM disks.
- Calibration constants.
- LAN configuration.

SYSTem:TIME command/query

Specifies the system time.

Command Syntax: `SYSTem:TIME <hour>,<minute>,<second>`

<hour> ::= a real number (NRf data)
limits: 1:23

<minute> ::= a real number (NRf data)
limits: 0:59

<second> ::= a real number (NRF data)
limits: 0:59

Example Statements:

```
OUTPUT 719;"SYSTEM:TIME 15,55,0"
OUTPUT 719;"syst:time 8,30,20"
```

Query Syntax: `SYSTem:TIME?`

Return Format: `Integer, Integer, Integer`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not affected by Preset
SCPI Compliance:  confirmed
```

Description:

This information affects the filename times in the disk directory operations. The time information is saved with a trace and may be examined with the SDF (standard data format) utilities. Also, time information is part of a plot or print time stamp.

All numbers are rounded to the nearest integer, except for <second>, which is rounded to the resolution of the system clock.

Hours are specified and queried in the 24-hour format.

TRACe:BUFFer:COPY command

Copies the specified waterfall/spectrogram trace buffer into a data register.

Command Syntax: `TRACE:BUFFer:COPY <label1>, <label2>`

<label1> ::= {D1|D2|D3|D4|D5|D6}

<label2> ::= {TRACe1|TRACe2|TRACe3|TRACe4|CAL1|CAL2}

Example Statements:

```
OUTPUT 719;"trac:buff:copy D1, trac4"
OUTPUT 719;"Trac:Buff:Copy D2, TRACe1"
OUTPUT 719;"Trac:Buff:Copy D4, CAL1"
```

Attribute Summary:

```
Option:  AYB (waterfall/spectrogram display)
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

CAL1|CAL2 represent trace information consisting of calibration data for input channels 1 and 2, respectively.

Related Commands:

- To display the contents of a data register as a trace, use CALC:FEED.
- To store the contents of a data register to disk, use TRACE[:DATA]?.
- To move waterfall/spectrogram data between the trace buffers and mass storage units (such as disks), use MMEM:STOR:TRAC:BUFF and MMEMLOAD:TRAC:BUFF.
- To query the number of points in the data register, use TRAC[:DATA]:HEADer:POINTs?.

TRACe:COPY command

Copies the specified trace into a data register.

Command Syntax:

```
TRACE:COPY <label1>, <label2>
```

<label1> ::= {D1|D2|D3|D4|D5|D6}

<label2> ::= {TRACe1|TRACe2|TRACe3|TRACe4|CAL1|CAL2}

Example Statements:

```
OUTPUT 719;"trac:copy D1, trac4"
OUTPUT 719;"Trac:Copy D2, TRACe1"
OUTPUT 719;"Trac:Copy D4, CAL1"
```

Attribute Summary:

```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

CAL1|CAL2 represent trace information consisting of calibration data for input channels 1 and 2, respectively.

Related Commands:

- To display the contents of a data register as a trace, use CALC:FEED.
- To store the contents of a data register to disk, use TRACE[:DATA]?.
- To move trace data between the trace registers and mass storage units (as disk files), use MMEM:STOR:TRAC and MMEM:LOAD:TRAC.
- To query the number of points in the data register, use TRAC[:DATA]:HEADer:POINts?.

Trace data may be transferred directly to the controller with the LAN file-transfer utility, FTP (option UG7).

TRACe[:DATA] command/query

Moves data between the controller and the data registers.

Command Syntax: `TRACe[:DATA] {D1|D2|D3|D4|D5|D6}, <DATA>`

<DATA> ::= <DEF_BLOCK> (for real format)

or

::= <NRf>,<NRf>,<NRf>, . . . (ASCII format)

Example Statements: OUTPUT 719;"Trace:Data D6, USER2"
OUTPUT 719;"TRAC D4, USER2"

Query Syntax: `TRACe[:DATA]?`

Return Format: `If FORMat[:DATA] REAL:`

<DATA> ::= <DEF_BLOCK>

<DEF_BLOCK> ::= #<byte><length_bytes><1st_value>[...<last_value>]<NL>

<byte> ::= one byte; number of length_bytes to follow (ASCII encoded)

<length_bytes> ::= number of data bytes to follow (ASCII encoded)

<NL> ::= newline (line feed) character

`If FORMat[:DATA] ASCii:`

<DATA> ::= <1st_value>,[. . . ,<last_value>]

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  confirmed
```

Description:

The command form loads Y-axis, trace data into one of six data registers from the GPIB. The first parameter specifies the destination. The query form of this command transfers data from the analyzer over the bus to your controller. The query form requires the first parameter: TRAC? D1|D2|D3|D4|D5|D6.

Some of the data transferred with this command is not alias-protected. See following table.

Number of Displayed Frequency pts.	Baseband Data		Zoomed Data	
	Number of Points Transferred	Alias-Protected Index Range	Number of Points Transferred	Alias-Protected Index Range
51	65	0:50	64	7:57
101	129	0:100	128	14:114
201	257	0:200	256	28:228
401	513	0:400	512	56:456
801	1025	0:800	1024	112:912
1601	2049	0:1600	2048	224:1824
3201	4097	0:3200	4096	448:3648

Related Commands:

- To store a trace to a file, use MMEM:STOR:TRAC.
- To copy trace data between data registers or between a data register and a trace register, use TRACE:COPY.
- To get X-axis trace data, use TRAC:X[:DATA]?.
- To get coordinate-transformed data from a trace, use CALC:DATA?.
- To specify whether the data transformed is in ASCII or REAL, use FORM[:DATA].

Notes

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 the *GPIB Programmer's Guide*, "Block Parameters."

It is not possible to query Y-axis units – they depend on what measurement was saved in the data register. In general, they are V_{pk} for spectrums and time traces, V_{pk}^2/Hz for PSD, and V_{pk}^2 for correlation.

Trace data may be transferred directly to the controller with the LAN file-transfer utility, FTP (option UG7). Data transferred with FTP is in Standard Data Format (SDF).

TRACe[:DATA]:HEADer:POINts? query

Returns the number of points for the specified data register.

Query Syntax: `TRACe[:DATA]:HEADer:POINts? D1|D2|D3|D4|D5|D6`

Example Statements:
OUTPUT 719;":trace:head:points? D5"
OUTPUT 719;"Trac:Data:Header:Poin? D6"

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  not applicable
SCPI Compliance:  instrument-specific
```

Description:

This query is used to determine the number of data points in the specified data register. This information may be required to transfer the contents of the register from the analyzer to the controller.

The data points may be complex or real. If it is complex, then TRAC:DATA? outputs 2 numbers per point, corresponding to the real and imaginary parts of the data point.

Related Commands:

- To transfer trace data to a data register, use TRACe:COPY.
- To transfer the contents of the data register to or from the controller, use TRACe[:DATA].
- To copy a trace directly to disk, use MMEM:STOR:TRAC.

TRACe:X[:DATA]? query

Returns the X-axis data for trace displays.

Query Syntax: `TRACe:X[:DATA]? D1|D2|D3|D4|D5|D6|TRACe1|TRACe2|TRACe3|TRACe4`

Example Statements:
OUTPUT 719;"TRACE:X:DATA? D2"
OUTPUT 719;"trace:x? D6"

Return Format: `If FORMat[:DATA] REAL:`

<DATA>	::=	<DEF_BLOCK>
<DEF_BLOCK>	::=	#<byte><length_bytes><1st_value>[. . .<last_value>]<NL>
<byte>	::=	one byte specifying the number of length_bytes to follow (ASCII encoded)
<length_bytes>	::=	number of data bytes to follow (ASCII encoded)
<NL>	::=	newline (linefeed) character

`If FORMat[:DATA] ASCii:`

<DATA>	::=	<1st_value>,[. . . ,<last_value>]

Attribute Summary:
```
Synchronization Required:  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. This is the companion command for TRACE:DATA? which returns (alias-protected) data from the data registers.

Related Commands:

- To determine the units for the X-axis send TRAC:X:UNIT?.

TRACe:X:UNIT? query

Returns the x-axis units for trace displays.

Query Syntax: `TRACe:X:UNIT? D1|D2|D3|D4|D5|D6|TRACe1|TRACe2|TRACe3|TRACe4`

Example Statements:
OUTPUT 719;":Trace:X:Unit? D1"
OUTPUT 719;"TRACE:X:UNIT? D3"

Return Format: `STRING`

Attribute Summary:
```
Synchronization Required:  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.

Related Commands:

- To query the values along the X axis, use TRAC:X[:DATA]?.
- To query the Y-axis trace data, use TRAC[:DATA]?.

TRIGger:HOLDoff:DELay command/query

Specifies the delay after IF trig occurs before next trigger accepted.

Command Syntax: `TRIGger:HOLDoff:DELay {<number>[<unit>]}|<step>|<bound>`

<number>	::=	a real number (NRf data) limits: 0:41.94
<unit>	::=	[S]
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

OUTPUT 719;"trig:hold:delay 6.61666"
OUTPUT 719;"Trig:Holdoff:Del 22.9661"

Query Syntax: `TRIGger:HOLDoff:DELay?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0 s
SCPI Compliance:  instrument-specific
```

Description:

For more information on this feature, see online help (press the [**Help**] key on the front panel, then press the softkey of interest).

Related Commands:

- To activate trigger holdoff, use TRIG:HOLD:STAT.
- To specify the IF trigger type, use TRIG:SOUR.

TRIGger:HOLDoff:STATe command/query

Specifies the state (on/off) of the trigger holdoff.

Command Syntax: `TRIGger:HOLDoff:STATe OFF|0|ON|1`

Example Statements:
```
OUTPUT 719;":TRIG:HOLDOFF:STAT OFF"
OUTPUT 719;"trigger:hold:stat OFF"
```

Query Syntax: `TRIGger:HOLDoff:STATe?`

Return Format: `Integer`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  0 (OFF)
SCPI Compliance:  instrument-specific
```

Description:

For more information on this feature, see online help (press the [**Help**] key on the front panel, then press the softkey of interest).

Related Commands:

- To specify the amount of trigger holdoff delay, use TRIG:HOLD:DEL.
- To specify the IF trigger type, use TRIG:SOUR.

TRIGger:LEVel — command/query

Specifies the signal level at which a measurement is triggered.

Command Syntax: `TRIGger:LEVel <param>`

<param>	::=	<number>[<unit>]|<step>|<bound>
<number>	::=	a real number (NRf data) limits: (range depends on configuration)
<unit>	::=	PCT|V (and dB for IF trigger)
<step>	::=	UP|DOWN
<bound>	::=	MAX|MIN

Example Statements:

```
OUTPUT 719;"TRIG:LEV 5V"
OUTPUT 719;"trig:level -2"
```

Query Syntax: `TRIGger:LEVel?`

Return Format: `Real`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  0%
SCPI Compliance:  confirmed
```

Description:

If the trigger typs is internal channel (TRIG:SOUR INT1), the trigger level is entered as a percent of the input range (VOLT:RANG).

If the trigger type is external trigger (TRIG:SOUR EXT), the trigger signal is connected to the front panel connector labeled EXT TRIGGER and trigger level is entered in units of volts or %. The level is an analog voltage between –11V and +11V or –110% to +110%.

If the trigger type is *IF*, the trigger level may be entered in units of volts, %, or dB.

Trigger level is not applicable for GPIB trigger (TRIG:SOUR BUS), source trigger (TRIG:SOUR OUTP), or free run (TRIG:SOUR IMM).

Trigger level can be entered over GPIB in percentage or as a fraction. A query always returns the fractional value. If the trigger type is *external*, the trigger level is entered in Volts and the returned value is in volts. IF trigger also returns volts.

See TRIG:SOUR for more information.

TRIGger:SLOPe **command/query**

Specifies whether a trigger occurs when the selected trigger source signal is positive-with-respect-to or negative-with-respect-to the specified trigger level.

Command Syntax: `TRIGger:SLOPe POSitive|NEGative`

Example Statements:
OUTPUT 719;"Trigger:Slop POSITIVE"
OUTPUT 719;"TRIGGER:SLOP POSITIVE"

Query Syntax: `TRIGger:SLOPe?`

Return Format: `CHAR`

Attribute Summary:
```
Synchronization Required:  no
Preset State:  POS
SCPI Compliance:  confirmed
```

Description:

Slope does not apply to free run, source, or GPIB trigger types. These trigger types operate independently of the slope setting. Trigger delay may also be specified with the SWE:TIM:DEL command.

Trigger type is specified with the TRIG:SOUR command.
Trigger level is specified with the TRIG:LEV command.

The analyzer must be armed before triggering can start a measurement. See ARM:SOUR for more information.

TRIGger:SOURce command/query

Specifies the source (or type) of triggering used to start measurements.

Command Syntax: `TRIGger:SOURce <param>`

<param> ::= IMMediate|INT1|INT2|IF1|IF2|OUTPut|BUS|EXTernal

Example Statements:

```
OUTPUT 719;"trigger:source INT1"
OUTPUT 719;"Trig:Source OUTP"
```

Query Syntax: `TRIGger:SOURce?`

Return Format: `CHAR`

Attribute Summary:

```
Synchronization Required:  no
Preset State:  IMM
SCPI Compliance:  confirmed
```

Description:

The analyzer must be armed before triggering can occur. See ARM:SOUR.

IMM (immediate or free run) specifies that successive data blocks are to be taken as quickly as possible without waiting for a trigger signal.

INT1|INT2 specifies that the input data blocks should be acquired when the channel (1|2) sampled input data stream matches the trigger conditions (slope and level are set with TRIG:SLOP and TRIG:LEV). Note that the second channel is an option and may not be installed in the analyzer.

IF1|IF2 specifies the IF signal from either input channel 1 or 2 as the trigger signal. This means the analyzer begins a measurement when the IF signal meets the specified trigger slope and level conditions. IF trigger is a narrow-band trigger compared to the INT trigger. Some signal types may require the use of trigger holdoff; see TRIG:HOLD. Note that the second channel is an option and may not be installed in the analyzer.

OUTP specifies that the input data blocks should be acquired synchronously with the source.

BUS specifies that the input data blocks should be acquired when the GPIB Group Execute Trigger command is received.

EXT specifies that the input data blocks are acquired when the front panel EXT TRIGGER signal meets the level and slope criteria set with the TRIG:LEV and TRIG:SLOP commands. The level is an analog voltage between –11.0V and +11.0V. Slope, in this case, indicates which side of the level value the external signal must be to trigger a measurement (POS means ‘’more positive’’ and NEG means ‘’more negative’’).

Note Channel 2 is optional and may not be installed. If the second channel is not installed, IF2 and INT2 generate error messages. If no number is used, “1" is assumed.

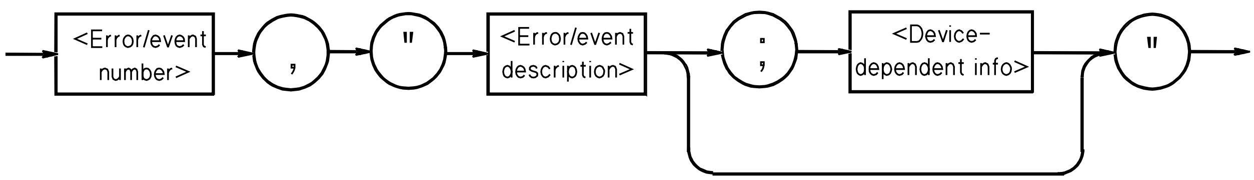
A

Error Messages

SYSTem:ERRor?

The queue query message is a request for the next entry from the instrument's error/event queue. This queue contains an integer in the range [-32768, 32767]. Negative error numbers are reserved by the SCPI standard and defined in this document. Positive error numbers are instrument-dependent. An error/event value of zero indicates that no error or event has occurred (see next section, ''The queue'').

The instrument responds to SYSTem:ERRor? query using the following form:



The <error/event number> is a unique error/event descriptor. Certain standard error/event numbers are described in this document. The <error description> is a short description of the error/event, (optionally) followed by further information about the error/event. Short descriptions of the standard error/event numbers are given in this document.

The <device-dependent info> part of the response may contain information which allows you to determine the exact error/event and context. For example,

-131, "Invalid suffix; FREQuency:CENT 2.0E+5 dBuV"

The maximum string length of <error description> plus <device-dependent information> is 255 characters.

The Error/Event Queue

As errors and events are detected, they are placed in a queue. This queue is first in, first out. If the queue overflows, the last error/event in the queue is replaced with error

-350, "Queue overflow"

Any time the queue overflows, the least recent errors remain in the queue, and the most recent error/event is discarded. Reading an error/event from the head of the queue removes that error/event from the queue, and opens a position at the tail of the queue for a new error/event, if one is subsequently detected.

When all errors/events have been read from the queue, further error/event queries return

0, ''No error''

Individual erros and events may be enabled into the queue. The STATus:QUEue:ENABle command accomplishes this. At STATus:PRESet, only errors are enabled. This means that both SYSTem:ERRor? and STATus:QUEue[:NEXT]? report only errors unless the user changes the neable mask.

The error/event queue is cleared when any of the following occur (*IEEE 488.2*, section 11.4.3.4):

- Upon power up.
- Upon receipt of a *CLS command.
- Upon reading the last item from the queue.

Error numbers

The system-defined error/event numbers are chosen on an enumerated ("1 of N") basis. The SCPI-defined error/event numbers and the <error description> portions of the ERRor query response are listed here. The first error/event described in each class (for example, -100, -200, -300, -400) is a ''generic'' error. In selecting the proper Error/event number to report, more specific error/event codes are preferred, and the generic error/event is used only if the others are inappropriate.

Note the organization of the following tables. A ''simple-minded'' parser might implement only the XX0 errors, and a smarter one might implement all of them. A ''smart and friendly'' parser might use the instrument-dependent part of the error/event message string to point out the offending part of the command.

No Error

This message indicates that the device has no errors.

Error Number	SCPI No-Error Description
0	No error. The queue is competely empty. Every error/event in the queue has been read or the queue was purposely cleared by power-on, *CLS

Command Error

An <error/event number> in the range [-199 , -100] indicates that an *IEEE 488.2* syntax error has been detected by the instrument's parser. The occurrence of any error in this class causes the command error bit (bit 5) in the event status register (*IEEE 488.2*, section 11.5.1) to be set. One of the following has occurred:

- An *IEEE 488.2* syntax error has been detected by the parser. That is, a controller-to-device message was received which is in violation of the *IEEE 488.2* standard. Possible violations include a data element which violates the device listening formats or whose type is unacceptable to the device.
- An unrecognized header was received. Unrecognized headers include incorrect device-specific headers and incorrect or unimplemented *IEEE 488.2* common commands.
- A Group Execute Trigger (GET) was entered into the input buffer inside of an *IEEE 488.2* <PROGRAM MESSAGE>.

Events that generate command errors do not generate execution errors, device-specific errors, or query errors; see the other error definitions in this chapter.

Error Number	SCPI Command Error Descriptions
-100	Command error. This is the generic syntax error for **devices** that cannot detect more specific errors. This code indicates only that a Command Error as defined in *IEEE 488.2*, 11.5.1.1.4 has occurred.
-101	Invalid character. A syntactic element contains a character which is invalid for that type; for example, a header containing an ampersand, SETUP&. This error might be used in place of errors -114, -121, -141, and perhaps others.
-102	Syntax error. An unrecognized command or data type was encountered; for example, a string was received when the **device** does not accept strings.
-103	Invalid separator. The parser was expecting a separator and encountered an illegal character; for example, the semicolon was omitted after a program message unit, *EMC 1 :CH1:VOLTS 5.
-104	Data type error. The parser recognized a data element different than one allowed; for example, numeric or string data was expected but block data was encountered.
-105	GET not allowed. A Group Execute Trigger was received within a program message (see *IEEE 488.2*, 7.7).
-108	Parameter not allowed. More parameters were received than expected for the header; for example, the *EMC common command only accepts one parameter, so receiving *EMC 0,,1 is not allowed.
-109	Missing parameter. Fewer parameters were recieved than required for the header; for example, the *EMC common command requires one parameter, so receiving *EMC is not allowed.
-110	Command header error. An error was detected in the header. This error message is used when the **device** cannot detect the more specific errors described for errors -111 through -119.
-111	Header separator error. A character which is not a legal header separator was encountered while parsing the header; for example, no white shace followed the header, thus *GMC"MACRO" is an error.
-112	Program mnemonic too long. The header contains more that twelve characters (see *IEEE 488.2*, 7.6.1.4.1).]
-113	Undefined header. The header is syntactically correct, but it is undefined for this specific **device**; for example, *XYZ is not defined for any **device**.
-114	Header suffix out of range. The value of a numeric suffix attached to a program mnemonic, see Syntax and Style section 6.2.5.2, makes the header invalid.

Error Number	SCPI Command Error Descriptions
-120	Numeric data error. This error, as well as errors -121 through -129, are generated when parsing a data element which apprears to be numeric, including the nondecimal numeric types. This particular error message is used if the **device** cannot detect a more specific error.
-121	Invalid character in number. An invalid character for the data type being parsed was encountered; for example, an alpha in a decimal numeric or a ''9'' in octal data.
-123	Exponent too large. The magnitude of the exponent was larger than 32000 (see *IEEE 488.2*, 7.7.2.4.1).
-124	Too many digits. The mantissa of a decimal numeric data element contained more than 255 digits excluding leading zeros (see *IEEE 488.2*, 7.7.2.4.1).
-128	Numeric data not allowed. A legal numeric data element was received, but the **device** does not accept one in this position for the header.
-130	Suffix error. This error, as well as errors -131 through -139, are generated when parsing a suffix. This particular error message is used if the **device** cannot detect a more specific error.
-131	Invalid suffix. The suffix does not follow the syntax described in *IEEE 488.2*, 7.7.3.2, or the suffix is inappropriate for this **device.**
-134	Suffix too long. The suffix contained more than 12 characters (see *IEEE 488.2*, 7.7.3.4).
-138	Suffix not allowed. A suffix was encountered after a numeric element which does not allow suffixes.
-140	Character data error. This error, as well as errors -141 through -149, are generated when parsing a character data element. This particular error message is used if the **device** cannot detect a more specific error.
-141	Invalid character data. Either the character data element contains an invalid character or the particular element received is not valid for the header.
-144	Character data too long. The character data element contains more than twelve characters (see *IEEE 488.2*, 7.7.1.4).
-148	Character data not allowed. A legal character data element was encountered where prohibited by the **device**.
-150	String data error. This error, as well as errors -151 through -159, are generated when parsing a string data element. This particular error message is used if the **device** cannot detect a more specific error.
-151	Invalid string data. A string data element was expected, but was invalid for some reason (see *IEEE 488.2*, 7.7.5.2); for example, an END message was received before the terminal quote character.
-158	String data not allowed. A string data element was encountered but was not allowed by the **device** at this point in parsing.
-160	Block data error. This error, as well as errors -161 through -169, are generated when parsing a block data element. This particular error message is used if the **device** cannot detect a more specific error.
-161	Invalid block data. A block data element was expected, but was invalid for some reason (see *IEEE 488.2*, 7.7.6.2); for example, an END message was received before the length was satisfied.
-168	Block data not allowed. A legal block data element was encountered but was not allowed by the **device** at this point in parsing.
-170	Expression error. This error, as well as errors -171 through -179, are generated when parsing an expression data element. This particular error message is used if the **device** cannot detect a more specific error.
-171	Invalid expression. The expression data element was invalid (see *IEEE 488.2*, 7.7.7.2); for example, unmatched parentheses or an illegal character.

Error Number	SCPI Command Error Descriptions
-178	Expression data not allowed. A legal expression data was encountered but was not allowed by the **device** at this point in parsing.
-180	Macro error. This error, as well as errors -181 through -189, are generated when defining a macro or executing a macro. This particular error message is used if the **device** cannot detect a more specific error.
-181	Invalid outside macro definition. Indicates that a macro parameter placeholder ($ < number) was encountered outside of a macro definition.
-183	Invalid inside macro definition. Indicates that the program message unit sequence, sent with a *DDT or *DMC command, is syntactically invalid (see *IEEE 488.2*, 10.7.6.3).
-184	Macro parameter error. Indicates that a command inside the macro definition had the wrong number or type of parameters.

Execution Error

An <error/event number> in the range [-299 ,, -200] indicates that an error has been detected by the instrument's execution control block. The occurrence of any error in this class causes the execution error bit (bit 4) in the event status register (*IEEE 488.2*,, section 11.5.1) to be set. One of the following events has occurred:

- A <PROGRAM DATA> element following a header was evaluated by the device as outside of its legal input range or is otherwise inconsistent with the device's capabilities.
- A valid program message could not be properly executed due to some device condition.

Execution errors are reported by the device after rounding and expression evaluation operations have taken place. Rounding a numeric data element, for example, is not be reported as an execution error. Events that generate execution errors do not generate Command Errors, device-specific errors, or Query Errors; see the other error definitions in this section.

Error Number	SCPI Execution Error Descriptions
-200	Execution error. This is the generic syntax error for **devices** that cannot detect more specific errors. This code indicates only that an Execution Error as defined in *IEEE 488.2*, 11.5.1.1.5 has occurred.
-201	Invalid while in local. Indicates that a command is not executable while the **device** is in local due to a hard local control (see *IEEE 488.2*, 5.6.1.5); for example, a **device** with a rotary switch receives a message which would change the switches state, but the **device** is in local so the message can not be executed.
-202	Settings lost due to rtl. Indicates that a setting associated with a hard local control (see *IEEE 488.2*, 5.6.1.5) was lost when the **device** changed to LOCS from REMS or to LWLS from RWLS.
-210	Trigger error
-211	Trigger ignored. Indicates that a GET, *TRG, or triggering signal was received and recognized by the device but was ignored because of device timing considerations; for example, the device was not ready to respond. Note: a DTO device always ignores GET and treats *TRG as a Command Error.
-212	Arm ignored. Indicates that an arming signal was received and recognized by the **device** but was ignored.
-213	Init ignored. Indicates that a request for a measurement initiation was ignored as another measurement was already in progress.
-214	Trigger deadlock. Indicates that the trigger source for the initiation of a measurement is set to GET and subsequent measurement query is received. The measurement cannot be started until a GET is received, but the GET would cause an INTERRUPTED error.
-215	Arm deadlock. Indicates that the arm source for the initiation of a measurement is set to GET and subsequent measurement query is received. The measurement cannot be started until a GET is received, but the GET would cause an INTERRUPTED error.
-220	Parameter error. Indicates that a program data element related error occurred. This error message is used when the **device** cannot detect the more specific errors described for errors -221 through -229.
-221	Settings conflict. Indicates that a legal program data element was parsed but could not be executed due to the current device state (see *IEEE 488.2*, 6.4.5.3 and 11.5.1.1.5.)
-222	Data out of range. Indicates that a legal program data element was parsed but could not be executed because the interpreted value was outside the legal range as defined by the **device** (see *IEEE 488.2*, 11.5.1.1.5.)
-223	Too much data. Indicates that a legal program data element of block, expression, or string type was received that contained more data than the device could handle due to memory or related device-specific requirements.

Error Number	SCPI Execution Error Descriptions
-224	Illegal parameter value. Used where exact value, from a list of possibles, was expected.
-225	Out of memory. The device has insufficent memory to perform the requested operation.
-226	Lists not same length. Attempted to use LIST structure having individual LIST's of unequal lengths.
-230	Data corrupt or stale. Possibly invalid data; new reading started but not completed since last access.
-231	Data questionable. Indicates that measurement accuracy is suspect.
-240	Hardware error. Indicates that a legal program command or query could not be executed because of a hardware problem in the **device**. Definition of what constitutes a hardware problem is completely device-specific. This error message is used when the **device** cannot detect the more specific errors described for errors -241 through -249.
-241	Hardware missing. Indicates that a legal program command or query could not be executed because of missing **device** hardware; for example, an option was not installed. Definition of what constitutes missing hardware is completely device-specific.
-250	Mass storage error. Indicates that a mass storage error occurred. This error message is used when the **device** cannot detect the more specific errors described for errors -251 through -259.
-251	Missing mass storage. Indicates that a legal program command or query could not be executed because of missing mass storage; for example, an option that was not installed. Definition of what constitutes missing mass storage is device-specific.
-252	Missing media. Indicates that a legal program command or query could not be executed because of a missing media; for example, no disk. The definition of what constitutes missing media is device-specific.
-253	Corrupt media. Indicates that a legal program command or query could not be executed because of corrupt media; for example, bad disk or wrong format. The definition of what constitutes corrupt media is device-specific.
-254	Media full. Indicates that a legal program command or query could not be executed because the media was full; for example, there is no room on the disk. The definition of what constitutes a full media is device-specific.
-255	Directory full. Indicates that a legal program command or query could not be executed because the media directory was full. The definition of what constitutes a full media directory is device-specific.
-256	File name not found. Indicates that a legal program command or query could not be executed because the file name on the device media was not found; for example, an attempt was made to read or copy a nonexistent file. The definition of what constitutes a file not being found is device-specific.
-257	File name error. Indicates that a legal program command or query could not be executed because the file name on the device media was in error; for example, an attempt was made to copy to a duplicate file name. The definition of what constitutes a file name error is device-specific.
-258	Media protected. Indicates that a legal program command or query could not be executed because the media was protected; for example, the write-protect tab on a disk was present. The definition of what constitutes protected media is device-specific.
-260	Expression error. Indicates that a expression program data element related error occurred. This error message is used when the **device** cannot detect the more specific errors described for errors -261 through -269.
-261	Math error in expression. Indicates that a syntactically legal expression program data element could not be executed due to a math error; for example, a divide-by-zero was attempted. The definition of math error is device-specific.
-270	Macro error. Indicates that a macro-related execution error occurred. This error message is used when the **device** cannot detect the more specific errors described for errors -271 through -279.

Error Number	SCPI Execution Error Descriptions
-271	Macro syntax error. Indicates that that a syntactically legal macro program data sequence, according to *IEEE 488.2*, 10.7.2, could not be executed due to a syntax error within the macro definition (see *IEEE 488.2*, 10.7.6.3.)
-272	Macro execution error. Indicates that a syntactically legal macro program data sequence could not be executed due to some error in the macro definition (see *IEEE 488.2*, 10.7.6.3.)
-273	Illegal macro label. Indicates that the macro label defined in the *DMC command was a legal string syntax, but could not be accepted by the **device** (see *IEEE 488.2*, 10.7.3 and 10.7.6.2); for example, the label was too long, the same as a common command header, or contained invalid header syntax.
-274	Macro parameter error. Indicates that the macro definition improperly used a macro parameter placeholder (see *IEEE 488.2*, 10.7.3).
-275	Macro definition too long. Indicates that a syntactically legal macro program data sequence could not be executed because the string or block contents were too long for the device to handle (see *IEEE 488.2*, 10.7.6.1).
-276	Macro recursion error. Indicates that a syntactically legal macro program data sequence could not be executed because the device found it to be recursive (see *IEEE 488.2*, 10.7.6.6).
-277	Macro redefinition not allowed. Indicates that a syntactically legal macro label in the *DMC command could not be executed because the macro label was already defined (see *IEEE 488.2*, 10.7.6.4).
-278	Macro header not found. Indicates that a syntactically legal macro label in the *GMC? query could not be executed because the header was not previously defined.
-280	Program error. Indicates that a downloaded program-related execution error occurred. This error message is used when the **device** cannot detect the more specific errors described for errors -281 through -289. A downloaded program is used to add algorithmic capability to a **device.** The syntax used in the program and the mechanism for downloading a program is device-specific.
-281	Cannot create program. Indicates that an attempt to create a program was unsuccessful. A reason for the failure might include not enough memory.
-282	Illegal program name. The name used to reference a program was invalid; for example, redefining an existing program, deleting a nonexistent program, or in general, referencing a nonexistent program.
-283	Illegal variable name. An attempt was made to reference a nonexistent variable in a program.
-284	Program currently running. Certain operations dealing with programs may be illegal while the program is running; for example, deleting a running program might not be possible.
-285	Program syntax error. Indicates that a syntax error appears in a downloaded program. The syntax used when parsing the downloaded program is device-specific.
-286	Program runtime error

Device-Specific Error

An <error/event number> in the range [-399 , -300] or [1 , 32767] indicates that the instrument has detected an error which is not a command error, a query error, or an execution error; some device operations did not properly complete, possibly due to an abnormal hardware or firmware condition. These codes are also used for self-test response errors. The occurrence of any error in this class causes the device-specific error bit (bit 3) in the event status register (*IEEE 488.2*, section 11.5.1) to be set. The meaning of positive error codes is device-dependent and may be enumerated or bit mapped; the <error message> string for positive error codes is not defined by SCPI and available to the device designer. Note that the string is not optional; if the designer does not wish to implement a string for a particular error, the null string is be sent (for example, 42,""). The occurrence of any error in this class causes the device-specific error bit (bit 3) in the event status register (*IEEE 488.2*, section 11.5.1) to be set. Events that generate device-specific errors do not generate command errors, execution errors, or query errors; see the other error definitions in this section.

Error Number	SCPI Device-Specific Error Description
-300	Device-specific error. This is the generic device-dependent error for **devices** that cannot detect more specific errors. This code indicates only that a Device-Dependent Error as defined in *IEEE 488.2*
-310	System error. Indicates that some error
-311	Memory error. Indicates that an error was detected in the **device's** memory. The scope of this error is device-dependent.
-312	PUD memory lost. Indicates that the protected user data saved by the *PUD command has been lost.
-313	Calibration memory lost. Indicates that nonvolatile calibration data used by the *CAL? command has been lost.
-314	Save/recall memory lost. Indicates that the nonvolatile data saved by the *SAV? command has been lost.
-315	Configuration memory lost. Indicates that nonvolatile configuration data saved by the **device** has been lost. The meaning of this error is device-specific.
-330	Self-test failed
-350	Queue overflow. A specific code entered into the queue in lieu of the code that caused the error. This code indicates that there is no room in the queue and an error occurred but was not recorded.

Query Error

An <error/event number> in the range [-499 , -400] indicates that the output queue control of the instrument has detected a problem with the message exchange protocol described in *IEEE 488.2*, chapter 6. The occurrence of any error in this class cause the query error bit (bit 2) in the event status register (*IEEE 488.2*, section 11.5.1) to be set. These errors correspond to message exchange protocol errors described in *IEEE 488.2*, section 6.5. One of the following is true:

- An attempt is being made to read data from the output queue when no output is present or pending;
- Data in the output queue has been lost.

Events that generate query errors do not generate command errors, execution errors, or device-specific errors; see the other error definitions in this section.

Error Number	SCPI Query Error Description
-400	Query error. This is the generic query error for **devices** that cannot detect more specific errors. This code indicates only that a Query Error as defined in *IEEE 488.2*, 11.5.1.1.7 and 6.3 has occurred.
-410	Query INTERRUPTED. Indicates that a condition causing an INTERRUPTED Query error occurred (see *IEEE 488.2*, 6.3.2.3); for example, a query followed by DAB or GET before a response was completely sent.
-420	Query UNTERMINATED. Indicates that a condition causing an UNTERMINATED Query error occurred (see *IEEE 488.2*, 6.3.2.2); for example, the **device** was addressed to talk and an incomplete program message was received.
-430	Query DEADLOCKED. Indicates that a condition causing an DEADLOCKED Query error occurred (see *IEEE 488.2*, 6.3.1.7); for example, both input buffer and output buffer are full and the device cannot continue.
-440	Query UNTERMINATED after indefinite response. Indicates that a query was received in the same program message after an query requesting an indefinite response was executed (see *IEEE 488.2*, 6.5.7.5).

Instrument-Specific Error Messages

Error Number	Error Message
-100	“Command is query-only.”
-108	“Parameter not allowed”
-109	“Missing parameter”
-131	“Invalid suffix”
-141	“Invalid character data”
-151	“Invalid string data”
-161	“Data block does not contain valid math function definitions.”
-161	“Invalid block data”
-200	“Application already loaded.”
-200	“Application incompatible with other loaded applications.n”
-200	“Application not loaded.”
-200	“Beginning of capture data on channel %d.”
-200	“Capture aborted. %s”
-200	“Capture span is too narrow for symbol rate.”
-200	“Capture span is too wide for symbol rate.”
-200	“Data register is undefined.”
-200	“Delete program is not allowed while recording enabled.”
-200	“Download program is not allowed while recording enabled.”
-200	“End of capture data on channel %d.”
-200	“F%d definition is not valid for execution.”
-200	“F%d execution requires recursion.”
-200	“F%d is not defined.”
-200	“Failure of communication link with RF section. Option configuration cannot be changed.”
-200	“File operation aborted, capture changed.”
-200	“File operation aborted, capture in progress.”
-200	“File operation aborted, waterfall/spectrogram changed.”
-200	“File operation not completed.”
-200	“Function definition is not valid. Operand in math function is not supported in this instrument configuration.”

Error Number	Error Message
-200	"Function definition is not valid."
-200	"Function definition is too long."
-200	"Function definition may not reference higher numbered functions."
-200	"General Export Option Installed. Command not allowed."
-200	"GPIB control received without requesting it."
-200	"GPIB control was not received."
-200	"Instrument BASIC is not installed."
-200	"Instrument mode must be Vector to measure from capture."
-200	"Insufficient memory for %s on trace %s. %s limited to %d scans. Use [System Utility] → [memory usage] → [remove trace buffers] to reclaim all available memory."
-200	"Insufficient memory for this instrument mode. See [System Utility] → [memory usage] to free more memory."
-200	"Insufficient memory. Max freq points and/or max time points adjusted."
-200	"Invalid X11 display IP address"
-200	"Invalid date entered. Date unchanged."
-200	"Invalid function code"
-200	"Invalid instrument state"
-200	"Marker value is not valid."
-200	"Meas Restart, Pause or Continue received while capture in progress. Abort capture first."
-200	"No capture data for channel %d."
-200	"No capture data."
-200	"No more applications allowed."
-200	"Not a valid serial number"
-200	"Not enough capture data for any measurement result."
-200	"Offset freq is a display only paramater in external IF input mode."
-200	"Offset value of 0 is not allowed."
-200	"Option is not installed."
-200	"Plot/print is already in progress."
-200	"Printer/plotter is not on line."
-200	"Printer/plotter out of paper."
-200	"Printer/plotter reports error."

Error Number	Error Message
-200	“Program memory re-size is not allowed while recording enabled.”
-200	“Program variable access not allowed while recording enabled.”
-200	“Receiver must be 0-10 MHz to measure from capture.”
-200	“Receiver must be 2-1800 MHz to measure from capture.”
-200	“Receiver must be 2-2650 MHz to measure from capture.”
-200	“Receiver must be External to measure from capture.”
-200	“Recording mode canceled because: %s”
-200	“Remote X11 display connection closed”
-200	“Result length truncated to fit within search length.”
-200	“Save/recall program is not allowed during power-on calibration.”
-200	“Save/recall program is not allowed while recording enabled.”
-200	“Search length or result length adjusted due to insufficient Time Capture RAM. For details, see online help for the [search length] softkey.”
-200	“Search length, result length or points/symbol adjusted See [System Utility] → [memory usage] to set the ‘max time points’ limit and/or ‘max span/symR’ ratio.”
-200	“Search length, result length or points/symbol adjusted See [System Utility] → [memory usage] to set the ‘max time points’ limit.”
-200	“Search target not found.”
-200	“Span adjusted. Maximum span limited by symbol rate and maximum span/symbol rate ratio. See [System Utility] → [memory usage] to set the ‘max span/symR’ ratio.”
-200	“Span adjusted. Minimum span limited by symbol rate.”
-200	“Sync offset adjusted. Sync offset limited to maintain sync pattern length within search length.”
-200	“Temporary math buffer(s) unavailable. Math expression too complicated. Simplify math expression in terms of other math functions.”
-200	“Temporary math buffer(s) unavailable. See [System Utility] → [memory usage].”
-200	“The Agilent Technologies Instrument BASIC editor has been disabled.”
-200	“The custom application is not compatible with the analyzer’s main firmware. Please contact your local Agilent Technologies Service Center for firmware upgrade information.”
-200	“This parameter is set at the factory.”
-200	“Trace does not contain valid waterfall/spectrogram data.”
-200	“Unable to open remote display”
-200	“Unsupported operation”

Error Number	Error Message
-200	“Value limited by Time Capture RAM, set to limit. For details, see online help for the [search length] softkey.”
-200	“Value out of range, set to limit. Maximum span limited by symbol rate and maximum span/symbol rate ratio. See [System Utility] → [memory usage] to set the ‘max span/symR’ ratio.”
-200	“Value out of range, set to limit. Minimum span limited by symbol rate.”
-200	“Value out of range, set to limit. See [System Utility] → [memory usage] to set the ‘max time points’ limit and/or ‘max span/symR’ ratio.”
-200	“Value out of range, set to limit. See [System Utility] → [memory usage] to set the ‘max time points’ limit.”
-200	“Value out of range, set to limit. See [System Utility] → [memory usage] to set the maximum limit.”
-200	“Value out of range, set to limit. Sync offset limited to maintain sync pattern length within search length.”
-200	“Value out of range, set to limit.”
-200	“Warning: Input Time exceeds ‘max time points’ and will be truncated. See [System Utility] → [memory usage] to set the ‘max time points’ limit.”
-200	“Warning: Insufficient memory. Trace buffer truncated to %d scans.”
-200	“Warning: Math data results truncated.”
-220	“Parameter error”
-221	“%s memory configuration cannot be changed unless the IBASIC memory size is 0. IBASIC can be redefined after the %s configuration is complete.”
-221	“Capture functionality is not available in Scalar instrument mode.”
-221	“Duplicate state detected in %s definition.”
-221	“Feature is disabled until the instrument is powered up with LAN active.”
-221	“Invalid domain for marker operation.”
-221	“Invalid program state change requested.”
-221	“Marker is not on.”
-221	“Multiple overlaid waterfall/spectrograms not allowed.”
-221	“Offset marker is not on.”
-221	“Peak hold averaging valid only for spectrum, PSD, and time measurement data.”
-221	“Register data not compatible with user defined filtering.”
-221	“Request not supported with selected device.”
-221	“Resolution bandwidth limited by maximum time record length.”
-221	“Resolution bandwidth limited by minimum time record length.”
-221	“Scalar instrument mode is not available when capture is on.”

Error Number	Error Message
-221	"Scalar instrument mode is not available when receiver mode is IF section (ch1 + j*ch2)"
-221	"Select DC coupling on both input channels to avoid IQ offset errors."
-221	"Source Level is 0 Volts. Measurement aborted."
-221	"The LAN port cannot be changed while the X11 display is on."
-221	"Type of unit setting does not match the unit query."
-222	"Data out of range"
-224	"Illegal parameter value"
-224	"REAL format length is only 32 and 64."
-230	"Trace contains invalid data."
-240	"%s Please see your Service Guide for instructions."
-240	"10 MHz reference loop unlocked in RF section."
-240	"600 MHz loop unlocked in RF section."
-240	"Error when writing Cal Data to Flash memory in RF section."
-240	"Failure of communication link with RF section."
-240	"RF section external reference not detected."
-241	"Channel 2 is not available."
-241	"Option 1D4 Source is not installed."
-250	"Bad disk"
-250	"Bad mass storage parameter"
-250	"Can't name split file. Enter a shorter filename."
-250	"Disk file/unit may be corrupt."
-250	"External disk is not SS/80 protocol."
-250	"External disk is not responding."
-250	"External system controller detected. Change to system controller not allowed."
-250	"File does not contain a capture."
-250	"File does not contain a state."
-250	"File does not contain a trace."
-250	"File does not contain a waterfall/spectrogram."
-250	"File does not contain math function definitions."
-250	"File system error"

Error Number	Error Message
-250	“Format aborted: file(s) are open.”
-250	“GPIB system controller needed”
-250	“Improper file name”
-250	“Improper file type”
-250	“Improper mass storage unit specifier”
-250	“Install aborted: invalid option.”
-250	“Invalid SDF file format.”
-250	“Mass storage units must be the same when renaming.”
-250	“No memory available”
-250	“Operation failed on one (or more) files.”
-250	“Permission denied”
-250	“SDF feature not supported.”
-250	“Source and destination units are the same.”
-250	“State not recalled. Number of trace data points is greater than current maximum frequency points.”
-250	“Too many disk units active”
-250	“Unexpected end of file”
-250	“Wildcard expands to more than one file.”
-250	“Wildcard not allowed”
-251	“Mass storage unit is not present.”
-252	“Disk not in drive”
-253	“Not a valid directory”
-254	“File is too large. Press ‘continue save’ to split file.”
-254	“Insert next disk with file ‘%s’. Press ‘continue recall’ to proceed.”
-254	“Insert next disk. Press ‘continue save’ to proceed.”
-254	“Insufficient disk space”
-255	“Directory full”
-256	“File name not found”
-257	“Duplicate file name”
-258	“Write protected disk”
-283	“Illegal variable name”

Error Number	Error Message
-284	"IBASIC program is currently running."
-285	"Downloaded program line must have a line number."
-285	"ERROR 949 Syntax error at cursor."
-310	"Calibration failure Check front panel connections."
-310	"Calibration failure"
-310	"Warning: Invalid calibration data. Single cal required."
-311	"Out of memory"
-350	"Instrument Busy"

%s is a variable string such as (for Caputure aborted) to indicate the amount of time signal captured.

%d is a variable number that is used to indicate channel number (1 or 2) or function number (1–6).

B

Sockets Example Program

```
/*
 ****************************************************************************
 *
 *  $Description:     Functions to talk to an HP89410/440/441 via TCP/IP $
 *
 *  $Examples: $
 *
 *   Query the center frequency:
 *        lanio 15.8.99.200 'freq:cent?'
 *
 *   Select Vector instrument mode:
 *        lanio koala3.lsid.hp.com 'INST VECT'
 *
 *   Turn averaging on, wait for the average, move mkr to peak and query x pos:
 *        lanio hp89440 'AVER:STAT ON; :abort;*wai; :calc:mark:max; x?'
 *
 ****************************************************************************
 *
 * To compile on hpux:
 *  cc -Aa -o lanio lanio.c
 *
*/

#define _HPUX_SOURCE

#include <stdio.h>         /* for fprintf and NULL  */
#include <string.h>        /* for memcpy and memset */
#include <errno.h>         /* for strerror          */

#include <sys/socket.h>    /* for connect and socket*/
#include <netinet/in.h>    /* for sockaddr_in       */
#include <netdb.h>         /* for gethostbyname     */
```

```
/***************************************************************************
 *
 > $Function: openSocket$
 *
 * $Description:  open a TCP/IP socket connection to the instrument $
 *
 * $Parameters:  $
 *       (const char *) hostname . . . . Network name of instrument.
 *                                       This can be in dotted decimal notation.
 *       (int) portNumber  . . . . . . . The TCP/IP port to talk to.
 *
 * $Return:     (int)  . . . . . . . . . A file descriptor similar to open(1).$
 *
 * $Errors:     returns -1 if anything goes wrong $
 *
 ***************************************************************************/
int openSocket(const char *hostname, int portNumber)
{
    struct hostent *hostPtr;
    struct sockaddr_in peeraddr_in;
    int s;

    memset(&peeraddr_in, 0, sizeof(struct sockaddr_in));

    /***********************************************/
    /* map the desired host name to internal form. */
    /***********************************************/
    hostPtr = gethostbyname(hostname);
    if (hostPtr == NULL)
    {
        fprintf(stderr,"unable to resolve hostname ‘%s’\n", hostname);
        return -1;
    }

    /*******************/
    /* create a socket */
    /*******************/
    s = socket(AF_INET, SOCK_STREAM, 0);
    if (s == -1)
    {
        fprintf(stderr,"unable to create socket to ‘%s’: %s\n",
                hostname, strerror(errno));
        return -1;
    }

    memcpy(&peeraddr_in.sin_addr.s_addr, hostPtr->h_addr, hostPtr->h_length);
    peeraddr_in.sin_family = AF_INET;
    peeraddr_in.sin_port = htons(portNumber);

    if (connect(s, (char*) &peeraddr_in, sizeof(struct sockaddr_in)) == -1)
    {
        fprintf(stderr,"unable to create socket to ‘%s’: %s\n",
                hostname, strerror(errno));
        return -1;
    }

    return s;
}
```

```
/***************************************************************************
 *
 > $Function: queryInstrument$
 *
 * $Description:  send a SCPI command to the instrument and return a response.$
 *
 * $Parameters:  $
 *       (FILE *) . . . . . . . . . file pointer associated with TCP/IP socket.
 *       (const char *command)  . . SCPI command string.
 *       (char *result) . . . . . . where to put the result.
 *       (size_t) maxLength . . . . maximum size of result array in bytes.
 * $Return:  (char *) . . . . . . . a pointer to the result string.
 *
 * $Errors:   returns NULL if anything goes wrong $
 *
 ***************************************************************************/
char *queryInstrument(FILE *file,
                      const char *command, char *result, size_t maxLength)
{
    int length;

    if (fprintf(file,"%s\n", command) < 0) return NULL;
    fflush(file);

    if (fgets(result, maxLength, file) == NULL) return NULL;

    /* REMOVE TRAILING NEWLINE, IF PRESENT */
    length = strlen(result);
    if (result[length-1] == ‘\n’) result[length-1] = ‘\0’;

    return result;
}
```

```
/***************************************************************************
 *
 > $Function: commandInstrument$
 *
 * $Description:  send a SCPI command to the instrument.$
 *
 * $Parameters:  $
 *       (FILE *) . . . . . . . . . file pointer associated with TCP/IP socket.
 *       (const char *command)  . . SCPI command string.
 * $Return:  (char *) . . . . . . . a pointer to the result string.
 *
 * $Errors:   returns 0 if fprintf fails $
 *
 ***************************************************************************/
int commandInstrument(FILE *file,
                      const char *command)
{
    if (fprintf(file,"%s\n", command) < 0) return 0;
    fflush(file);

    return 1;
}

int main(int argc, char *argv[])
{
    int instrument;
    FILE *instFile;
    char charBuf[256];
    char *command;
    char *destination;

    if (argc < 3)
    {
        fprintf(stderr,"Usage: %s <hostname> <command>\n", argv[0]);
        return 1;
    }

    destination = argv[1];
    command = argv[2];

    /*********************************************/
    /* open a socket connection to the instrument */
    /*********************************************/
    instrument = openSocket(destination, 5025);
    if (instrument == -1) return 1;

    /*********************************************************/
    /* create a FILE * pointer so we can use fprintf and friends */
    /*********************************************************/
    instFile = fdopen(instrument,"r+");
    if (instFile == NULL)
    {
        fprintf(stderr, “Unable to create FILE * structure : %s\n”,
                strerror(errno));
        return 1;
    }
```

```
    /******************************************************/
    /* if the command has a '?' in it, use queryInstrument. */
    /* otherwise, simply send the command.                  */
    /******************************************************/
    if (strchr(command,'?'))
    {
        printf("%s\n", queryInstrument(instFile,command,
                                       charBuf,
                                       sizeof(charBuf)));
    }
    else
    {
        commandInstrument(instFile, command);
    }

    return 0;
}
```

INDEX

A

B

C

D

E

F

G

H

I

K

L

M

N

O

Q

R

S

T

U

V

W

X

Y

Z

Need Assistance?

If you need assistance, contact your nearest Agilent Technologies Sales and Service Office listed in the Agilent Catalog. You can also find a list of local service representatives on the Web at: ***http://www.agilent-tech.com/services/English/index.html*** or contact your nearest regional office listed below.

If you are contacting Agilent Technologies about a problem with your Agilent 89400A Vector Signal Analyzer, please provide the following information:

- ❑ Model number: Agilent 89400A
- ❑ Serial number:
- ❑ 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?

You may find the serial number and options from the front panel of your analyzer by executing the following:

Press [**System Utility**], [more], [serial number].

Press [**System Utility**], [options setup].

If you do not have access to the Internet, one of these centers can direct you to your nearest representative:

United States	Test and Measurement Call Center (800) 452-4844 (Toll free in US)
Canada	(905) 206-4725
Europe	(31 20) 547 9900
Japan	Measurement Assistance Center (81) 426 56 7832 (81) 426 56 7840 (FAX)
Latin America	(305) 267 4245 (305) 267 4288 (FAX)
Australia/New Zealand	1 800 629 485 (Australia) 0800 738 378 (New Zealand)
Asia-Pacific	(852) 2599 7777 (FAX) (852) 2506 9285