

# PART TWO

# COMMANDS

**Part Two describes the commands and queries you will need to operate your instrument remotely.**

**Important Note for users of other LeCroy DSOs with existing remote control software.**

**Trace labels** TA, TB, TC and TD have been replaced by traces F1, F2, F3 and F4 respectively. Existing software that includes the old trace labels will work with the WaveMaster scopes, but new software should use the new labels unless it will also be used on an earlier DSO. In addition to these four traces, the instrument includes F5, F6, F7, and F8. Responses to queries will always use the new labels, even if the old labels are used, however. All eight traces F1 through F8 are completely equivalent in ability to perform zooms or processing.

**Parameter labels** have also changed. These are now P1 through P8, but the older labels Cust1 through Cust5, etc. will still work. Responses to queries will always use the new labels, even if the old labels are used.

## **PART TWO: COMMANDS**

**In this part of the manual, you will find the commands and queries to**

- ***Run the instrument remotely.***

## Use Commands and Queries

This part of the manual describes the remote control commands and queries recognized by the instrument. All of them can be executed in either local or remote state.

The commands and queries are listed in alphabetical order according to the long form of their name. For example, the description of `ATTENUATION`, whose short form is `ATTN`, is listed before that of `AUTO SETUP`, whose short form is `ASET`. Each command or query description starts on a new page. The name (header) is given in both long and short form at the top of the first page of each description.

Queries perform actions such as obtaining information. They are recognized by `?` following their headers. Many commands can be used as queries simply by adding the question mark. In order to find out the correct form of a command, it is very useful to set up the scope manually to the exact condition that you require, and then to send a query which corresponds to the required command. The reply from the scope can be copied into your program as a command.

A brief explanation of the operation performed by the command or query is followed by the formal syntax, with the full-name header given in lowercase characters and the short form derived from it in uppercase characters (e.g., `DOT_JOIN` gives `DTJN`). Where applicable, the syntax of the query is given with the format of its response. For each command, a short GPIB example illustrating a typical use is also provided. The GPIB examples assume that the controller is equipped with a National Instruments interface board, which calls to the related interface subroutines in BASIC, though the principles will be similar in other languages. The device name of the oscilloscope is defined as **SCOPE%** in the examples, but you can substitute any valid device name.

Use the two tables that precede the descriptions to quickly find a command or query. The first of these lists the commands and queries in alphabetical order according to their long form. The second table groups them according to the subsystem or category they belong to.

### COMMAND NOTATION

The following notation is used in the commands:

- `<`      Angular brackets enclose words that are used as placeholders, of which there are two types: the header
- `>`      path and the data parameter of a command.
- `:`      A colon followed by an equals sign separates a placeholder from the description of the type and range
- `=`      of values that can be used in a command instead of the placeholder.
- `{`  
`}`      Braces enclose a list of choices, one of which must be made.
- `[`      Square brackets enclose optional items.

## PART TWO: COMMANDS

---

- 1
- ... An ellipsis indicates that the items left and right of it can be repeated any number of times.

**Example:** consider the syntax notation for the command to set the vertical input sensitivity:

1. `<channel> : VOLT_DIV <v_gain>`
2. `<channel> := {C1, C2}`
3. `<v_gain> := 5.0 mV to 2.5 V`

The first line shows the formal appearance of the command: `<channel>` denotes the placeholder for the header path; `<v_gain>` is the placeholder for the vertical gain value.

The second line indicates that either `C1` or `C2` must be chosen for the header path.

The third line means that the actual vertical gain can be set to any value from 5 mV to 2.5 V.

## Table of Commands and Queries — By Short Form

SHORT FORM	LONG FORM	SUBSYSTEM (CATEGORY)	WHAT THE COMMAND OR QUERY DOES
ACAL	AUTO_CALIBRATE	MISCELLANEOUS	Enables and disables automatic calibration
ALST?	ALL_STATUS?	STATUS	Reads and clears the contents of all status registers.
ARM	ARM_ACQUISITION	ACQUISITION	Changes acquisition state from “stopped” to “single.”
ASET	AUTO_SETUP	ACQUISITION	Adjusts vertical, timebase and trigger parameters.
ATTN	ATTENUATION	ACQUISITION	Selects the vertical attenuation factor of the probe.
BUZZ	BUZZER	MISCELLANEOUS	Controls the buzzer in the DSO.
BWL	BANDWIDTH_LIMIT	ACQUISITION	Enables/disables bandwidth-limiting low-pass filter.
*CAL?	*CAL?	MISCELLANEOUS	Performs a complete internal calibration of the DSO.
CFMT	COMM_FORMAT	COMMUNICATION	Selects the format for sending waveform data.
CHDR	COMM_HEADER	COMMUNICATION	Controls formatting of query responses.
CHLP	COMM_HELP	COMMUNICATION	Controls operational level of the RC Assistant.
CHL	COMM_HELP_LOG	COMMUNICATION	Returns the contents of the RC Assistant log.
CLM	CLEAR_MEMORY	FUNCTION	Clears the specified memory.
*CLS	*CLS	STATUS	Clears all status data registers.
CLSW	CLEAR_SWEEPS	FUNCTION	Restarts the cumulative processing functions.
CMR?	CMR?	STATUS	Reads and clears the CoMmand error Register (CMR).
COMB	COMBINE_CHANNELS	ACQUISITION	Controls the channel interleaving function.
CORD	COMM_ORDER	COMMUNICATION	Controls the byte order of waveform data transfers.
COUT	CAL_OUTPUT	MISCELLANEOUS	Sets signal type put out at the CAL connector.
CPL	COUPLING	ACQUISITION	Selects the specified input channel's coupling mode.
CRMS	CURSOR_MEASURE	CURSOR	Specifies the type of cursor/parameter measurement.
CRST	CURSOR_SET	CURSOR	Allows positioning of any cursor.
CRVA?	CURSOR_VALUE?	CURSOR	Returns trace values measured by specified cursors.
CRS	CURSORS	CURSOR	Sets the cursor type.
DDR?	DDR?	STATUS	Reads, clears the Device Dependent Register (DDR).
DEF	DEFINE	FUNCTION	Specifies math expression for function evaluation.
DELF	DELETE_FILE	MISCELLANEOUS	Deletes a file from the currently selected directory.
DISP	DISPLAY	DISPLAY	Controls the display screen.
DTJN	DOT_JOIN	DISPLAY	Controls the interpolation lines between data points.
MAIL	EMAIL	MISCELLANEOUS	Sets up email protocol and addresses.
*ESE	*ESE	STATUS	Sets the Standard Event Status Enable register (ESE).
*ESR?	*ESR?	STATUS	Reads, clears the Event Status Register (ESR).
EXR?	EXR?	STATUS	Reads, clears the EXecution error Register (EXR).
FRTR	FORCE_TRIGGER	ACQUISITION	Forces the DSO to make one acquisition.
FRST	FUNCTION_RESET	FUNCTION	Resets a waveform-processing function.
GRID	GRID	DISPLAY	Specifies single-, dual- or quad-mode grid display.
HCSU	HARDCOPY_SETUP	HARD COPY	Configures the hard-copy driver.

SHORT FORM	LONG FORM	SUBSYSTEM (CATEGORY)	WHAT THE COMMAND OR QUERY DOES
HMAG	HOR_MAGNIFY	DISPLAY	Horizontally expands the selected expansion trace.
HPOS	HOR_POSITION	DISPLAY	Horizontally positions intensified zone's center.
*IDN?	*IDN?	MISCELLANEOUS	For identification purposes.
INE	INE	STATUS	Sets the INternal state change Enable register (INE).
INR?	INR?	STATUS	Reads, clears INternal state change Register (INR).
INSP?	INSPECT?	WAVEFORM TRANSFER	Allows acquired waveform parts to be read.
ILVD	INTERLEAVED	ACQUISITION	Enables/disables Random Interleaved Sampling (RIS).
INTS	INTENSITY	DISPLAY	Controls the brightness of the grid.
IST?	IST?	STATUS	Reads the current state of the IEEE 488.
MSG	MESSAGE	DISPLAY	Displays a character string on the DSO screen.
MSIZ	MEMORY_SIZE	ACQUISITION	Selects max. memory length.
OFST	OFFSET	ACQUISITION	Allows output channel vertical offset adjustment.
OFCT	OFFSET_CONSTANT	CURSOR	Sets offset to be fixed in either divisions or volts.
*OPC	*OPC	STATUS	Sets the OPC bit in the Event Status Register (ESR).
*OPT?	*OPT?	MISCELLANEOUS	Identifies oscilloscope options.
PNSU	PANEL_SETUP	SAVE/RECALL	Complements the *SAV/*RST commands.
PARM	PARAMETER	CURSOR	Controls the parameter mode.
PACL	PARAMETER_CLR	CURSOR	Clears all current parameters in Custom, Pass/Fail.
PACU	PARAMETER_CUSTOM	CURSOR	Controls parameters with customizable qualifiers.
PADL	PARAMETER_DELETE	CURSOR	Deletes a specified parameter in Custom, Pass/Fail.
PAST?	PARAMETER_STATISTICS?	CURSOR	Returns parameter statistics results.
PAVA?	PARAMETER_VALUE?	CURSOR	Returns current parameter, mask test values.
PF	PASS_FAIL	CURSOR	Sets up pass fail system.
PFDO	PASS_FAIL_DO	CURSOR	Defines outcome and actions for Pass/Fail
PERS	PERSIST	DISPLAY	Enables or disables the persistence display mode.
PECL	PERSIST_COLOR	DISPLAY	Controls color rendering method of persistence traces.
PECS	PER_CURSOR_SET	CURSOR	Positions one of the six independent cursors.
PELT	PERSIST_LAST	DISPLAY	Shows the last trace drawn in a persistence data map.
PESA	PERSIST_SAT	DISPLAY	Sets the color saturation level in persistence.
PESU	PERSIST_SETUP	DISPLAY	Selects display persistence duration.
*PRE	*PRE	STATUS	Sets the PaRallel poll Enable register (PRE).
*RCL	*RCL	SAVE/RECALL	Recalls one of five non-volatile panel setups.
RCPN	RECALL_PANEL	SAVE/RECALL	Recalls a front panel setup from mass storage.
*RST	*RST	SAVE/RECALL	Initiates a device reset.
*SAV	*SAV	SAVE/RECALL	Stores current state in non-volatile internal memory.
SCLK	SAMPLE_CLOCK	ACQUISITION	Toggles between internal clock and external clock
SCDP	SCREEN_DUMP	HARD COPY	Initiates a screen dump.
SEQ	SEQUENCE	AQUISITION	Controls the sequence mode of acquisition.
*SRE	*SRE	STATUS	Sets the Service Request Enable register (SRE).

## PART TWO: COMMANDS

SHORT FORM	LONG FORM	SUBSYSTEM (CATEGORY)	WHAT THE COMMAND OR QUERY DOES
*STB?	*STB?	STATUS	Reads the contents of the IEEE 488.
STOP	STOP	ACQUISITION	Immediately stops signal acquisition.
STO	STORE	WAVEFORM TRANSFER	Stores a trace in internal memory or mass storage.
STPN	STORE_PANEL	SAVE/RECALL	Stores front panel setup to mass storage.
STST	STORE_SETUP	WAVEFORM TRANSFER	Sets up waveform storage.
TDIV	TIME_DIV	ACQUISITION	Modifies the timebase setting.
TMPL?	TEMPLATE?	WAVEFORM TRANSFER	Produces a complete waveform template copy.
TRA	TRACE	DISPLAY	Enables or disables the display of a trace.
TRFL	TRANSFER_FILE	WAVEFORM TRANSFER	Transfers ASCII files to and from storage media, or between scope and computer.
*TRG	*TRG	ACQUISITION	Executes an ARM command.
TRCP	TRIG_COUPLING	ACQUISITION	Sets the coupling mode of the specified trigger source.
TRDL	TRIG_DELAY	ACQUISITION	Sets the time at which the trigger is to occur.
TRLV	TRIG_LEVEL	ACQUISITION	Adjusts the trigger level of the specified trigger source.
TRMD	TRIG_MODE	ACQUISITION	Specifies the trigger mode.
TRPA	TRIG_PATTERN	ACQUISITION	Defines a trigger pattern.
TRSE	TRIG_SELECT	ACQUISITION	Selects the condition that will trigger acquisition.
TRSL	TRIG_SLOPE	ACQUISITION	Sets the trigger slope of the specified trigger source.
VBS	VBS	AUTOMATION	Sends an automation command
VDIV	VOLT_DIV	ACQUISITION	Sets the vertical sensitivity.
VMAG	VERT_MAGNIFY	DISPLAY	Vertically expands the specified trace.
VPOS	VERT_POSITION	DISPLAY	Adjusts the vertical position of the specified trace.
*WAI	*WAI	STATUS	WAIT to continue - required by the IEEE 488.
WAIT	WAIT	ACQUISITION	Prevents new analysis until current is completed.
WF	WAVEFORM	WAVEFORM TRANSFER	Transfers a waveform from controller to scope.
WFSU	WAVEFORM_SETUP	WAVEFORM TRANSFER	Specifies amount of waveform data to go to controller.

# Table of Commands and Queries — By Subsystem

SHORT FORM	LONG FORM	WHAT THE COMMAND OR QUERY DOES
<b>ACQUISITION — TO CONTROL WAVEFORM CAPTURE</b>		
ARM	ARM_ACQUISITION	Changes acquisition state from “stopped” to “single.”
ASET	AUTO_SETUP	Adjusts vertical, timebase and trigger parameters for signal display.
ATTN	ATTENUATION	Selects the vertical attenuation factor of the probe.
BWL	BANDWIDTH_LIMIT	Enables or disables the bandwidth-limiting low-pass filter.
COMB	COMBINE_CHANNELS	Controls the channel interleaving function.
CPL	COUPLING	Selects the specified input channel's coupling mode.
FRTR	FORCE_TRIGGER	Forces the DSO to make one acquisition.
ILVD	INTERLEAVED	Enables or disables Random Interleaved Sampling (RIS).
MSIZ	MEMORY_SIZE	Allows selection of maximum memory length.
OFST	OFFSET	Allows vertical offset adjustment of the specified input channel.
SCLK	SAMPLE_CLOCK	Toggles between internal clock and external clock.
SEQ	SEQUENCE	Controls the sequence mode of acquisition.
STOP	STOP	Immediately stops signal acquisition.
TDIV	TIME_DIV	Modifies the timebase setting.
*TRG	*TRG	Executes an ARM command.
TRCP	TRIG_COUPLING	Sets the coupling mode of the specified trigger source.
TRDL	TRIG_DELAY	Sets the time at which the trigger is to occur.
TRLV	TRIG_LEVEL	Adjusts the level of the specified trigger source.
TRMD	TRIG_MODE	Specifies Trigger mode.
TRPA	TRIG_PATTERN	Defines a trigger pattern.
TRSE	TRIG_SELECT	Selects the condition that will trigger acquisition.
TRSL	TRIG_SLOPE	Sets the slope of the specified trigger source.
VDIV	VOLT_DIV	Sets the vertical sensitivity in volts/div.
WAIT	WAIT	Prevents new command analysis until current acquisition completion.
<b>AUTOMATION — TO SEND AUTOMATION COMMANDS</b>		
VBS	VBS	Send Automation commands.
<b>COMMUNICATION — TO SET COMMUNICATION CHARACTERISTICS</b>		
CFMT	COMM_FORMAT	Selects the format to be used for sending waveform data.
CHDR	COMM_HEADER	Controls formatting of query responses.
CHLP	COMM_HELP	Controls operational level of the RC Assistant.
CHL	COMM_HELP_LOG	Returns the contents of the RC Assistant log.
CORD	COMM_ORDER	Controls the byte order of waveform data transfers.

## PART TWO: COMMANDS

SHORT FORM	LONG FORM	WHAT THE COMMAND OR QUERY DOES
<b>CURSOR — To PERFORM MEASUREMENTS</b>		
CRMS	CURSOR_MEASURE	Specifies the type of cursor or parameter measurement for display.
CRST	CURSOR_SET	Allows positioning of any cursor.
CRVA?	CURSOR_VALUE?	Returns the values measured by the specified cursors for a given trace.
CRS	CURSORS	Sets the cursor type.
OFCT	OFFSET_CONSTANT	Sets offset to be constant in divisions or volts.
PARM	PARAMETER	Controls the parameter mode.
PACL	PARAMETER_CLR	Clears all current parameters in Custom and Pass/Fail modes.
PACU	PARAMETER_CUSTOM	Controls parameters with customizable qualifiers.
PADL	PARAMETER_DELETE	Deletes a specified parameter in Custom and Pass/Fail modes.
PAST?	PARAMETER_STATISTICS	Returns parameter statistics results.
PAVA?	PARAMETER_VALUE?	Returns current value(s) of parameter(s) and mask tests.
PF	PASS_FAIL	Sets up the Pass / Fail system.
PFDO	PASS_FAIL_DO	Defines outcome and actions for the Pass/Fail system.
PECS	PER_CURSOR_SET	Positions one of the six independent cursors.
<b>DISPLAY — To DISPLAY WAVEFORMS</b>		
DISP	DISPLAY	Controls the oscilloscope display screen.
DTJN	DOT_JOIN	Controls the interpolation lines between data points.
GRID	GRID	Specifies grid display in single, dual or quad mode.
HMAG	HOR_MAGNIFY	Horizontally expands the selected expansion trace.
HPOS	HOR_POSITION	Horizontally positions the intensified zone's center on the source trace.
INTS	INTENSITY	Controls the brightness of the grid on the DSO screen.
MSG	MESSAGE	Displays a string of characters on the DSO screen for a short time.
PERS	PERSIST	Enables or disables the Persistence Display mode.
PECL	PERSIST_COLOR	Controls color rendering method of persistence traces.
PELT	PERSIST_LAST	Shows the last trace drawn in a persistence data map.
PESA	PERSIST_SAT	Sets the color saturation level in persistence.
PESU	PERSIST_SETUP	Selects display persistence duration in Persistence mode.
TRA	TRACE	Enables or disables the display of a trace.
VMAG	VERT_MAGNIFY	Vertically expands the specified trace.
VPOS	VERT_POSITION	Adjusts the vertical position of the specified trace.
<b>FUNCTION — To PERFORM WAVEFORM MATHEMATICAL OPERATIONS</b>		
CLM	CLEAR_MEMORY	Clears the specified memory.
CLSW	CLEAR_SWEEPS	Restarts the cumulative processing functions.
DEF	DEFINE	Specifies math expression for function evaluation.
FRST	FUNCTION_RESET	Resets a waveform processing function.

SHORT FORM	LONG FORM	WHAT THE COMMAND OR QUERY DOES
<b>HARD COPY — To PRINT THE CONTENTS OF THE DISPLAY</b>		
HCSU	HARDCOPY_SETUP	Configures the hard-copy driver.
SCDP	HARDCOPY_SETUPSCREEN_DUMP	Initiates a screen dump.
<b>MISCELLANEOUS</b>		
ACAL	AUTO_CALIBRATE	Enables or disables automatic calibration
BUZZ	BUZZER	Controls the buzzer in the DSO.
*CAL?	*CAL?	Performs a complete internal calibration
COU	CAL_OUTPUT	Sets the type of signal put out at the CAL connector.
DELF	DELETE_FILE	Deletes a file from the currently selected directory.
MAIL	EMAIL	Sets up email proto col and addresses.
*IDN?	*IDN?	Used for identification purposes.
*OPT?	*OPT?	Identifies the installed oscilloscope options.
<b>SAVE/RECALL SETUP — To PRESERVE AND RESTORE FRONT PANEL SETTINGS</b>		
PNSU	PANEL_SETUP	Complements the *SAV/*RST commands.
*RCL	*RCL	Recalls one of five non-volatile panel setups.
RCPN	RECALL_PANEL	Recalls a front panel setup from mass storage.
*RST	*RST	Initiates a device reset.
*SAV	*SAV	Stores the current state in non-volatile internal memory.
STPN	STORE_PANEL	Stores the complete front panel setup on a mass-storage file.
<b>STATUS — To OBTAIN STATUS INFORMATION AND SET UP SERVICE REQUESTS</b>		
ALST?	ALL_STATUS?	Reads and clears the contents of all (but one) of the status registers.
*CLS	*CLS	Clears all the status data registers.
CMR?	CMR?	Reads and clears the contents of the CoMmand error Register (CMR).
DDR?	DDR?	Reads and clears the Device-Dependent error Register (DDR).
*ESE	*ESE	Sets the standard Event Status Enable (ESE) register.
*ESR?	*ESR?	Reads and clears the Event Status Register (ESR).
EXR?	EXR?	Reads and clears the EXecution error Register (EXR).
INE	INE	Sets the INternal state change Enable register (INE).
INR?	INR?	Reads and clears the INternal state change Register (INR).
IST?	IST?	Individual STatus reads the current state of IEEE 488.
*OPC	*OPC	Sets to true the OPC bit (0) in the Event Status Register (ESR).
*PRE	*PRE	Sets the PaRallel poll Enable register (PRE).
*SRE	*SRE	Sets the Service Request Enable register (SRE).
*STB?	*STB?	Reads the contents of IEEE 488.
*WAI	*WAI	WAI to continue — required by IEEE 488.

SHORT FORM	LONG FORM	WHAT THE COMMAND OR QUERY DOES
WAVEFORM TRANSFER — To PRESERVE AND RESTORE WAVEFORMS		
INSP?	INSPECT?	Allows acquired waveform parts to be read.
STO	STORE	Stores a trace in one of the internal memories M1–4 or mass storage.
STST	STORE_SETUP	Sets up waveform storage
TMPL?	TEMPLATE?	Produces a copy of the template describing a complete waveform.
TRFL	TRANSFER_FILE	Transfers ASCII files to and from storage media, or between scope and computer.
WF	WAVEFORM	Transfers a waveform from the controller to the oscilloscope.
WFSU	WAVEFORM_SETUP	Specifies amount of waveform data for transmission to controller.

### MISCELLANEOUS

### AUTO\_CALIBRATE, ACAL Command/Query

#### DESCRIPTION

The AUTO\_CALIBRATE command, ACAL ON, is used to enable or disable the automatic calibration of your WaveMaster oscilloscope. At power-up, auto-calibration is turned ON, i.e. all input channels are periodically calibrated for the current input amplifier and timebase settings, whether the DSO is adjusted or not.

NOTE that whenever you adjust a gain or offset control, the DSO will perform a calibration. This action occurs whatever the current state of ACAL, and it does not change the state of ACAL.

The automatic calibration can be disabled by issuing the command ACAL OFF. Whenever convenient, a \*CAL? query may be issued to fully calibrate the oscilloscope. When the oscilloscope is returned to local control, the periodic calibrations are resumed, if the last ACAL value was ON. The command \*CAL? has no effect on the ACAL status.

The response to the AUTO\_CALIBRATE? query indicates whether auto-calibration is enabled or disabled.

#### COMMAND SYNTAX

```
Auto_CALibrate <state>  
<state> := {ON, OFF}
```

#### QUERY SYNTAX

```
Auto_CALibrate?
```

#### RESPONSE FORMAT

```
Auto_CALibrate <state>
```

#### EXAMPLE (GPIB)

The following disables auto-calibration:

```
CMD$="ACAL OFF": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

```
*CAL?
```

### **STATUS**

**ALL\_STATUS?, ALST?**

Query

#### **DESCRIPTION**

The ALL\_STATUS? query reads and clears the contents of all status registers: STB, ESR, INR, DDR, CMR, EXR and URR except for the MAV bit (bit 6) of the STB register. For an interpretation of the contents of each register, refer to the appropriate status register.

The query is useful in a complete overview of the state of your oscilloscope.

#### **QUERY SYNTAX**

AlL\_Status?

#### **RESPONSE FORMAT**

AlL\_Status STB,<value>,ESR,<value>,INR,<value>,  
DDR,<value>,CMR,<value>,EXR,<value>,URR,<value>

<value> : = 0 to 65535

#### **EXAMPLE (GPIB)**

The following reads the contents of all the status registers:

```
CMD$="ALST?": CALL IBWRT(SCOPE%,CMD$):  
CALL IBRD(SCOPE%,RSP$): PRINT RSP$
```

Response message:

```
ALST  
F2,000000,ESR,000052,INR,000005,DDR,000000,  
CMR,000004,EXR,000024,URR,000000
```

#### **RELATED COMMANDS**

\*CLS, CMR?, DDR?, \*ESR?, EXR?, \*STB?, URR?

### **ACQUISITION**

**ARM\_ACQUISITION, ARM**  
Command

#### **DESCRIPTION**

The ARM\_ACQUISITION command enables the signal acquisition process by changing the acquisition state (trigger mode) from “stopped” to “single”.

#### **COMMAND SYNTAX**

ARM\_acquisition

#### **EXAMPLE**

The following enables signal acquisition:

```
CMD$="ARM": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

STOP, \*TRG, TRIG\_MODE, WAIT, FRTR

### ACQUISITION

### ATTENUATION, ATTN Command/Query

#### DESCRIPTION

The ATTENUATION command selects the vertical attenuation factor of the probe. Values of 1, 2, 5, 10, 20, 25, 50, 100, 200, 500, 1000 or 10000 can be specified.

The ATTENUATION? query returns the attenuation factor of the specified channel.

#### COMMAND SYNTAX

```
<channel>: ATTeNuatiOn <attenuation>  
<channel> := {C1, C2, C3, C4, EX, EX10}  
<attenuation>:= {1, 2, 5, 10, 20, 25, 50, 100, 200,  
500, 1000, 10000}
```

#### QUERY SYNTAX

```
<channel> : ATTeNuatiOn?
```

#### RESPONSE FORMAT

```
<channel> : ATTeNuatiOn <attenuation>
```

#### AVAILABILITY

```
<channel> : {C3, C4} available only on four-channel oscilloscopes.
```

#### EXAMPLE (GPIB)

The following sets to 100 the attenuation factor of Channel 1:

```
CMD$="C1:ATTN 100": CALL IBWRT(SCOPE%,CMD$)
```

### ACQUISITION

**AUTO\_SETUP, ASET**  
Command

#### DESCRIPTION

The `AUTO_SETUP` command attempts to display the input signal(s) by adjusting the vertical, timebase and trigger parameters. `AUTO_SETUP` operates only on the channels whose traces are currently turned on. If no traces are turned on, `AUTO_SETUP` operates on all channels and turns on all of the traces.

If signals are detected on several channels, the lowest numbered channel with a signal determines the selection of the timebase and trigger source.

If only one input channel is turned on, the timebase will be adjusted for that channel.

The `<channel> : AUTO_SETUP FIND` command adjusts gain and offset only for the specified channel.

#### COMMAND SYNTAX

`<channel> : Auto_SETUP [FIND]`

`<channel> := {C1, C2, C3, C4}`

If the `FIND` keyword is present, gain and offset adjustments will be performed only on the specified channel. In this case, if no `<channel>` prefix is added, then an auto-setup will be performed on the channel used on the last `ASET FIND` remote command. In the absence of the `FIND` keyword, the normal auto-setup will be performed, regardless of the `<channel>` prefix.

#### AVAILABILITY

`<channel> := {C3, C4}` only on four-channel oscilloscopes.

#### EXAMPLE

The following instructs the oscilloscope to perform an auto-setup:

```
CMD$="ASET": CALL IBWRT(SCOPE%,CMD$)
```

### ACQUISITION

**BANDWIDTH\_LIMIT, BWL**  
Command/Query

#### DESCRIPTION

BANDWIDTH\_LIMIT enables or disables the bandwidth-limiting low-pass filter. The command is used to set the bandwidth individually for each channel. The response to the BANDWIDTH\_LIMIT? query indicates whether the bandwidth filters are on or off.

#### COMMAND SYNTAX

```
BandWidth_Limit <mode>
BandWidth_Limit <channel>,<mode>[,<channel>,<mode>
[,<channel>,<mode>[,<channel>,<mode>]]]

<mode> : = {OFF, ON, 200MHZ, 1GHZ, 3GHZ*, 4GHZ*}
<channel> : = {C1, C2, C3, C4}
```

#### QUERY SYNTAX

```
BandWidth_Limit?
```

#### RESPONSE FORMAT

```
BandWidth_Limit <mode>

If at least two channels have their bandwidth limit filters set
differently from one another, the response is:

BandWidth_Limit <channel>,<mode>[,<channel>,<mode>
[,<channel>,<mode>[,<channel>,<mode>]]]
```

#### AVAILABILITY

\* Available only on 5 GHz and 6 GHz instruments, DDA-5005, and SDA

#### PIB EXAMPLE

The following asserts bandwidth limit on channel 1:

```
CMD$="BWL C1,ON": CALL IBWRT(SCOPE%,CMD$)
```

### MISCELLANEOUS

**BUZZER, BUZZ**  
Command

#### DESCRIPTION

The buzzer command controls the built-in buzzer. The buzzer can be activated for short beeps using BEEP,. The value ON has the same effect as BEEP, unlike the behavior with earlier DSOs. ON and OFF are accepted only for compatibility. OFF has no effect.

#### COMMAND SYNTAX

BUZZer <state>

<state>:= {BEEP, ON, OFF}

#### EXAMPLE (GPIB)

Sending the following will cause the DSO to sound two short tones:

```
CMD$="BUZZ BEEP;BUZZ BEEP":  
CALL IBWRT(SCOPE%<CMD$)
```

### MISCELLANEOUS

#### **\*CAL?**

Query

#### DESCRIPTION

The \*CAL? query causes the oscilloscope to perform an internal self-calibration and generates a response that indicates whether or not your oscilloscope completed the calibration without error. This internal calibration sequence is the same as that which occurs at power-up. At the end of the calibration, after the response has indicated how the calibration terminated, the oscilloscope returns to the state it was in just prior to the calibration cycle. This includes the ACAL status, which is not affected by the use of \*CAL?, and \*CAL? May be used whether ACAL has been set on or off.

#### QUERY SYNTAX

\*CAL?

#### RESPONSE FORMAT

\*CAL <diagnostics>  
<diagnostics> : = 0 or other  
0 = Calibration successful

#### EXAMPLE (GPIB)

The following forces a self-calibration:

```
CMD$="*CAL?": CALL IBWRT(SCOPE%,CMD$):
```

```
CALL IBRD(SCOPE%,RD$): PRINT RD$
```

Response message (if no failure): \*CAL 0

#### RELATED COMMANDS

AUTO\_CALIBRATE

### MISCELLANEOUS

**CAL\_OUTPUT, COUT**  
Command/Query

#### DESCRIPTION

The CAL\_OUTPUT command is used to set the type of signal put out at the instrument front panel CAL BNC connector.

#### COMMAND SYNTAX

Cal\_OUTput <mode>[, <level>[, <rate>]]

Cal\_OUTput PULSE[, <width>]

<mode> := {OFF, CALSQ, PF, TRIG, LEVEL, PULSE, TRDY}

<level> := 5 mV to 1.00 V into 1 M $\Omega$

<rate> := 5 Hz to 5 MHz.

#### QUERY SYNTAX

Cal\_OUTput?

#### RESPONSE FORMAT

Cal\_OUTput <mode>,<level>[,<rate>]

#### EXAMPLE (GPIB)

The following sets the calibration signal to give a 0 to 0.2 volt 10 kHz square wave:

```
CMD$="COUT CALSQ,0.2 V,10 kHz":
```

```
CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

PASS\_FAIL\_DO

#### ADDITIONAL INFORMATION

NOTATION	
CALSQ	Provides a square signal
LEVEL	Provides a DC signal at the requested level
OFF	Provides no signal (ground level)
PF	Pass/Fail mode
PULSE	Provides a single pulse
TRIG	Trigger Out mode

### **FUNCTION**

**CLEAR\_MEMORY, CLM**  
Command

### **DESCRIPTION**

The CLEAR\_MEMORY command clears the specified memory. Data previously stored in this memory are erased and memory space is returned to the free memory pool.

### **COMMAND SYNTAX**

CLear\_Memory < memory>  
<memory> := {M1, M2, M3, M4}

### **EXAMPLE (GPIB)**

The following clears the memory M2.

```
CMD$="CLM M2": CALL IBWRT(SCOPE%,CMD$)
```

### **RELATED COMMANDS**

STORE

FUNCTION	CLEAR_SWEEPS, CLSW Command
DESCRIPTION	The CLEAR_SWEEPS command restarts the cumulative processing functions: summed or continuous average, extrema, FFT power average, histogram, pulse parameter statistics, Pass/Fail counters, and persistence.
COMMAND SYNTAX	CLear SWeeps
EXAMPLE (GPIB)	The following example will restart the cumulative processing: CMD\$="CLSW": CALL IBWRT(SCOPE%,CMD\$)
RELATED COMMANDS	INR

### **STATUS**

**\*CLS**  
Command

#### **DESCRIPTION**

The \*CLS command clears all status data registers.

#### **COMMAND SYNTAX**

\*CLS

#### **EXAMPLE (GPIB)**

The following causes all the status data registers to be cleared:

```
CMD$="*CLS": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

ALL\_STATUS, CMR, DDR, \*ESR, EXR, \*STB, URR

**STATUS**

**CMR?**  
Query

**DESCRIPTION**

The `CMR?` query reads and clears the contents of the CoMmand error Register (CMR) — see table next page — which specifies the last syntax error type detected by your oscilloscope.

**QUERY SYNTAX**

`CMR?`

**RESPONSE FORMAT**

`CMR <value>`  
`<value> : = 0 to 13`

**EXAMPLE (GPIB)**

The following reads the contents of the CMR register:  
`CMD$="CMR?" : CALL IBWRT(SCOPE%,CMD$) :`  
`CALL IBRD(SCOPE%,RSP$) : PRINT RSP$`

**RELATED COMMANDS**

`ALL_STATUS?` , `*CLS`

**ADDITIONAL INFORMATION**

<b>COMMAND ERROR STATUS REGISTER STRUCTURE (CMR)</b>	
<b>Value</b>	<b>Description</b>
1	Unrecognized command/query header
2	Illegal header path
3	Illegal number
4	Illegal number suffix
5	Unrecognized keyword
6	String error
7	GET embedded in another message
10	Arbitrary data block expected
11	Non-digit character in byte count field of arbitrary data block
12	EOI detected during definite length data block transfer
13	Extra bytes detected during definite length data block transfer

### ACQUISITION

### COMBINE\_CHANNELS, COMB Command/Query

#### DESCRIPTION

The COMBINE\_CHANNELS command controls the channel interleaving function of the acquisition system. The COMBINE\_CHANNELS? query returns the channel interleaving function's current status.

#### COMMAND SYNTAX

COMBine\_channels <state>  
<state> := {1, 2, 4, AUTO}

#### QUERY SYNTAX

COMBine\_channels?

#### RESPONSE FORMAT

COMB <state>

#### EXAMPLE (GPIB)

The following instruction engages channel interleaving between Channels 1 and 2, and Channels 3 and 4:

CMD\$="COMB 2": CALL IBWRT(SCOPE%,CMD\$)

The following instruction sets Auto-Combine mode:

CMD\$="COMB AUTO": CALL IBWRT(SCOPE%,CMD\$)

#### RELATED COMMANDS

TDIV

### COMMUNICATION

**COMM\_FORMAT, CFMT**  
Command/Query

#### DESCRIPTION

The COMM\_FORMAT command selects the format the oscilloscope uses to send waveform data. The available options allow the block format, the data type and the encoding mode to be modified from the default settings. Note that some of the settings available for existing DSOs are not available in your instrument.

The COMM\_FORMAT? query returns the currently selected waveform data format.

#### COMMAND SYNTAX

```
Comm_FoRMaT <block_format> , <data_type> , <encoding>
<block_format> := {DEF9}
<data_type> := {BYTE, WORD}
<encoding> := {BIN}
```

Initial settings (i.e. after power-on) are:

For GPIB and LAN: DEF9, WORD, BIN

#### QUERY SYNTAX

```
Comm_FoRMaT?
```

#### RESPONSE FORMAT

```
Comm_FoRMaT <block_format>,<data_type>,<encoding>
```

#### EXAMPLE (GPIB)

The following redefines the transmission format of waveform data. The data will be transmitted as a block of definite length. Data will be coded in binary and represented as eight-bit integers.

```
CMD$="CFMT DEF9,BYTE,BIN": CALL
IBWRT(SCOPE%,CMD$)
```

### ADDITIONAL INFORMATION

#### Block Format

DEF9:

Uses the IEEE 488.2 definite length arbitrary block response data format. The digit 9 indicates that the byte count consists of 9 digits. The data block directly follows the byte count field.

For example, a data block consisting of three data bytes would be

sent as:

WF DAT1 , #9000000003<DAB><DAB><DAB>

where <DAB> represents an eight-bit binary data byte.

A <NL^END> (new line with EOI) signifies that block transmission has ended.

The same data bytes as above would be sent as:

WF DAT1 , #0<DAB><DAB><DAB><NL^END>

**NOTE: The format OFF does not conform to the IEEE 488.2 standard and is only provided for special applications where the absolute minimum of data transfer may be important.**

Data Type

BYTE:

Transmits the waveform data as eight-bit signed integers (one byte).

WORD:

Transmits the waveform data as 16-bit signed integers (two bytes).

**NOTE: The data type BYTE transmits only the high-order bits of the internal 16-bit representation. The precision contained in the low-order bits is lost.**

Encoding

BIN:

Binary encoding

RELATED COMMANDS

WAVEFORM

## COMMUNICATION

**COMM\_HEADER, CHDR**  
Command/Query

### DESCRIPTION

The COMM\_HEADER command controls the way the oscilloscope formats responses to queries. There are three response formats: LONG, in which responses start with the long form of the header word; SHORT, where responses start with the short form of the header word; and OFF, for which headers are omitted from the response and suffix units in numbers are suppressed.

Unless you request otherwise, the SHORT response format is used.

**NOTE:** The default format, i.e. that just after power-on, is **SHORT**.

- This command does not affect the interpretation of messages sent to the oscilloscope. Headers can be sent in their long or short form regardless of the COMM\_HEADER setting.
- Querying the vertical sensitivity of Channel 1 may result in one of the following responses:

COMM_HEADER	RESPONSE
LONG	C1:VOLT_DIV 200E-3 V
SHORT	C1:VDIV 200E-3 V
OFF	200E-3

### COMMAND SYNTAX

Comm\_HeaDeR <mode>

<mode> := {SHORT, LONG, OFF}

### QUERY SYNTAX

Comm\_HeaDeR?

### RESPONSE FORMAT

Comm\_HeaDeR <mode>

### EXAMPLE (GPIB)

The following code sets the response header format to SHORT:  
 CMD\$="CHDR SHORT": CALL IBWRT(SCOPE%, CMD\$)

### RELATED COMMANDS

COMM\_HELP\_LOG

COMMUNICATION

COMM\_HELP, CHLP  
Command/Query

DESCRIPTION

The COMM\_HELP command controls the level of operation of the diagnostics utility Remote Control, which assists in debugging remote control programs. Selected using your instrument's front panel (see the Operator's Manual), Remote Control Assistant can log all message transactions occurring between the external controller and the oscilloscope. You can view the log at any time on-screen and can choose from four levels:

OFF	Don't assist at all.
EO	Log detected Errors Only (default after power-on).
FD	Log the Full Dialog between the controller and the oscilloscope.

COMMAND SYNTAX

Comm\_HeLP <level>,<resetAtPowerOn>  
<level> : = { OFF, EO, FD, RS, }  
<resetAtPowerOn> : = { NO , YES }

The default <level> is EO.

If <resetAtPowerOn> is set to YES (default), at power-on, the logging level is set to EO and the log is cleared. If set to NO, the user-set logging level is preserved and the log is not cleared.

**Note:** Setting CHLP EO, NO is useful in logging command sequences that include rebooting the oscilloscope.

QUERY SYNTAX

Comm\_HeLP?

RESPONSE FORMAT

Comm\_HeLP <level>,<resetAtPowerOn>

EXAMPLE (GPIB)

After sending this command, all the following commands and responses will be logged:

CMD\$="CHLP FD" : CALL IBWRT ( SCOPE% , CMD\$ )

RELATED COMMANDS

COMM\_HELP\_LOG

### COMMUNICATION

COMM\_HELP\_LOG?, CHL?  
Query

### DESCRIPTION

The COMM\_HELP\_LOG query returns the current contents of the log generated by the Remote Control Assistant (see CHLP description). If the optional parameter CLR is specified, the log will be cleared after the transmission. Otherwise, it will be kept.

### QUERY SYNTAX

Comm\_HeLP\_Log? [CLR]

### RESPONSE FORMAT

Comm\_Help\_Log <string containing the logged text>

### EXAMPLE (GPIB)

The following reads the remote control log and prints it:

```
CMD$="CHL?" : CALL IBWRT( SCOPE% , CMD$ ) PRINT
```

### RELATED COMMANDS

COMM\_HELP

COMMUNICATION

COMM\_ORDER, CORD  
Command/Query

DESCRIPTION

The COMM\_ORDER command controls the byte order of waveform data transfers. Waveform data can be sent with the most significant byte (MSB) or the least significant byte (LSB) in the first position. The default mode is to send the MSB first. COMM\_ORDER applies equally to the waveform's descriptor and time blocks. In the descriptor some values are 16 bits long ("word"), 32 bits long ("long" or "float"), or 64 bits long ("double"). In the time block all values are floating values, i.e. 32 bits long. When COMM\_ORDER HI is specified, the MSB is sent first; when COMM\_ORDER LO is specified, the LSB is sent first.

The COMM\_ORDER? query returns the byte transmission order in current use.

COMMAND SYNTAX

Comm\_ORDeR <mode>  
<mode> := {HI, LO}

**NOTE: The initial mode, i.e. the mode after power-on, is HI.**

QUERY SYNTAX

Comm\_ORDeR?

RESPONSE FORMAT

Comm\_ORDeR <mode>

EXAMPLE

The order of transmission of waveform data depends on the data type. The following table illustrates the different possibilities:

TYPE	CORD HI	CORD LO
<b>Word</b>	<MSB><LSB>	<LSB><MSB>
<b>Long or Float</b>	<MSB><byte2><byte3><LSB>	<LSB><byte3><byte2><MSB>
<b>Double</b>	<MSB><byte2>...<byte7><LSB>	<LSB><byte7>...<byte2><MSB>

**RELATED COMMANDS**

WAVEFORM

### ACQUISITION

### COUPLING, CPL Command/Query

#### DESCRIPTION

The COUPLING command selects the coupling mode of the specified input channel.

The COUPLING? query returns the coupling mode of the specified channel.

#### COMMAND SYNTAX

`<channel>:CouPLing <coupling>`

`<channel> := {C1, C2, C3, C4, EX, EX10, ETM10}`

`<coupling> := {A1M*, D1M*, D50, GND}`

#### QUERY SYNTAX

`<channel>:CouPLing?`

#### RESPONSE FORMAT

`<channel>:CouPLing <coupling>`

`<coupling> := {A1M, D1M, D50, GND, OVL}`

`<coupling>: OVL` is returned in the event of signal overload while in DC 50  $\Omega$  coupling. In this condition, the oscilloscope will disconnect the input.

#### AVAILABILITY

`<channel> := {C3, C4}` only on four-channel instruments.

\* only for instruments with a probe

#### EXAMPLE (GPIB)

The following sets the coupling of Channel 2 to 50  $\Omega$  DC:

`CMD$="C2:CPL D50": CALL IBWRT(SCOPE%,CMD$)`

## CURSOR

## CURSOR\_MEASURE, CRMS

Command/Query

### DESCRIPTION

The CURSOR\_MEASURE command specifies the type of cursor or parameter measurement to be displayed, and is the main command for displaying parameters and Pass/Fail.

Note that PARAM and SHOW DASH and LIST, used in some earlier LeCroy DSOs, are NOT available in X-Stream DSOs.

The CURSOR\_MEASURE? query indicates which cursors or parameter measurements are currently displayed.

NOTATION	
CUST [ , STAT ]	custom parameters
FAIL	Pass/Fail: fail
HABS	horizontal absolute cursors
HPAR [ , STAT ]	standard time parameters
HREL [ , ABS , DELTA , SLOPE ]	horizontal relative cursors
OFF	cursors and parameters off
PASS	Pass/Fail: pass
VABS	vertical absolute cursors
VPAR [ , STAT ]	standard voltage parameters
VREL [ , ABS , DELTA ]	vertical relative cursors

### COMMAND SYNTAX

CuRsor\_MeaSure <mode>[,<submode>]

<mode> := {CUST, FAIL, HABS, HPAR, HREL, OFF, PASS, VABS, VPAR, VREL}

<submode> := {STAT, ABS, DELTA, SLOPE}

**NOTE:** The keyword *STAT* is optional with modes *CUST*, *HPAR*, and *VPAR*. If present, *STAT* turns parameter statistics on. Absence of *STAT* turns parameter statistics off.

The keywords *ABS*, *DELTA*, *SLOPE* are optional with mode *HREL*. If it is present, *ABS* chooses absolute amplitude reading of relative cursors. Absence of keyword selects relative, *DELTA*, amplitude reading of relative cursors. Keywords *ABS* and *DELTA* are used with *VREL*; *ABS* selects absolute amplitude, absence selects *DELTA*..

### QUERY SYNTAX

CuRsor\_MeaSure?

### RESPONSE FORMAT

CuRsor\_MeaSure <mode>

### EXAMPLE (GPIB)

The following switches on the vertical relative cursors:

```
CMD$="CRMS VREL": CALL IBWRT(SCOPE%,CMD$)
```

The following determines which cursor is currently turned on:

```
CMD$="CRMS?": CALL IBWRT(SCOPE%,CMD$):
```

```
CALL IBRD(SCOPE%,RD$): PRINT RD$
```

Example of response message:

```
CRMS OFF
```

### RELATED COMMANDS

CURSOR\_SET, PARAMETER\_STATISTICS,  
PARAMETER\_VALUE, CURSORS, PARAMETER,  
PASS\_FAIL

### ADDITIONAL INFORMATION

To turn off the cursors, parameter measurements or Pass/Fail tests, use:

```
CURSOR_MEASURE OFF
```

To turn on a cursor display, use one of these five forms:

CURSOR\_MEASURE HABS

CURSOR\_MEASURE HREL

CURSOR\_MEASURE VABS

CURSOR\_MEASURE VREL

CURSOR\_MEASURE FAIL

CURSOR

CURSOR\_SET, CRST  
Command/Query

DESCRIPTION

The CURSOR\_SET command allows you to position any one of the independent cursors at a given screen location.

When setting a cursor position, a trace must be specified, relative to which the cursor will be positioned. This means that the trace must be turned on, a requirement which does not apply to all earlier LeCroy DSOs. Also new to X-Stream DSOs is that PREF and PDIF are not supported, because each parameter has its own independent gate.

**NOTE: To change only the trace without repositioning the cursors, the CURSOR\_SET command can be given with no argument (for example, F2:CRST).**

The CURSOR\_SET? query indicates the current position of the cursor(s). The values returned depend on the grid type selected.

NOTATION			
HABS	horizontal absolute	VABS	vertical absolute
HDIF	horizontal difference	VDIF	vertical difference
HREF	horizontal reference	VREF	vertical reference

COMMAND SYNTAX

<trace> : CuRsor\_SeT <cursor> , <position>[ , <cursor> , <position> , <cursor> , <position>]

<trace> := {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD, C1, C2, C3, C4 , M1 , M2 , M3 , M4} TA through TD are for compatibility with existing software with earlier DSOs. These four mnemonics are not returned by queries.

<cursor> := {HABS, VABS, HREF, HDIF, VREF, VDIF}

<position> := 0 to 10 DIV (horizontal)

<position> := -3.99 to 3.99 DIV (vertical)

**NOTE:** Parameters are grouped in pairs. The first parameter specifies the cursor to be modified and the second one indicates its new value. Parameters can be grouped in any order and restricted to those items to be changed.

**The suffix DIV is optional.**

### QUERY SYNTAX

<trace> : CuRsor\_SeT? <cursor> – only if the cursors are visible.  
No <cursor> implies ALL, even though all are not visible.

<cursor> := {HABS, VABS, HREF, HDIF, VREF, VDIF, ALL}

### RESPONSE FORMAT

<trace> : CuRsor\_SeT  
<cursor> , <position> [ , <cursor> , <position> , ... <cursor> , <position> ]

If <cursor> is not specified, ALL will be assumed. If the position of a cursor cannot be determined in a particular situation, its position will be indicated as UNDEF.

### AVAILABILITY

<trace> : {C3, C4} available only on four-channel oscilloscopes.

### EXAMPLE (GPIB)

The following positions the VREF and VDIF cursors at +3 DIV and –2 DIV respectively, using Trace F1 as a reference:

```
CMD$="F1:CRST VREF,3DIV,VDIF,-2DIV":
```

```
CALL IBWRT(SCOPE%,CMD$)
```

### RELATED COMMANDS

CURSOR\_MEASURE , CURSOR\_VALUE ,  
PARAMETER\_VALUE

CURSOR

CURSOR\_VALUE?, CRVA?

Query

DESCRIPTION

The CURSOR\_VALUE? query returns the values measured by the specified cursors for a given trace. (The PARAMETER\_VALUE? query is used to obtain measured waveform parameter values.)

There are important differences in the function of this command between X-Stream DSOs and earlier LeCroy DSOs.

The keyword ALL should NOT be used, neither should multiple keywords. If they are used, the word UNDEF will be returned.

For the command or query to work, the specified trace MUST be visible. For a query to work, the current cursor mode must be the same as in the query. If it is not the same, UNDEF will be returned.

NOTATION	
HABS	horizontal absolute
HREL	horizontal relative

QUERY SYNTAX

<trace> : CuRsor\_VAlue? [<mode>]  
<trace> := {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD, C1, C2, C3, C4} TA through TD are for compatibility with existing software with earlier DSOs. These four mnemonics are not returned by queries.  
<mode> := {HABS, HREL, VABS, VREL }

RESPONSE FORMAT

<trace> : CuRsor\_VAlue HABS, <abs\_hori>, <abs\_vert>  
<trace> : CuRsor\_VAlue HREL, <delta\_hori>, <delta\_vert>, <abs\_vert\_ref>, <abs\_vert\_dif>, <slope>. The dV/dt value <slope> is displayed in the appropriate trace label box.  
<trace> : CuRsor\_VAlue VABS, <abs\_vert>  
<trace> : CuRsor\_VAlue VREL, <delta\_vert>

For horizontal cursors, both horizontal as well as vertical values are given. For vertical cursors only vertical values are given.

**NOTE:** The cursor mode must be set to the correct type in order to be queried, and the specified trace must be displayed.

### AVAILABILITY

<trace> := {C3, C4} available only on four-channel oscilloscopes.

### EXAMPLE (GPIB)

The following query reads the measured absolute horizontal value of the cross-hair cursor (HABS) on Channel 2:

```
CMD$="C2:CRVA? HABS": CALL
```

```
IBWRT(SCOPE%, CMD$):
```

```
CALL IBRD(SCOPE%, RSP$): PRINT RSP$
```

Response message:

```
C2:CRVA HABS, 34.2E-6 S, 244 E-3 V
```

### ***CURSORS***

### **CURSORS, CRS**

Command/Query

#### **DESCRIPTION**

Sets the type of cursor to be used and the readout. Unlike CRMS, this will not change the state of parameters or pass/fail.

#### **COMMAND SYNTAX**

```
CURSORS <type>[ ,<readout>]  
<type>:=[OFF,HREL,HABS,VREL,VABS]  
<readout>:=[ABS,SLOPE,DELTA] for HREL  
<readout>:=[ABS,DELTA] for VREL
```

#### **QUERY SYNTAX**

```
CURSORS? or CRS?
```

#### **EXAMPLE (GPIB)**

The following instruction sets the cursors to horizontal relative, and readout to the difference between them.

```
CMD$="CRS HREL,DELTA":  
CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

CRMS

<i><b>DATE</b></i>	<b>DATE</b>
<b>DESCRIPTION</b>	<b>Command/Query</b>
	Sets the date and time of the real-time clock in the instrument. Note that you do not need to specify any parameters after the one you want to change, but you <b>MUST</b> include all those before it.
<b>COMMAND SYNTAX A</b>	DATE <day>[ , <month>][ , <year>][ , <hour>][ , <minute>][ , <second>]
<b>COMMAND SYNTAX B</b>	DATE SNTP (to set the date and the time from the internet.
<b>QUERY SYNTAX</b>	DATE?
<b>EXAMPLE (GPIB)</b>	The following instruction sets the date to January 1, 1997, and the time to 1:21:16. CMD\$="DATE 1,JAN,1997,13,21,16" CALL IBWRT(SCOPE%,CMD\$)

STATUS

DDR?  
Query

DESCRIPTION

The DDR? query reads and clears the contents of the Device Dependent or device specific error Register (DDR). In the case of a hardware failure, the DDR register specifies the origin of the failure. The following table gives details.

BIT	BIT VALUE	DESCRIPTION	
15...14		0	Reserved
13	8192	1	Timebase hardware failure detected
12	4096	1	Trigger hardware failure detected
11	2048	1	Channel 4 hardware failure detected
10	1024	1	Channel 3 hardware failure detected
9	512	1	Channel 2 hardware failure detected
8	256	1	Channel 1 hardware failure detected
7	128	1	External input overload condition detected
6...4		0	Reserved
3	8	1	Channel 4 overload condition detected
2	4	1	Channel 3 overload condition detected
1	2	1	Channel 2 overload condition detected
0	1	1	Channel 1 overload condition detected

QUERY SYNTAX

DDR?

RESPONSE FORMAT

DDR <value>  
<value> : = 0 to 65535

## FUNCTION

**DEFINE, DEF**  
Command/Query

### DESCRIPTION

The **DEFINE** command specifies the mathematical expression to be evaluated by a function. This command is used to control all math tools and zoom in the standard oscilloscope as well as those in the Extended Math and WaveAnalyzer options. See the *Operator's Manual* for more about Math Tools.

### COMMAND SYNTAX

<function> : **DEFine** EQN, '<equation>'  
[, <param\_name> , <value> , ...]

**NOTE: Function parameters are grouped in pairs. The first in the pair names the variable to be modified, <param\_name>, while the second one gives the new value to be assigned. Pairs can be given in any order and restricted to the variables to be changed.**

**Space (blank) characters inside equations are optional.**

### QUERY SYNTAX

<function> : **DEFine?**

### RESPONSE FORMAT

<function> : **DEFine** EQN, '<equation>'[, **MAXPTS** , <max\_points>]  
[, **SWEEPS** , <max\_sweeps>][, **WEIGHT** , <weight>][, **BITS** , <bits>]

FUNCTION PARAMETERS		
<param_name>	<value>	Description
<b>BITS</b>	<bits>	Number of ERES bits
<b>CENTER</b>	<center>	Horizontal center position for histogram display.
<b>EQN</b>	'<equation>'	Function equation as defined below
<b>LENGTH</b>	<length>	Number of points to use from first waveform
<b>MAX_EVENTS</b>	<max_values>	Maximum number of values in histogram
<b>MAXBINS</b>	<bins>	Number of bins in histogram
<b>MAXPTS</b>	<max_points>	Maximum number of points to compute

PART TWO: COMMANDS

START	<start>	Starting point in second waveform
SWEEPS	<max_sweeps>	Maximum number of sweeps
UNITS	<units>	Physical units
VERT	<vert_scale>	Vertical scaling type
WEIGHT	<weight>	Continuous Average weight
WIDTH	<width>	Width of histogram display
WINDOW	<window_type>	FFT window function

FUNCTION EQUATIONS AND NAMES		
<b>NOTES:</b> <i>These functions are available according to the options installed in your WaveMaster oscilloscope. It can be very useful to run a function in query mode to determine whether it is available in your oscilloscope, and to determine the current form of the command in your model. The response to any query can be copied straight into a program and used as a command. Functions can also be defined using the VBS command (see Chapter 6).</i>		
<b>Underlined names in the left column of this table represent Boolean expressions, which take the values ON and OFF.</b>		
(<source1>-<source2>	DIFFERENCE	Difference of two waveforms
(-<source>)	INVERSION	Inversion (negation) of waveform
(<source1>*<source2>)	PRODUCT	Product of two waveforms
(<source1>/<source2>)	RATIO	Ratio of two waveforms
(1/<source>)	RECIPROCAL	Reciprocal of a waveform
(<source1>+<source2>)	SUM	Sum of two waveforms
ABS(<source>)		Absolute Value
AVG(<source>,<averagetype>,<sweeps>,< <u>sumhelptext</u> >,< <u>continuoushelptext</u> >)		AVERAGETYPE is SUMMED or CONTINUOUS.
BOXCAR(<source>,<length>)		Boxcar average with specified filter length
CORR(<source1>,<source2>,<corrlength>,<corrstart>)		Correlation of two waveforms with specified correlation length and start point
DERI(<source>,<verscale>,<veroffset>,< <u>enableautosc</u> <u>ale</u> >)		Derivative of waveform using subtraction of adjacent samples
(<source1>-<source2>	DIFFERENCE	Difference between two waveforms

DESKEW(<source>),<wavedeskew>	Shift a waveform in time by a specified amount
ERES(<source>),<bits>	Smoothing function defined by 0.5, 1.0, 1.5, 2.0, 2.5 or 3.0 extra bits of resolution
EXCELMATH(<source1>,<source2>,<ource1cell>,<source2cell>,<outputcell>,<source1headercell>,<source2headercell>,<outputheadercell>,<withheader>,<outputenable>,<source1enable>,<source2enable>,<newsheet>,<advanced>,<scaling>,<spreadsheetfilename>	Perform math in Excel. SCALING can be AUTOMATIC, MANUAL or FROM SHEET.
EXP(<source>)	Exponential (power of e)
EXP10(<source>)	Exponential (power of 10)
EXTR(<source>),<sweeps>	Extrema (Roof and Floor) of waveform
FFT(<source>,<type>,<>window>,<algorithm>,<filltype>,<suppressdc>	Fast Fourier transform of waveform. Available values are as follows – REAL, IMAGINARY, MAGNITUDE, PHASE, POWERSPECTRUM, POWERDENSITY BLACKMANHARRIS, FLATTOP, HAMMING, RECTANGULAR, VONHANN LEASTPRIME, POWER2 TRUNCATE, ZEROFILL
FILTER(<source>,<firoriir>,<filterkind>,<filtertype>,<window>,<kaiserbeta>,<gaussianbt>,<cosinebeta>,<transitionwidth>,<f3dbwidth>,<stopbandattenuation>,<passbandripple>,<lowfreqstop>,<lowfreqpass>,<f3dbfreq>,<cornerfreq>,<centerfreq>,<highfreqstop>,<highfreqpass>,<rolloff>,<numberoftaps>,<numberofstages>,<advanced>,<autolength>	Digital filter
FIR(<source>,<type>,<>window>,<cutoff>,<coefficients>)	Finite impulse response filter
FLOOR(<source>),<sweeps>	Lowest vertical value at each X value in N sweeps
GFILL(<source>,<samplesperperiod>,<sourcefrequency>,<time1>,<time2>,<setwindow>,<areatofill>,<>windowstart>,<>windowstop>	
HIST(<source>,<values>,<bins>,<horscale>,<center>,<verscaletype>,<autofindscale>	Histogram of parameter values. VERSCALETYPER can be LINEAR or LINCONSTMAX.
HISTD(<source>,<verscaletype>	Histogram of data
HSUM(<source>,<verscaletype>	Sum of a sequence of histograms

## PART TWO: COMMANDS

IFFT(<source>)	Inverse FFT, real output from complex input
INTG(<source>,<multiplier>,<adder>,<verscale>,<verroffset>)	
INTRP(<source>,<interpolatetype>,<expand>)	Interpolate extra points in waveform. INTERPOLATETYPE IS LINEAR, QUADRATIC, OR SINXX.
(-<source>) INVERT	Inversion (negation) of waveform
JITTERSIM	
LN(<source>)	Natural logarithm of waveform values
LOG10(<source>)	Base 10 logarithm of waveform values
LPFIR	
MCAD(<source1>,<source2>,<source1var>,<source2var>,<outputvar>,<source1headervar>,<source2headervar>,<outputheadervar>,<withheader>)	Produces a waveform using Mathcad
MATLAB(<source1>,<source2>,<matlabcode>,<matlabplot>,<matlabzerooffset>,<matlabscaleperdiv>)	Produces a waveform using MATLAB
PERHIST(<source>,<vercutcenter>,<vercutwidth>,<horocutcenter>,<horocutwidth>,<cutdirection>,<cuttype>,<cutwidth>)	Histogram of a slice through a persistence map
PMEAN(<source>)	Waveform derived from the mean of a persistence map
PRANGE(<source>,<pctpopulation>)	Waveform derived from the range of a persistence map
PSIGMA(<source>,<sigma>)	Waveform derived from the st dev of a persistence map
PHSHIFT(<source>,<phase>)	Change the phase of a set of complex data.
(<source1>*<source2>) PRODUCT	Product of two waveforms
(<source1>/<source2>) RATIO	Ratio of two waveforms
(1/<source>) RECIPROCAL	Reciprocal of a waveform
DESKEW(<source>,<wavedeskew> RESAMPLE	Shift a waveform in time.
RESC(<source>,<multiplier>,<adder>,<customunit>,<unit>)	Rescale waveform as Out = A*Wave(In) + B
ROOF(<source>,<sweeps>)	Highest Y value at each X in a set of waveforms
SEG(<source>,<selectedsegment>)	Select one segment from a sequence
SINX(<source>)	Ten times interpolation using sin (x) / x

SLICE(<source1>,<source2>,<frequency>,<priorperiods>,<postperiods>,<parameter>)	Slices waveform 2 to make many waveforms, using wf 1
SPACK(<source>)	Magnitude of complex result
SPARSE(<source>,<sparsingfactor>,<sparsingphase>)	Produces a waveform with fewer points than the input.
SQR(<source>)	Square of a waveform
SQRT(<source>)	Square root of waveform values
(<source1>+<source2>)                      SUM	Sum of two waveforms
TRACK(<source>,<autofindscale>,<verscale>,<center>)	Track of the values of a parameter
TREND(<source>,<verscale>,<center>,<autofindscale>)	Trend of the values of a parameter
WAVESCRIP(<source1>,<source2>,<language>,<code>,<status>,<timeout>)	Visual Basic script producing a waveform
ZOOMONLY(<source>)	Zoom of waveform
<b>NOTE: The numbers in CUST1 through CUST8 refer to the column numbers of the selected custom parameters. TA, TB, TC and TD are included for compatibility with existing scopes such as WavePro scopes and Waverunner scopes. These four mnemonics are not returned in responses to queries.</b>	

### SOURCE VALUES

<sourceN> : = {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, C3, C4}. TA through TD are for compatibility with existing software with earlier DSOs. These four mnemonics are not returned by queries. This is true for the following entries also.

<function> : = {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD}

<custom\_column> : = {CUST1, CUST2, CUST3, CUST4, CUST5, P1, P2, P3, P4, P5, P6, P7, P8} CUSTn are for backward compatibility, and are not returned by queries.

<extended\_source> : = {C1, C2, C3, C4, F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD, M1, M2, M3, M4}

### VALUES TO DEFINE NUMBER OF POINTS/SWEEPS:

<max\_points> : = 50 to 10 000 000

<max\_sweeps> : = 1 to 1000

<max\_sweeps> : = 1 to 1 000 000

<max\_sweeps> : = 1 to 50 000

PART TWO: COMMANDS

VALUES FOR RESCALE  
FUNCTION:

<addend> := 0.0 to 1e15  
<multiplier> := 0.0 to 1e15  
<units> := {UNCHANGED, A, CEL, C, HZ, K, N, OHM, PAL, V, W, DB, DEG, PCT, RAD, S}

RESCALE PHYSICAL UNITS VALUE NOTATION			
UNCHANGED	The unit remains unchanged.	PAL	Pascal
A	Amperes	V	Volt
CEL	Celsius	W	Watt
C	Coulomb	DB	decibel
HZ	Hertz	DEG	degree
K	Kelvin	PCT	percent
N	Newton	RAD	radian
OHM	Ohm	S	second

AVAILABILITY

<sourceN> := {C3, C4} only on four-channel oscilloscopes.  
<extended\_source> := {C3, C4} only on four-channel oscilloscopes

EXAMPLES (GPIB)

The following defines Trace A to compute the summed average of Channel 1 using 5000 points over 200 sweeps:

```
CMD$="TA:DEF  
EQN, 'AVGS(C1)',MAXPTS,5000,SWEEPS,200":  
CALL IBWRT(SCOPE%,CMD$)
```

The following defines Trace A to compute the product of Channel 1 and Channel 2, using a maximum of 10 000 input points:

```
CMD$="TA:DEF EQN, 'C1*C2',MAXPTS,10000": CALL  
IBWRT(SCOPE%,CMD$)
```

The following defines Trace A to compute the Power Spectrum of the FFT of Channel 1. A maximum of 1000 points will be used for the input. The window function is Rectangular.

```
CMD$="TA:DEF EQN, 'PS(FFT(C1))', MAXPTS, 1000, WINDOW,
RECT": CALL IBWRT(SCOPE%, CMD$)
```

The following defines Trace B to compute the Power Spectrum of the Power Average of the FFT being computed by Trace A, over a maximum of 244 sweeps.

```
CMD$="TB:DEF EQN, 'PS(AVGP(TA))', SWEEPS, 244":
CALL IBWRT(SCOPE%, CMD$)
```

The following defines Trace C to construct the histogram of the all rise time measurements made on source Channel 1. The rise time measurement is defined on custom line 2. The histogram has a linear vertical scaling and the rise time parameter values are binned into 100 bins.

```
CMD$="PACU 2, RISE, C1": CALL IBWRT(SCOPE%, CMD$)

CMD$="TC:DEF
EQN, 'HIST(CUST2)', VERT, LIN, MAXBINS, 100":
CALL IBWRT(SCOPE%, CMD$)
```

### RELATED COMMANDS

```
FIND_CTR_RANGE, FUNCTION_RESET, INR?,
PARAMETER_CUSTOM, PARAMETER_VALUE?,
PASS_FAIL_CONDITION
```

### MORE EXAMPLES OBTAINED USING F1:DEF?

**NOTE** New lines in these examples have been inserted for clarity. The actual strings are continuous.

```
F1:DEF EQN, "CORR(C1,C2)", CORRLength, 5 DIV, CORRSTART, 0E-3 DIV
```

```
F1:DEF EQN, "DERI(C1)", VERSCALE, 1E+6 V/S, VEROFFSET, 48E+3 V/S, ENABLEAUTOSCALE, ON
```

```
F1:DEF EQN, "ERES(C1)", BITS, 0.5
```

```
F1:DEF EQN, "EXCEL(C1,C2)", SOURCE1CELL, A2, SOURCE2CELL, B2, OUTPUTCELL, C2,
```

## PART TWO: COMMANDS

---

SOURCE1HEADERCELL,F2,SOURCE2HEADERCELL,G2,OUTPUTHEADERCELL,H2,  
WITHHEADER,ON,OUTPUTENABLE,ON,SOURCE1ENABLE,ON,SOURCE2ENABLE,OFF,  
NEWSHEET,ON,ADVANCED,OFF,SCALING,AUTOMATIC,  
SPREADSHEETFILENAME,D:\TEST34.XLS

F1:DEF EQN,"FFT(C1)",TYPE,POWERSPECTRUM,WINDOW,RECTANGULAR,  
ALGORITHM,LEASTPRIME,FILLTYPE,TRUNCATE,SUPPRESSDC,ON

F1:DEF EQN,"FILTER(C1)",FIORIRIIR,FIR,FILTERKIND,LOWPASS,FILTERTYPE,BUTTERWORTH,  
WINDOW,RECTANGULAR,KAISERBETA,0,GAUSSIANBT,30 PCT,COSINEBETA,30 PCT,  
TRANSITIONWIDTH,100E+3 HZ,F3DBWIDTH,100E+3 HZ,STOPBANDATTENUATION,40 DB,  
PASSBANDRIPPLE,1 DB,PASSBANDATTENUATION,1 DB,LOWFREQSTOP,1.1E+6 HZ,  
LOWFREQPASS,1E+6 HZ,F3DBFREQ,1E+6 HZ,CORNERFREQ,1E+6 HZ,CENTERFREQ,1E+6 HZ,  
HIGHFREQSTOP,10E+6 HZ,HIGHFREQPASS,10E+6 HZ,ROLLOFF,5 DB,  
NUMBEROFTAPS,11,NUMBEROFSTAGES,4,ADVANCED,OFF,AUTOLENGTH,OFF

F1:DEF EQN,"FIR(C1)",TYPE,LOWPASS,WINDOW,RECTANGULAR,LFCUTOFF,1E+6 HZ,  
COEFFICIENTS,32

F1:DEF EQN,"HIST(C1)",VALUES,1000,BINS,100,HORSCALE,50E-3 V,CENTER,-2E-3 V,  
VERSCALETYPE,LINEAR,AUTOFINDSCALE,ON

F1:DEF EQN,"INTG(C1)",MULTIPLIER,1,ADDER,0E-12,VERSCALE,5E-9,VEROFFSET,7.703E-9

F1:DEF EQN,"INTRP(C1)",INTERPOLATETYPE,SINXX,EXPAND,10

F1:DEF EQN,"MCAD(C1, C2)",SOURCE1VAR,S1,SOURCE2VAR,S2,OUTPUTVAR,S1,  
SOURCE1HEADERVAR,S1HDR,SOURCE2HEADERVAR,S2HDR,OUTPUTHEADERVAR,O1HDR,  
WITHHEADER,ON,OUTPUTENABLE,ON,SOURCE1ENABLE,ON,SOURCE2ENABLE,OFF,  
NEWSHEET,ON,ADVANCED,OFF,SCALING,AUTOMATIC,

WORKSHEETFILENAME,C:\MATHCAD89.MCD

F1:DEF EQN,"MATLAB(C1,C2)",MATLABCODE,WFORMOUT = -0.5 \* WFORMIN1;,  
MATLABPLOT,OFF,MATLABZEROOFFSET,0E-9,MATLABSCALEPERDIV,50E-3

F1:DEF EQN,"SPARSE(C1)",SPARSINGFACTOR,1,SPARSINGPHASE,0

F1:DEF EQN,"TRACK(C1)",AUTOFINDSCALE,ON,VERSCALE,50E-3,CENTER,-1.56249851E-3

F1:DEF EQN,"SCRIPT(C1,C2)",LANGUAGE,VBSCRIPT,  
CODE,FUNCTION UPDATE() + vbcrlf + "OUTRESULT.SAMPLES = INRESULT.SAMPLES" + vbcrlf  
+ "STARTDATA = 0" + vbcrlf + "ENDDATA = OUTRESULT.SAMPLES" + vbcrlf +  
"REDIM NEWDATAARRAY(OUTRESULT.SAMPLES)" + vbcrlf +  
"UNSCALEDATA = INRESULT.DATAARRAY(FALSE)" + vbcrlf +  
"LASTPOINT = ENDDATA - 1" + vbcrlf + "FOR I = 0 TO LASTPOINT" +  
vbcrlf + "NEWDATAARRAY(I) = -UNSCALEDATA(I)" + vbcrlf + "NEXT" +  
vbcrlf + "OUTRESULT.DATAARRAY(FALSE) = NEWDATAARRAY" + vbcrlf + "END FUNCTION" +  
vbcrlf + ",STATUS,OK,TIMEOUT,360 S

### *MISCELLANEOUS*

### **DELETE\_FILE, DELF** Command

**DESCRIPTION** The DELETE\_FILE command deletes a file from the currently selected directory.

**COMMAND SYNTAX** `DELF,DISK,<medium>,FILE,'<filename>'`  
`<medium>:= {FLOPPY,HDD}`

**EXAMPLE (GPIB)** The following deletes a front panel setup from the floppy disk:

```
CMD$="DELF,DISK,FLOPPY,FILE,'TESTRUN.TRC'  
CALL IBWRT(SCOPE%,CMD$)
```

### DISPLAY

**DISPLAY, DISP**  
Command/Query

#### DESCRIPTION

The **DISPLAY** command controls the display screen of the oscilloscope. When remotely controlling the oscilloscope, and if you do not need to use the display, it can be useful to switch off the display via the **DISPLAY OFF** command. This improves oscilloscope response time, since the waveform graphic generation procedure is suppressed.

The response to the **DISPLAY?** query indicates the display state of the oscilloscope.

**NOTE: When you set the display to OFF, the real-time clock and the message field are updated. But waveforms and associated texts remain unchanged.**

#### COMMAND SYNTAX

**DISPlay** <state>  
<state> := {ON, OFF}

#### QUERY SYNTAX

**DISPlay?**

#### RESPONSE FORMAT

**DISPlay** <state>

#### EXAMPLE (GPIB)

The following turns off the display:

```
CMD$="DISP OFF": CALL IBWRT(SCOPE%,CMD$)
```

### ***DISPLAY***

**DOT\_JOIN, DTJN**  
Command/Query

**DESCRIPTION** The DOT\_JOIN command controls the interpolation lines between data points.

**COMMAND SYNTAX** DoT\_JoiN <state>  
<state> := {ON, OFF}

**QUERY SYNTAX** DoT\_JoiN?

**RESPONSE FORMAT** DoT\_JoiN <state>

**EXAMPLE (GPIB)** The following turns off the interpolation lines:  
CMD\$="DTJN OFF": CALL IBWRT(SCOPE%, CMD\$)

### **DISPLAY**

**EMAIL, MAIL**  
Command/Query

### **DESCRIPTION**

The EMAIL command sets up the Email system.

### **COMMAND SYNTAX**

EMAIL  
MODE, <smtpOrMapi>, TO, ' <toAddress> ', FROM, ' <fromAddress> ', SERVER, ' <smtpServer> '  
  
<smtpOrMapi>:= {SMTP, MAPI}  
  
<toAddress>:= valid recipient address, e.g.,  
"myName@myProvider.com"  
  
<fromAddress>:= valid originator address, e.g.,  
"myXStreamDSO@LeCroy.com"  
  
<smtpServer>:= valid SMTP server address, e.g.,  
"domino.lecroy.com"

### **QUERY SYNTAX**

eMAIL?

### **RESPONSE FORMAT**

MAIL  
MODE, SMTP, TO, "MYNAME@MYPROVIDER.COM", FROM, "MYXSTREAMDSO@LECROY.COM", SERVER, "DOMINO.LECROY.COM"

### **EXAMPLE (GPIB)**

```
CMD$=" MAIL
MODE, SMTP, TO, ' MYNAME@MYPROVIDER.COM' , FROM, ' MY
XSTREAMDSO@LECROY.COM' , SERVER, ' DOMINO.LECROY.
COM' " : CALL IBWRT( SCOPE%, CMD$ )
```

### **STATUS**

**\*ESE**  
Command/Query

#### **DESCRIPTION**

The \*ESE command sets the Standard Event Status Enable register (ESE). This command allows one or more events in the ESR register to be reflected in the ESB summary message bit (bit 5) of the STB register. For an overview of the ESB defined events, refer to the ESR table on page 138.

The \*ESE? query reads the contents of the ESE register.

#### **COMMAND SYNTAX**

\*ESE <value>  
<value> := 0 to 255

#### **QUERY SYNTAX**

\*ESE?

#### **RESPONSE FORMAT**

\*ESE <value>

#### **EXAMPLE (GPIB)**

The following allows the ESB bit to be set if a user request (URQ bit 6, i.e. decimal 64) and/or a device dependent error (DDE bit 3, i.e. decimal 8) occurs. Summing these values yields the ESE register mask  $64+8=72$ .

```
CMD$="*ESE 72": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

\*ESR

### **STATUS**

**\*ESR?**  
Query

#### **DESCRIPTION**

The \*ESR? query reads and clears the contents of the Event Status Register (ESR). The response represents the sum of the binary values of the register bits 0 to 7. The table below gives an overview of the ESR register structure.

#### **QUERY SYNTAX**

\*ESR?

#### **RESPONSE FORMAT**

\*ESR <value>  
<value> := 0 to 255

#### **EXAMPLE (GPIB)**

The following reads and clears the contents of the ESR register:

```
CMD$="*ESR?": CALL IBWRT(SCOPE%,CMD$):
```

```
CALL IBRD(SCOPE%,RSP$): PRINT RSP$
```

Response message:

```
*ESR 0
```

#### **RELATED COMMANDS**

ALL\_STATUS, \*CLS, \*ESE

ADDITIONAL INFORMATION

STANDARD EVENT STATUS REGISTER (ES)					
Bit	Bit Value	Bit Name	Description		See Note ...
15...8			0	Reserved by IEEE 488.2	
7	128	PON	1	Power off-to-ON transition has occurred	1.
6	64	URQ	1	User ReQuest has been issued	2.
5	32	CME	1	CoMmand parser Error has been detected	3.
4	16	EXE	1	EXecution Error detected	4.
3	8	DDE	1	Device Dependent (specific) Error occurred	5.
2	4	QYE	1	QuerY Error occurred	6.
1	2	RQC	0	Oscilloscope never ReQuests bus Control	7.
0	1	OPC	0	OPeration Complete bit not used	8.

**NOTE:** (refer to table above)

- The Power On (PON) bit is always turned on (1) when the unit is powered up.**
- The User Request (URQ) bit is set true (1) when a soft key is pressed. An associated register URR identifies which key was selected. For further details refer to the URR? query.**
- The CoMmand parser Error bit (CME) is set true (1) whenever a command syntax error is detected. The CME bit has an associated CoMmand parser Register (CMR) which specifies the error code. Refer to the query CMR? for further details.**
- The EXecution Error bit (EXE) is set true (1) when a command cannot be executed due to some device condition (e.g. oscilloscope in local state) or a semantic error. The EXE bit has an associated Execution Error Register (EXR) which specifies the error code. Refer to query EXR? for further details.**



5. **The Device specific Error (DDE) is set true (1) whenever a hardware failure has occurred at power-up, or execution time, such as a channel overload condition, a trigger or a timebase circuit defect. The origin of the failure can be localized via the DDR? query.**
6. **The QuerY Error bit (QYE) is set true (1) whenever (a) an attempt is made to read data from the Output Queue when no output is either present or pending, (b) data in the Output Queue has been lost, (c) both output and input buffers are full (deadlock state), (d) an attempt is made by the controller to read before having sent an <END>, (e) a command is received before the response to the previous query was read (output buffer flushed).**
7. **The ReQuest Control bit (RQC) is always false (0), as the oscilloscope has no GPIB controlling capability.**
8. **The OPeration Complete bit (OPC) is set true (1) whenever \*OPC has been received, since commands and queries are strictly executed in sequential order. The oscilloscope starts processing a command only when the previous command has been entirely executed.**

<b>STATUS</b>	<b>EXR?</b> Query
<b>DESCRIPTION</b>	The EXR? query reads and clears the contents of the EXecution error Register (EXR). The EXR register specifies the type of the last error detected during execution. Refer to the table next page.
<b>QUERY SYNTAX</b>	EXR?
<b>RESPONSE FORMAT</b>	EXR <value> <value> : = 21 to 64
<b>EXAMPLE (GPIB)</b>	<p>The following reads the contents of the EXR register:</p> <pre>CMD\$="EXR?" : CALL IBWRT( SCOPE% ,CMD\$ ) : CALL IBRD( SCOPE% ,RSP\$ ) : PRINT RSP\$</pre> <p>Response message (if no fault):</p> <pre>EXR 0</pre>
<b>RELATED COMMANDS</b>	ALL_STATUS, *CLS

## ADDITIONAL INFORMATION

EXECUTION ERROR STATUS REGISTER STRUCTURE (EXR)	
Value	Description
21	Permission error. The command cannot be executed in local mode.
22	Environment error. The oscilloscope is not configured to correctly process a command. For instance, the oscilloscope cannot be set to RIS at a slow timebase.
23	Option error. The command applies to an option which has not been installed.
24	Unresolved parsing error.
25	Parameter error. Too many parameters specified.
26	Non-implemented command.
30	Hex data error. A non-hexadecimal character has been detected in a hex data block.
31	Waveform error. The amount of data received does not correspond to descriptor indicators.
32	Waveform descriptor error. An invalid waveform descriptor has been detected.
33	Waveform text error. A corrupted waveform user text has been detected.
34	Waveform time error. Invalid RIS or TRIG time data has been detected.
35	Waveform data error. Invalid waveform data have been detected.
36	Panel setup error. An invalid panel setup data block has been detected.
50	No mass storage present when user attempted to access it. *
51	Mass storage not formatted when user attempted to access it. *
53	Mass storage was write protected when user attempted to create a file, to delete a file, or to format the device. *
54	Bad mass storage detected during formatting. *
55	Mass storage root directory full. Cannot add directory. *
56	Mass storage full when user attempted to write to it. *
57	Mass storage file sequence numbers exhausted (999 reached). *
58	Mass storage file not found. *
59	Requested directory not found. *
61	Mass storage filename not DOS compatible, or illegal filename. *
62	Cannot write on mass storage because filename already exists. *

\* Only with memory card or removable hard disk option.

### ***FUNCTION***

**FIND\_CTR\_RANGE**

Command

### **DESCRIPTION**

Sets the center and range of a graphical function, such as a histogram, to provide an optimal display of the values.

### **COMMAND SYNTAX**

Find\_Ctr\_Range, FCR

### **EXAMPLE**

Assuming that Trace F4 has been defined as a histogram of a parameter, the following will determine suitable settings for the center and the range:

```
CMD$="F4:FCR": CALL IBWRT(SCOPE%,CMD$)
```

### **ACQUISITION**

**FORCE\_TRIGGER, FRTR**  
Command

#### **DESCRIPTION**

Causes the DSO to make one acquisition.

#### **COMMAND\_SYNTAX**

FoRce\_TRigger

#### **EXAMPLE (GPIB)**

Either of the following pairs of instructions will force the scope to make one acquisition:

```
CMD$="TRMD SINGLE;ARM;FRTR"  
CALL IBWRT(Scope%, CMD$)
```

```
CMD$ = "TRMD STOP;ARM;FRTR"  
CALL IBWRT(Scope%, CMD$)
```

### ***FUNCTION***

**FUNCTION\_RESET, FRST**  
Command

#### **DESCRIPTION**

The FUNCTION\_RESET command resets a waveform processing function. The number of sweeps will be reset to zero and the process restarted.

#### **COMMAND SYNTAX**

<function> : Function\_ReSeT

#### **EXAMPLE (GPIB)**

<function> : = {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD}  
TA through TD are included for backward compatibility with software designed for earlier LeCroy DSOs. They are not used in responses to queries.

Assuming that Trace F1 has been defined as the summed average of Channel 1, the following will restart the averaging process:

```
CMD$="F1:FRST": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

INR

### **DISPLAY**

#### **GRID**

Command/Query

#### **DESCRIPTION**

The GRID command defines the style of the grid used in the display.  
The GRID? query returns the grid style currently in use.

#### **COMMAND SYNTAX**

GRID <grid>

<grid> := {SINGLE, DUAL, QUAD,  
OCTAL, AUTO, XY, XYSINGLE, XYDUAL}

#### **QUERY SYNTAX**

GRID?

#### **RESPONSE FORMAT**

GRID <grid>

#### **EXAMPLE (GPIB)**

The following sets the screen display to dual grid mode:

```
CMD$="GRID DUAL": CALL IBWRT(SCOPE%,CMD$)
```

### ***HARD COPY***

### **HARDCOPY\_SETUP, HCSU**

Command/Query

#### **DESCRIPTION**

The HARDCOPY\_SETUP command configures the DSO's hard copy driver. It enables you to specify the device type and transmission mode of the hard copy unit connected to the DSO. This can be clipboard or disk drive as well as a printer. One or more individual settings can be changed by specifying the appropriate keywords, together with the new values. As with all multiple entry commands, you can send any combination or permutation of keywords. If you send contradictory values within a command, the result is governed by the last one sent. See the following pages for command notation. Note that some values are in quotation marks, because they are to be the names of directories, files, or printers, which may contain spaces and other non-alphanumeric characters. The other values must be chosen from the lists provided below.

#### **COMMAND SYNTAX**

```
HardCopy_SetUp DEV,  
<device>, FORMAT, <format>, BCKG, <bckg>, DEST, "<dest  
ination>", DIR, "<directory>", FILE, "<filename>",  
AREA, <hardcopyarea>, PRINTER, "<printername>"
```

<device> := {PSD, BMP, BMPCOMP, JPEG, PNG, TIFF } <-Fonts

<portname> := {GPIB, RS232, NET} only accepted with a command (for backward compatibility) and sets DEST, REMOTE.

<hardcopyArea>:= {GRIDAREAONLY, DSOWINDOW, FULLSCREEN }

<destination> := {PRINTER, CLIPBOARD, EMAIL, FILE, REMOTE }

<filename> := valid destination filename, auto incremented, for FILE mode only.

<directory> := valid destination directory name, for FILE mode only.

<printername> := valid printer name, for PRINTER mode only.

<format> := {PORTRAIT, LANDSCAPE}

<bckg> := {BLACK, WHITE}

#### **QUERY SYNTAX**

HCSU?

### RESPONSE FORMAT

(If preceded, for example, by `hcsu dest,file;` with `CHDR OFF`)

```
DEV,PNG,FORMAT, PORTRAIT,BCKG,BLACK,DEST,REMOTE,
DIR,"C:\LECROY\XSTREAM\HARDCOPY",
FILE,"IRHCP1.PNG",AREA,GRIDAREAONLY,PRINTER,"GE
NEVSVRHP4050RD"
```

### EXAMPLE (GPIB)

The following example selects output to the printer named "Local Printer":

```
CMD$="HCSU DEST,PRINTER,PRINTER,"Local
Printer" "
```

```
CALL IBWRT(SCOPE%,CMD$)
```

### RELATED COMMANDS

`HARDCOPY_TRANSMIT`, `SCREEN_DUMP`

### ADDITIONAL INFORMATION

Hard-copy command parameters are grouped in pairs. The first in the pair names the variable to be modified, while the second gives the new value to be assigned. Pairs can be given in any order and can be restricted to those variables to be changed. Omitted variables will be unaffected.

### DISPLAY

### HOR\_MAGNIFY, HMAG

Command/Query

#### DESCRIPTION

The HOR\_MAGNIFY command horizontally expands the selected expansion trace by a specified factor. Magnification factors not within the range of permissible values will be rounded off to the closest legal value.

If multiple zoom is enabled, the magnification factor for all expansion traces is set to the specified factor. If the specified factor is too large for any of the expanded traces (depending on their current source), it is reduced to an acceptable value and only then applied to the traces.

The VAB bit (bit 2) in the STB register (see table on page 208) is set when a factor outside the legal range is specified.

The HOR\_MAGNIFY? query returns the current magnification factor for the specified expansion function.

#### COMMAND SYNTAX

<exp\_trace> : Hor\_MAGnify <factor>  
<exp\_trace> := {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD} TA through TD are included for backward compatibility with software designed for earlier LeCroy DSOs. These four mnemonics will not be returned in response to queries.  
<factor> := 1 to 20000

#### QUERY SYNTAX

<exp\_source> : Hor\_MAGnify?

#### RESPONSE FORMAT

<exp\_source> : Hor\_MAGnify <factor>

#### EXAMPLE (GPIB)

The following horizontally magnifies Trace A (TA) by a factor of 5:

```
CMD$="F1:HMAG 5": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

DUAL\_ZOOM, MULTI\_ZOOM

### DISPLAY

**HOR\_POSITION, HPOS**  
Command/Query

#### DESCRIPTION

The HOR\_POSITION command horizontally positions the geometric center of the intensified zone on the source trace. Allowed positions range from division 0 through 10. If the source trace was acquired in sequence mode, horizontal shifting will only apply to a single segment at a time.

If the multiple zoom is enabled, the difference between the specified and the current horizontal position of the specified trace is applied to all expanded traces. If this would cause the horizontal position of any expanded trace to go outside the left or right screen boundaries, the difference of positions is adapted and then applied to the traces.

If the sources of expanded traces are sequence waveforms, and the multiple zoom is enabled, the difference between the specified and the current segment of the specified trace is applied to all expanded traces. If this would cause the segment of any expanded trace to go outside the range of the number of source segments, the difference is adapted and then applied to the traces.

The VAB bit (bit 2) in the STB register (see table on page 208) is set if a value outside the legal range is specified.

The HOR\_POSITION? query returns the position of the geometric center of the intensified zone on the source trace.

**NOTE: Segment number 0 has the special meaning “Show All Segments Unexpanded”.**

#### COMMAND SYNTAX

`<exp_trace> : Hor_POSition <hor_position> , <segment>`  
`<exp_trace> : = {F1 , F2 , F3 , F4 , F5 , F6 , F7 , F8 , TA , TB , TC , TD , M1 , M2 , M3 , M4}` TA through TD are included for backward compatibility with software designed for earlier LeCroy DSOs. These four mnemonics will not be returned in response to queries.  
`<hor_position> : = 0 to 10 DIV`  
`<segment> : = 0 to max segments`

**NOTE:** The segment number is only relevant for waveforms acquired in sequence mode; it is ignored in single waveform acquisitions. When the segment number is set to 0, all segments will be shown.

The suffix DIV is optional.

### QUERY SYNTAX

<exp\_trace>:Hor\_POSition?

### RESPONSE FORMAT

<exp\_trace>:Hor\_POSition <hor\_position>[, <segment>]

**NOTE:** The segment number is only given for sequence waveforms.

### EXAMPLE (GPIB)

The following positions the center of the intensified zone on the trace currently viewed by Trace A (TA) at division 3:

```
CMD$="F1:HPOS 3": CALL IBWRT(SCOPE%,CMD$)
```

### RELATED COMMANDS

MULTI\_ZOOM

### MISCELLANEOUS

**\*IDN?**  
Query

#### DESCRIPTION

The \*IDN? query is used for identification purposes. The response consists of four different fields providing information on the manufacturer, the scope model, the serial number and the firmware revision level.

#### QUERY SYNTAX

\*IDN?

#### RESPONSE FORMAT

\*IDN LECROY, <model>, <serial\_number>, <firmware\_level>

<model> : = A six- or seven-character model identifier

<serial\_number> : = A nine- or 10-digit decimal code

<firmware\_level> : = two digits giving the major release level followed by a period, then one digit giving the minor release level followed by a period and a single-digit update level (xx.y.z)

#### EXAMPLE (GPIB)

This issues an identification request to the scope:

```
CMD$="*IDN?": CALL IBWRT(SCOPE%,CMD$):  
CALL IBRD(SCOPE%,RSP$): PRINT RSP$
```

Response message:

```
*IDN LECROY,WAVEMASTER,WM01000
```

### STATUS

**INE**

Command/Query

#### DESCRIPTION

The **INE** command sets the INternal state change Enable register (INE). This command allows one or more events in the INR register to be reflected in the INB summary message bit (bit 0) of the STB register. For an overview of the INR defined events, refer to the table next page.

The **INE?** query reads the contents of the INE register.

#### COMMAND SYNTAX

**INE** <value>

<value> : = 0 to 65535

#### QUERY SYNTAX

**INE?**

#### RESPONSE FORMAT

**INE** <value>

#### EXAMPLE (GPIB)

The following allows the INB bit to be set whenever a screen dump has finished (bit 1, i.e. decimal 2), or a waveform has been acquired (bit 0, i.e. decimal 1), or both of these. Summing these two values yields the INE mask  $2+1=3$ .

```
CMD$="INE 3":CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

**INR?**

## STATUS

**INR?**  
Query

## DESCRIPTION

The INR? Query reads and clears the contents of the Internal state change Register (INR). The INR register (table below) records the completion of various internal operations and state transitions.

INTERNAL STATE REGISTER STRUCTURE (INR)			
Bit	Bit Value	Description	
15		0	Reserved for future use.
14	16384	1	Probe was changed.
13	8192	1	Trigger is ready.
12	4096	1	Pass/Fail test detected desired outcome.
11	2048	1	Reserved
10	1024	1	Reserved
9	512	1	Reserved
8	256	1	Reserved
7	128	1	A floppy or hard disk exchange has been detected.
6	64	1	Floppy or hard disk has become full in AutoStore Fill mode.
5	32	0	Reserved for LeCroy use.
4	16	1	A segment of a sequence waveform has been acquired in acquisition memory but not yet read out into the main memory.
3	8	1	A time-out has occurred in a data block transfer.
2	4	1	A return to the local state is detected.
1	2	1	A screen dump has terminated.
0	1	1	A new signal has been acquired in acquisition memory and read out into the main memory.

### QUERY SYNTAX

INR?

### RESPONSE FORMAT

INR <state>

<state> := 0 to 65535

### EXAMPLE (GPIB)

The following reads the contents of the INR register:

```
CMD$="INR?": CALL IBWRT(SCOPE%,CMD$)
```

Response message:

INR 1026

i.e. waveform processing in Function C and a screen dump have both terminated.

### RELATED COMMANDS

ALL\_STATUS, \*CLS, INE

## WAVEFORM TRANSFER

INSPECT?, INSP?  
Query

### DESCRIPTION

The INSPECT? query allows you to read parts of an acquired waveform in intelligible form. The command is based on the explanation of the format of a waveform given by the template (use the TEMPLATE? query to obtain an up-to-date copy).

Any logical block of a waveform can be inspected using this query by giving its name enclosed in quotes as the first (string) parameter (see the template itself).

The special logical block named WAVEDESC can also be inspected in more detail. By giving the name of a variable in the block WAVEDESC, enclosed in quotes as the first (string) parameter, it is possible to inspect only the actual value of that variable. See Chapter 4 for more on INSPECT?.

NOTATION	
BYTE	raw data as integers (truncated to 8 most significant bits)
FLOAT	normalized data (gain, offset applied) as floating point numbers (gives measured values in volts or units)
WORD	raw data as integers (truncated to 16 most significant bits)

### QUERY SYNTAX

<trace> : INSPECT? '<string>' [, <data\_type>]

<trace> := {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, C3, C4} Note that TA through TD are included for compatibility with software designed for earlier LeCroy DSOs. These four labels will not be returned in responses to queries.

<data\_type> := {BYTE, WORD, FLOAT}. TA through TD are for backward compatibility, and are not returned by queries.

**NOTE: The optional parameter <data\_type> applies only for inspecting the data arrays. It selects the representation of the data. The default <data\_type> is FLOAT.**

### RESPONSE FORMAT

`<trace> : INSPect "<string>"`  
`<string> : =` A string giving name(s) and value(s) of a logical block or a variable.

### AVAILABILITY

`<trace> : = {C3, C4}` only on four-channel oscilloscopes.

### EXAMPLES (GPIB)

The following reads the value of the timebase at which the last waveform in Channel 1 was acquired:

```
CMD$="C1:INSP? `TIMEBASE` "  
CALL IBWRT(SCOPE%,CMD$)  
CALL IBRD(SCOPE%,RSP$)  
PRINT RSP$
```

Response message:

```
C1:INSP `TIMEBASE: 500 US/DIV`
```

The following reads the entire contents of the waveform descriptor block:

```
CMD$="C1:INSP? `WAVEDESC` "
```

### RELATED COMMANDS

TEMPLATE, WAVEFORM\_SETUP

### **DISPLAY**

### **INTENSITY, INTS**

Command/Query

#### **DESCRIPTION**

The INTenSity command sets the intensity level of the grid. The command TRACE,n is accepted for backward compatibility, but the actual trace intensity is always 100%.

#### **COMMAND SYNTAX**

INTenSity GRID,<value>,TRACE,<value>[PCT]  
<value>:= 0 to 100 (in percent). GRID and TRACE may be interchanged ? the order is immaterial.

#### **QUERY SYNTAX**

INTenSity?

#### **RESPONSE FORMAT**

INTenSity TRACE,<value>,GRID,<value>

#### **EXAMPLE (GPIB)**

The following example sets the grid intensity to 70%.

```
CMD$="INTS GRID,70" : CALL  
IBWRT(SCOPE%,CMD$)
```

### ACQUISITION

### INTERLEAVED, ILVD

Command/Query

#### DESCRIPTION

The INTERLEAVED command enables or disables random interleaved sampling (RIS) for timebase settings where both single shot and RIS mode are available. See the specifications in the Operator's Manual.

RIS is not available for sequence mode acquisitions. If sequence mode is on, ILVD ON turns it off.

The response to the INTERLEAVED? query indicates whether the oscilloscope is in RIS mode.

#### COMMAND SYNTAX

```
InterLeaVeD <mode>  
<mode> := {ON, OFF}
```

#### QUERY SYNTAX

```
InterLeaVeD?
```

#### RESPONSE FORMAT

```
InterLeaVeD <mode>
```

#### EXAMPLE

The following instructs the oscilloscope to use RIS mode:

```
CMD$="ILVD ON": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

```
TIME_DIV, TRIG_MODE, MEMORY_SIZE, SEQUENCE
```

### **STATUS**

**\*IST?**  
Query

#### **DESCRIPTION**

The \*IST? (Individual Status) query reads the current state of the IEEE 488.1-defined “ist” local message. The “ist” individual status message is the status bit sent during a parallel poll operation.

#### **QUERY SYNTAX**

\*IST?

#### **RESPONSE FORMAT**

\*IST <value>  
<value> : = 0 or 1

#### **EXAMPLE (GPIB)**

The following cause the contents of the IST bit to be read:

```
CMD$="*IST?": CALL IBWRT(SCOPE%,CMD$):  
CALL IBRD(SCOPE%,RSP$): PRINT RSP$
```

Response message

\*IST 0

#### **RELATED COMMANDS**

\*PRE

### ACQUISITION

#### MEMORY\_SIZE, MSIZ

Command/Query

#### DESCRIPTION

On most models where this command/query is available, MEMORY\_SIZE allows selection of the maximum memory length used for acquisition. See the specifications in the *Operator's Manual*.

**TIP: Reduce the number of data points for faster throughput.**

The MEMORY\_SIZE? query returns the current maximum memory length used to capture waveforms. When the optional suffix NUM is used with the query, the response will be returned in standard numeric format.

#### COMMAND SYNTAX

**NOTE: The oscilloscope will adapt to the closest valid <size> or numerical <value> according to available channel memory.**

Memory\_SIZE <size>

<size> : = {500, 1e+3, ..., 2.5e+6, 5e+6, 1e+7}, for example, in standard numeric format.

Or, alternatively:

= {500, 1000, 2500, 5000, 10K, 25K, 50K, 100K, 250K, 500K, 1MA, 2.5MA, 5MA, 10MA, 25MA}

However, values not absolutely identical to those listed immediately above will be recognized by the scope as *numerical data* (see the table under this heading in Chapter 1). For example, the scope will recognize 1.0M as 1 millisample. But it will recognize 1.0MA as 1 megasample.

#### QUERY SYNTAX

Memory\_SIZE? [NUM]

#### RESPONSE FORMAT

Memory\_SIZE <size>

#### EXAMPLE

The following will set the oscilloscope to acquire at most 10 000 data samples per single-shot or RIS acquisition:

```
CMD$="MSIZ 10K": CALL IBWRT(SCOPE%,CMD$)
```

```
or CMD$="MSIZ 10e+3": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

TDIV

### **DISPLAY**

#### **MESSAGE, MSG**

Command/Query

#### **DESCRIPTION**

The MESSAGE command displays a flashing string of characters in the message field at the bottom of the DSO screen.

#### **COMMAND SYNTAX**

MeSsaGe '<string>' or MSG "<string>"

<string>:= a string of up to 49 characters. Longer strings will be truncated to 49 characters, but the original string will be retained and returned by the MSG? Query.

#### **QUERY SYNTAX**

MeSsaGe?

#### **RESPONSE FORMAT**

MeSsaGe "<string>"

#### **EXAMPLE (GPIB)**

The following causes the message Touch Probe 2 to Test Point 7 to appear in the message field.

```
CMD$="MSG `Touch Probe 2 to Test Point 7`":  
CALL IBWRT(SCOPE%, CMD$)
```

### ACQUISITION

#### OFFSET, OFST

Command/Query

#### DESCRIPTION

The OFFSET command allows adjustment of the vertical offset of the specified input channel.

The maximum ranges depend on the fixed sensitivity setting. See the Operator's Manual.

If an out-of-range value is entered, the oscilloscope is set to the closest possible value and the VAB bit (bit 2) in the STB register is set.

**NOTE: The probe attenuation factor is not taken into account for adjusting the offset.**

**The suffix V is optional.**

The OFFSET? query returns the DC offset value of the specified channel.

#### COMMAND SYNTAX

<channel> : OFfSeT <offset>

<channel> : = {C1, C2, C3, C4}

<offset> : = See the Operator's Manual for specifications.

#### QUERY SYNTAX

<channel> : OFfSeT?

#### RESPONSE FORMAT

<channel> : OFfSeT <offset>

#### AVAILABILITY

<channel> : {C3, C4} only on four-channel oscilloscopes.

#### EXAMPLE (GPIB)

The following instruction sets the offset of Channel 2 to -3 V:

```
CMD$="C2:OFST -3V": CALL IBWRT(SCOPE%,CMD$)
```

### ***CURSOR***

### **OFFSET\_CONSTANT, OFCT**

Command/Query

#### **DESCRIPTION**

As you change the gain, this command allows you to either keep the vertical offset level indicator stationary (when Div is selected) or to have it move with the actual voltage level (when Volts is selected). The advantage of selecting Div is that the waveform will remain on the grid as you increase the gain; whereas, if Volts is selected, the waveform could move off the grid.

Regardless of whether you select Volts or Div, the "Offset" shown in the scope's channel setup dialog always indicates volts. However, when Div is selected for the Offset Control, the offset in volts is scaled proportional to the change in gain, thereby keeping the division on the grid constant.

#### **COMMAND SYNTAX**

OFFset\_Constant

#### **QUERY SYNTAX**

OFCT?

#### **RESPONSE FORMAT**

OFCT VOLTS

#### **EXAMPLE (GPIB)**

```
CMD$ = "OFCT VOLTS": CALL IBWRT(SCOPE%, CMD$)
```

STATUS	<div>*OPC</div> <div>Command/Query</div>
DESCRIPTION	<p>The *OPC (Operation Complete) command sets to true the OPC bit (bit 0) in the standard Event Status Register (ESR). This command has no other effect on the operation of the oscilloscope because the oscilloscope starts parsing a command or query only after it has completely processed the previous command or query.</p> <p>The *OPC? query always responds with the ASCII character “1” because the oscilloscope only responds to the query when the previous command has been entirely executed.</p>
COMMAND SYNTAX	*OPC
QUERY SYNTAX	*OPC?
RESPONSE FORMAT	*OPC 1
RELATED COMMANDS	*WAI

### MISCELLANEOUS

**\*OPT?**  
Query

#### DESCRIPTION

The \*OPT? query identifies oscilloscope options: installed firmware or hardware that is additional to the standard instrument configuration. The response consists of a series of response fields listing all the installed options.

#### QUERY SYNTAX

\*OPT?

#### RESPONSE FORMAT

\*OPT <option\_1> , <option\_2> , ... , <option\_N>

<option\_n> := A three- or four-character ASCII string

**NOTE: If no option is present, the character 0 will be returned.**

#### EXAMPLE (GPIB)

The following queries the installed options:

```
CMD$="*OPT?" : CALL IBWRT(SCOPE% , CMD$) :
```

```
CALL IBRD(SCOPE% , RSP$) : PRINT RSP$
```

If, for example, the waveform processing options DFP2, SDM, JTA2, and GPIB and are installed, the response will be returned as:

```
* DFP2 , SDM , JTA2 , GPIB
```

Response message if no options are installed:

```
*OPT 0
```

### SAVE/RECALL SETUP

**PANEL\_SETUP, PNSU**  
Command/Query

#### DESCRIPTION

The PANEL\_SETUP command complements the \*SAV or \*RST commands. PANEL\_SETUP allows you to archive panel setups in encoded form on external storage media.

Only setup data read by the PNSU? query can be recalled into the oscilloscope. A panel setup error (see table on page 141) will be generated if the setup data block contains invalid data.

**NOTE: The communication parameters (those modified by commands CFMT, CHDR, CHLP, CORD and WFSU) and the enable registers associated with the status reporting system (SRE, PRE, ESE, INE) are not saved by this command.**

#### COMMAND SYNTAX

PaNel\_SetUp <setup>

<setup> : = A setup data block previously read by PNSU?

#### QUERY SYNTAX

PaNel\_SetUp?

#### RESPONSE SYNTAX

PaNel\_SetUp <setup>

#### EXAMPLE (GPIB)

The following saves the oscilloscope's current panel setup in the file PANEL.SET:

```
FILE$ = "PANEL.SET": CMD$="PNSU?":  
CALL IBWRT(SCOPE%,CMD$): CALL IBRDF(SCOPE%,FILE$)
```

Whereas the following recalls the front panel setup, stored previously in the file PANEL.SET, into the oscilloscope:

```
CALL IBWRTF(SCOPE%,FILE$)
```

#### RELATED COMMANDS

\*RCL, \*SAV

### ***CURSORS***

**PARAMETER, PARM**

Command/Query

#### **DESCRIPTION**

This command turns statistics and histograms on or off.

#### **COMMAND SYNTAX**

PARM <type>,[readout]

Type:= CUST, HPAR, VPAR, OFF

Readout:= STAT, HISTICON, BOTH, OFF

Without argument, state of histograms and statistics is unchanged.

Unlike CRMS, PARM does not change the state of cursors or pass/fail.

#### **QUERY SYNTAX**

PARAMETER? OR PARM?

#### **RESPONSE SYNTAX**

PARM <type>,<readout>

#### **EXAMPLE (GPIB)**

The following turns the histograms on:

```
CMD$="PARM CUST,HISTICON";CALL  
IBWRT(SCOPE%,CMD$)
```

<b><i>CURSOR</i></b>	<b>PARAMETER_CLR, PACL</b> Command
----------------------	---------------------------------------

<b>DESCRIPTION</b>	The <code>PARAMETER_CLR</code> command clears all the current parameters from the five-line list used in the Custom and Pass/Fail modes.
--------------------	--

<b>COMMAND SYNTAX</b>	<code>Parameter_Clr</code>
-----------------------	----------------------------

<b>RELATED COMMANDS</b>	<code>PARAMETER_DELETE, PARAMETER_VALUE,</code>
-------------------------	---

## CURSOR

PARAMETER\_CUSTOM, PACU  
Command/Query

### DESCRIPTION

The PARAMETER\_CUSTOM command controls the parameters that have customizable qualifiers and can also be used to assign any parameter for histograms.

**NOTE:** When PAVA? is used to query a Custom parameter, the prefix is returned for consistency. However, the source for the measurement is the one configured using the PACU command..

**TIP:** Use PAVA? to read the measured value of a parameter that was set up with PACU. To determine the correct from of a PACU command in your scope model it is useful to set up the scope manually to the configuration that you want, and then send a PACU query. The response from the scope can be copied straight into your program for use as a query.

### COMMAND SYNTAX

Parameter\_Custom  
<column> , <parameter> , <qualifier>[ , <qualifier> , ...]  
<column> : = 1 to 8  
<parameter> : = {a parameter from the table below or any parameter listed in the PAVA? command}  
<qualifier> : = Measurement qualifier(s) specific to each <param>.  
See below.

CUSTOMIZABLE PARAMETERS ON ALL MODELS		
<param>	Definition	<qualifier> list
AMPL	Amplitude	
AREA	Area	<cyclic>,<source>
BASE	Base	
CYCL	Cycles on screen	
DLY	Delay	

## PART TWO: COMMANDS

DDLY	Delta delay	<source2>
DTLEV	Delta time at level	<slope1>,<pctabs>,<leveltype>,<percentlevel1>,<abslevel1>,<source2>,<slope2>,<pctabs>,<leveltype>,<percentlevel2>,<abslevel2>,<hysteresis>
DUR	duration of acquisition	
DUTY	duty cycle	
DULEV	Duty cycle at level	<slope>,<pcctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
EDLEV	Edges at level	<slope>,<pcctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
FALL82	Fall time 80 % to 20 %	
FALL	Fall time 90 % to 10 %	
FLEV	Fall at levels	<pctabs>,<levelsare>,<highpct>,<highabs>,<pctabs>,<levelsare>,<lowpct>,<lowabs>
FRST	First cursor position	
FREQ	Frequency	
HOLDLEV	Clock to data time	<Clockslope>,<pctabs>,<clocklevelis>,<clockpctlevel>,<clockabslevel>,<source2>,<dataslope>,<pctabs>,<datalevelis>,<datapctlevel>,<dataabslevel>,<clockhysteresis>,<datahysteresis>
LAST	Last cursor position	
MAX	Maximum value	
MEAN	Mean value	<cyclic>
MEDI	Median	<cyclic>
MIN	Minimum value	
PNTS	Period	
NULL	phase difference	
OVSN	Overshoot negative	
OVSP	Overshoot positive	
PKPK	Peak to peak	
PER	Period	
PHASE	Phase difference	

POPATX	Population of bin at x	<horvalue>,<cursorshape>
RISE	Rise time 10% to 90 %	
RISE28	Rise 20 % to 80 %	
RLEV	Rise time at levels	<pctabs>,<levelsare>,<lowpct>,<lowabs>,<pctabs>,<levelsare>,<highpct>,<highabs>
RMS	root mean square	<cyclic>
SETUP	Data edge to clock edge	<Clockslope>,<pctabs>,<clocklevelis>,<clockpctlevel>,<clockabslevel>,<source2>,<dataslope>,<pctabs>,<datalevelis>,<datapctlevel>,<dataabslevel>,<clockhysteresis>,<datahysteresis>
SDEV	Standard deviation	<cyclic>
TLEV	Time at level	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
TOP	Top	
WID	Width	
WIDL	Width at level	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
XMAX	Pos of max data value	
XMIN	Pos if min data value	
XAPK	Nth highest hist peak	<peaknumber>
CUSTn	Custom parameter	

CUSTOMIZABLE PARAMETERS WITH EXTENDED MATH OPTION		
<param>	Definition	<qualifier> list
DTLEV	Delta time at level	<source1>,<slope1>,<level1>,<source2>,<slope2>,<level2>,<hysteresis>
FLEV	fall at level	<source>,<high>,<low>
RLEV	rise at level	<source>,<low>,<high>
TLEV	Time at level	<source>,<slope>,<level>,<hysteresis>

## PART TWO: COMMANDS

CUSTOMIZABLE PARAMETERS WITH SDA & SDM OPTION		
<param>	Definition	<qualifier> list
AVG	Histogram mean	
CROSSPERCENT	Differential crossing %	
DCD		
DPLEV	Delta period at level	<signaltype>,<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>,<usebasefrequency>,<basefrequency>,<stdbasefrequency>
DTLEV	Delta time at levels	<slope1>,<pctabs>,<leveltype>,<percentlevel1>,<abslevel1>,<source2>,<slope2>,<pctabs>,<leveltype>,<percentlevel2>,<abslevel2>,<hysteresis>
DWIDLEV	Delta width at level	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
EDLEV	Edges at level	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
EXTRATIO	Eye level ratios	
EYEAMPL	Eye amplitude	
EYEBER	Eye bit error rate estim	
EYEHEIGHT	TBD	
EYEWIDTH	Eye width	
EYECRW		
EYECROSSINGCALC		
FREQLEV	Frequency at level	<signaltype>,<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>,<usebasefrequency>,<basefrequency>,<stdbasefrequency>
FWHM	Histogram FWHM	
FWXX	Hist peak FW at level	<hfractionht>
HPER	Half period	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
HAMPL	Histogram amplitude	
HBASE	Histogram base level	
HIGH	Histogram right bin	

HMEDI	Histogram median	
HRMS	Histogram rms	
HTOP	Histogram top level	
HOLDLEV	Clock to data edge time	<clockslope>,<pctabs>,<clocklevelis>,<clockpctlevel>,<clockabslevel>,<source2>,<dataslope>,<pctabs>,<datalevelis>,<datapctlevel>,<dataabslevel>,<clockhysteresis>,<datahysteresis>
LOW	Histogram left bin	
MAXP	Histogram highest peak	
MODE	Histogram mode	
ONELEVEL?		
PKS	Histogram no of peaks	
PCTL	Histogram percentile	
PLEV	Period at level	<signaltype>,<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>,<usebasefrequency>,<basefrequency>,<stadbasefrequency>
	PERIODIC JITTER	
POPATX	Histogram bin population	<horvalue>,<cursorshape>
EyeQ?		<patternlength>
RANGE	Histogram range	
SETUP	Data to clock edge time	<Clockslope>,<pctabs>,<clocklevelis>,<clockpctlevel>,<clockabslevel>,<source2>,<dataslope>,<pctabs>,<datalevelis>,<datapctlevel>,<dataabslevel>,<clockhysteresis>,<datahysteresis>
SIGMA	Histogram standard dev	
SKEW	Time clock to clock edge	<clock1slope>,<pctabs>,<clock1levelis>,<clock1pctlevel>,<clock1abslevel>,<source2>,<clock2slope>,<pctabs>,<clock2levelis>,<clock2pctlevel>,<clock2abslevel>,<clock1hysteresis>,<clock2hysteresis>
TIELEV	Time interval error at level	<signaltype>,<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<resultscaling>,<hysteresis>,<usebasefrequency>,<basfrequency>,<stdbasefrequency>
TJ	Total jitter at set BER	
TOTP	Histogram total populate	

## PART TWO: COMMANDS

WIDL	Width at level	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
XAPK	Histogram Nth peak	Peaknumber
ZEROLVL?	Eye diagram zero level	

CUSTOMIZABLE PARAMETERS WITH XMAP OPTION		
<param>	Definition	<qualifier> list
AVG	Histogram mean	
DPLEV	Delta period at level	<signaltype>,<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>,<usebasefrequency>,<basefrequency>,<stdbasefrequency>
DTLEV	Delta time at level	<slope1>,<pctabs>,<leveltype>,<percentlevel1>,<abslevel1>,<source2>,<slope2>,<pctabs>,<leveltype>,<percentlevel2>,<abslevel2>,<hysteresis>
DWIDLEV	Delta width at level	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
EDLEV	Edges at level	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
EXCELPARAM	Parameter using Excel	<source1var>,<source2var>,<outputvar>,<source1headervar>,<source2headervar>,<outputheadervar>,<withheader>,<outputenable>,<source1enable>,<source2enable>,<newsheet>,<advanced>,<worksheetfilename>
EXCELPARAMARITH	Param arithmetic Excel	
FLEV	Frequency at level	<pctabs>,<levelsare>,<highpct>,<hightabs>,<pctabs>,<levelsare>,<lowpct>,<lowabs>
FWHM	Histogram FWHM	
FWXX	Histogram FW at level	<hfractionht>
HPER	Half period	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
HAMPL	Histogram amplitude	
HBASE	Histogram base	
HIGH	Histogram right bin	
HMEDI	Histogram median	
HRMS	Histogram RMS	

HTOP	Histogram top	<hfractionht>
HOLDLEV	Time clock to data edge	<clockslope>,<pctabs>,<clocklevelis>,<clockpctlevel>,<clockabslevel>,<source2>,<dataslope>,<pctabs>,<datalevelis>,<datapctlevel>,<dataabslevel>,<clockhysteresis>,<datahysteresis>
LOW	Histogram left bin	
MATHCADPARAM	Param using Mathcad	<source1var>,<source2var>,<outputvar>,<source1headervar>,<source2headervar>,<outputheadervar>,<withheader>,<outputenable>,<source1enable>,<source2enable>,<newsheet>,<advanced>,<worksheetfilename>
MATHCADPARAMARITH	Param arith Mathcad	
MATLABPARAM	Parameter using MATLAB	
MAXP	Histogram highest peak	
MODE	Histogram mode	
NBPH	Narrow band phase	<frequency>
NBPW	Narrow band power	<frequency>
NUMMODES	Histogram no of peaks	
PCONST	Parameter constant	
PSCRIPT	Param VBS Meas – param	<language>,<code>
PARAMSCRIPT	Param VBS WF – param	<language>,<code>
PKS	Histogram peaks	
PCTL	Histogram percentile	<hpctpop>
PLEV	Period at level	<signaltype>,<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>,<usebasefrequency>,<basefrequency>,<stadbasefrequency>
POPATX	Histogram bin population	<horvalue>,<cursorshape>
RANGE	Histogram range	
SETUP	Time data to clock edge	<clockslope>,<pctabs>,<clocklevelis>,<clockpctlevel>,<clockabslevel>,<source2>,<dataslope>,<pctabs>,<datalevelis>,<datapctlevel>,<dataabslevel>,<clockhysteresis>,<datahysteresis>
SIGMA	Histogram standard dev	

## PART TWO: COMMANDS

SKREW	Time clock to clock edge	<clock1slope>,<pctabs>,<clock1levelis>,<clock1pctlevel>,<clock1abslevel>,<source2>,<clock2slope>,<pctabs>,<clock2levelis>,<clock2pctlevel>,<clock2abslevel>,<clock1hysteresis>,<clock2hysteresis>
TIELEV	Time interval error	<signaltype>,<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<resultscaling>,<hysteresis>,<usebasefrequency>,<basfrequency>,<stdbasefrequency>
TOTP	Histogram total pop	
WIDLW	Width at level	<slope>,<pctabs>,<leveltype>,<percentlevel>,<abslevel>,<hysteresis>
XAPK	Histogram Nth peak	<peaknumber>

CUSTOMIZABLE PARAMETERS WITH XMATH OPTION		
<param>	Definition	<qualifier> list
AVG	Histogram mean	
FWHM	Histogram FWHM	
FWXX	Histogram FW at level	<hfractionht>
HAMPL	Histogram amplitude	
HBASE	Histogram base	
HIGH	Histogram right bin	
HMEDI	Histogram median	
HRMS	Histogram RMS	
HTOP	Histogram top	<hfractionht>
LOW	Histogram left bin	
MAXP	Histogram highest peak	

Where:

<sourceN> : = {C1,C2,C3,C4,F1,F2,F3,F4,F5,F6,F7,F8,TA,TB,TC,TD}. TA through TD are for compatibility with existing software with earlier DSOs. These four mnemonics are not returned by queries.

<slopeN> : = {POS, NEG, FIRST}

<edgeN> : = {POS, NEG}

<clock edge> : = {POS, NEG, ALL}  
<levelN>, <low>, <high> : = 1 to 99 if level is specified in percent (PCT), or  
<levelN>, <low>, <high> : = Level in <sourceN> in the units of the waveform.  
<delay> : = -100 PCT to 100 PCT  
<freq> : = 10 to 1e9 Hz (Narrow Band center frequency)  
<hysteresis> : = 0.01 to 8 divisions  
<length> : = 1e-9 to 0.001 seconds  
<operator> : = {ADD, SUB, MUL, DIV} \*  
<rank> : = 1 to 100  
<threshold> : = 0 to 100 percent  
<angular unit> = {PCT, DEG, RAD}  
<cyclic> = {OFF, ON}

### QUERY SYNTAX

PParameter\_CUstom? <column>

### RESPONSE FORMAT

PParameter\_Custom  
<column> , <parameter> , <qualifier>[ , <qualifier> , ...]

### AVAILABILITY

<sourceN> : = {C3, C4} only on four-channel oscilloscopes.

### EXAMPLE 1

Command Example:

Query/Response Examples:

#### DTLEV

PACU 2,DTLEV,C1,POS,345E-3,C2,NEG,-789E-3

PACU? 2 returns:

PACU 2,DTLEV,C1,POS,345E-3,C2,NEG,-789E-3

PAVA? CUST2 returns:

C2:PAVA CUST2,789 NS

---

\* For Parameter Math option

## PART TWO: COMMANDS

---

### EXAMPLE 2

Command Example:

Query/Response Examples:

#### DDL<sub>Y</sub>

PACU 2,DDL<sub>Y</sub>,C1,C2

PACU? 2 returns:

PACU 2,DDL<sub>Y</sub>,C1,C2

PAVA? CUST2 returns:

C2:PAVA CUST2,123 NS

### EXAMPLE 3

Command Example:

Query/Response Examples:

#### RLE<sub>V</sub>

PACU 3,RLE<sub>V</sub>,C1,2PCT,67PCT

PACU? 3 returns:

PACU 3,RLE<sub>V</sub>,C1,2PCT,67PCT

PAVA? CUST3 returns:

C1:PAVA CUST3,23 MS

### EXAMPLE 4

Command Example:

Query/Response Examples:

#### FLE<sub>V</sub>

PACU 3,FLE<sub>V</sub>,C1,345E-3,122E-3

PACU? 3 returns:

PACU 3,FLE<sub>V</sub>,C1,345E-3,122E-3

PAVA? CUST3 returns:

C1:PAVA CUST3,23 MS

### EXAMPLE 5 (Parameter Math)

Command Example:

Query/Response Examples:

#### CALC<sub>x</sub>

PACU 5,CALC1,AMPL,C3,DIV,AMPL,C2

PACU? 5 returns:

PACU 5,CALC1,AMPL,C3,DIV,AMPL,C2

PAVA? CUST5 returns:

C2:PAVA CUST1,4.884,OK

### RELATED COMMANDS

PARAMETER\_DELETE, PARAMETER\_VALUE

**EXAMPLE RESPONSE** to PACU? 1 where Parameter 1 is a script –  
PACU 1,PARAMSCRIPT,C1,VBSCRIPT,FUNCTION UPDATE()  
TRACELENGTH = INRESULT.SAMPLES  
INDATA = INRESULT.DATAARRAY  
YTOTAL = 0  
XYTOTAL = 0  
FOR K = 0 TO TRACELENGTH - 1  
Y = INDATA(K) : YY = Y \* Y  
YTOTAL = YTOTAL + YY  
XYTOTAL = XYTOTAL + K \* YY  
NEXT  
OUTRESULT.VALUE = XYTOTAL / YTOTAL  
END FUNCTION

CURSOR

PARAMETER\_DELETE, PADL  
Command

DESCRIPTION

The PARAMETER\_DELETE command deletes a parameter at a specified column from the list of parameters used in the Custom mode.

NOTATION		
1	Column 1	of Custom or Pass/Fail display
2	Column 2	of Custom or Pass/Fail display
3	Column 3	of Custom or Pass/Fail display
4	Column 4	of Custom or Pass/Fail display
5	Column 5	of Custom or Pass/Fail display
6	Column 6	of Custom or Pass/Fail display
7	Column 7	of Custom or Pass/Fail display
8	Column 8	of Custom or Pass/Fail display

COMMAND SYNTAX

Parameter\_Delete <column>  
<column> := {1, 2, 3, 4, 5, 6, 7, 8}

EXAMPLE (GPIB)

The following instruction deletes the third test condition in the list:  
CMD\$="PADL 3": CALL IBWRT(SCOPE%,CMD\$)

RELATED COMMANDS

PARAMETER\_CLR, PARAMETER\_VALUE,

CURSOR

PARAMETER\_STATISTICS?, PAST?  
Query

DESCRIPTION

The PARAMETER\_STATISTICS? query returns the current values of statistics for the specified pulse parameter mode and specified statistic for all columns of the parameter display, or all statistics for the specified parameter.

QUERY SYNTAX A

PParameter\_Statistics? <mode>, <statistic>  
<mode>:={CUST, HPAR, VPAR}  
<statistic>:={AVG, LOW, HIGH, SIGMA, SWEEPS, LAST, PARAM}

NOTES	
AVG	average
CUST	custom parameters
HIGH	highest value
HPAR	horizontal standard
VPAR	vertical standard par
LOW	lowest value
PARAM	parameter definition
SIGMA	sigma (standard dev
SWEEPS	number of sweeps a
VPAR	vertical standard par

**NOTE:** If keyword *PARAM* is specified, the query returns the list of the five pairs <parameter\_name>,<source>.

QUERY SYNTAX B

PParameter\_Statistics? <mode>, <param>  
<mode>:={CUST, HPAR, VPAR}  
<param>:={P1, P2, P3, P4, P5, P6, P7, P8}

## PART TWO: COMMANDS

---

### EXAMPLE (GPIB) A

The following reads the average values for all the columns individually:

```
CMD$ = "PAST? CUST, AVG"  
CALL IBWRT (SCOPE%,CMD$)  
CALL IBRD(SCOPE%,RD$)
```

### EXAMPLE (GPIB) B

The following reads all the statistical values for parameter P2:

```
CMD$ = "PAST? CUST, P2"  
CALL IBWRT (SCOPE%,CMD$)  
CALL IBRD(SCOPE%,RD$)
```

### RESPONSE FORMAT A

```
PAST CUST,AVG,290.718E-3 V,389.25E-12 V.S,-  
144.589E-3 V,93.76604E-9 S,290.725E-3  
V,389.25E-12 V.S,-144.589E-3 V,229E-9 V
```

### RESPONSE FORMAT B

```
PAST CUST,P1,AMPL,C1,AVG,290.718E-3  
V,HIGH,297.5E-3 V,LAST,294.2E-3  
V,LOW,278.2E-3 V,SIGMA,3.047E-3 V,SWEEPS,182
```

## CURSOR

PARAMETER\_VALUE?, PAVA?

Query

### DESCRIPTION

The PARAMETER\_VALUE query returns the current value or values of the pulse waveform parameter or parameters and mask tests for the specified trace. Traces do not need to be displayed or selected to obtain the values measured by the pulse parameters or mask tests.

AVAILABLE ON ALL MODELS							
ALL	all parameters	FRST	First cursor position	POPATX	Population of bin at x		
AMPL	Amplitude	FREQ	Frequency	RISE	Rise time 10% to 90 %		
AREA	Area	HOLDLEV	Clock to data time	RISE28	Rise 20 % to 80 %		
BASE	Base	LAST	Last cursor position	RLEV	Rise time at levels		
CYCL	Cycles on screen	MAX	Maximum value	RMS	root mean square		
DLY	Delay	MEAN	Mean value	SETUP	Data edge to clock edge		
DDL	Delta delay	MEDI	Median	SDEV	Standard deviation		
DTLEV	Delta time at level	MIN	Minimum value	TLEV	Time at level		
DUR	duration of acquisition	PNTS	Period	TOP	Top		
DUTY	duty cycle	NULL	phase difference	WID	Width		
DULEV	Duty cycle at level	OVSN	Overshoot negative	WIDL	Width at level		
EDLEV	Edges at level	OVSP	Overshoot positive	XMAX	Pos of max data value		
FALL82	Fall time 80 % to 20 %	PKPK	Peak to peak	XMIN	Pos if min data value		
FALL	Fall time 90 % to 10 %	PER	Period	XAPK	Nth highest hist peak		
FLEV	Fall at levels	PHASE	Phase difference	CUSTn	Custom parameter		
CUSTOM PARAMETERS DEFINED USING PARAMETER_CUSTOM COMMAND*							
CUST1	CUST2	CUST3	CUST4	CUST5	CUST6	CUST7	CUST8

\* The numbers in the terms CUST1 through CUST8 refer to the line numbers of the selected custom parameters.

## PART TWO: COMMANDS

AVAILABLE WITH SDA OPTION			
AVG	Histogram mean	HTOP	Histogram top level
CROSSPERCENT	Differential crossing %	HOLDLEV	Clock to data edge time
DCD		LOW	Histogram left bin
DPLEV	Delta period at level	MAXP	Histogram highest peak
DTLEV	Delta time at levels	MODE	Histogram mode
DWIDLEV	Delta width at level	ONELEVEL?	
EDLEV	Edges at level	PKS	Histogram no of peaks
EXTRATIO	Eye level ratios	PCTL	Histogram percentile
EYEAMPL	Eye amplitude	PLEV	Period at level
EYEBER	Eye bit error rate estim		PERIODIC JITTER
EYEHEIGHT	TBD	POPATX	Histogram bin population
EYEWIDTH	Eye width	EyeQ?	
EYECRW		RANGE	Histogram range
EYECROSSINGCALC		SETUP	Data to clock edge time
FREQLEV	Frequency at level	SIGMA	Histogram standard dev
FWHM	Histogram FWHM	SKEW	Time clock to clock edge
FWXX	Hist peak FW at level	TIELEV	Time interval error at level
HPER	Half period	TJ	Total jitter at set BER
HAMPL	Histogram amplitude	TOTP	Histogram total populate
HBASE	Histogram base level	WIDLV	Width at level
HIGH	Histogram right bin	XAPK	Histogram Nth peak
HMEDI	Histogram median	ZEROLVL?	Eye diagram zero level
HRMS	Histogram rms		

AVAILABLE WITH XMAP OPTION			
AVG	Histogram mean	MATLABPARAM	Parameter using MATLAB
DPLEV	Delta period at level	MAXP	Histogram highest peak
DTLEV	Delta time at level	MODE	Histogram mode
DWIDLEV	Delta width at level	NBPH	Narrow band phase
EDLEV	Edges at level	NBPW	Narrow band power
EXCELPARAM	Parameter using Excel	NUMMODES	Histogram no of peaks
EXCELPARAMARITH	Param arithmetic Excel	PCONST	Parameter constant
FLEV	Frequency at level	PSCRIPT	Param VBS Meas – param
FWHM	Histogram FWHM	PARAMSCRIPT	Param VBS WF – param
FWXX	Histogram FW at level	PKS	Histogram peaks
HPER	Half period	PCTL	Histogram percentile
HAMPL	Histogram amplitude	PLEV	Period at level
HBASE	Histogram base	POPATX	Histogram bin population
HIGH	Histogram right bin	RANGE	Histogram range
HMEDI	Histogram median	SETUP	Time data to clock edge
HRMS	Histogram RMS	SIGMA	Histogram standard dev
HTOP	Histogram top	SKEW	Time clock to clock edge
HOLDLEV	Time clock to data edge	TIELEV	Time interval error
LOW	Histogram left bin	TOTP	Histogram total pop
MATHCADPARAM	Param using Mathcad	WIDLV	Width at level
MATHCADPARAMARITH	Param arith Mathcad	XAPK	Histogram Nth peak

## PART TWO: COMMANDS

### AVAILABLE WITH XMATH OPTION

AVG	Histogram mean	MODE	Histogram mode
FWHM	Histogram FWHM	NBPH	Narrow band phase
FWXX	Histogram FW peak	NBPW	Narrow band power
HAMPL	Histogram amplitude	PKS	Histogram no of peaks
HBASE	Histogram base	PCTL	Histogram percentile
HIGH	Histogram right bin	POPATX	Histogram bin population
HMEDI	Histogram median	RANGE	Histogram range
HRMS	Histogram RMS	SIGMA	Histogram standard dev
HTOP	Histogram top	TOTP	Histogram total pop
LOW	Histogram left bin	XAPK	Histogram Nth peak
MAXP	Histogram max pop		

### PARAMETER COMPUTATION STATES

AV	averaged over several (up to 100) periods	OF	Signal partially in overflow
GT	greater than given value	OK	Deemed to be determined without problem
IV	invalid value (insufficient data provided)	OU	Signal partially in overflow and underflow
LT	less than given value	PT	Window has been period truncated
NP	no pulse waveform	UF	signal partially in underflow

### MASK TEST NAMES

ALL_IN	all points of waveform inside mask (TRUE = 1, FALSE = 0)	SOME_IN	some points of waveform inside mask (TRUE = 1, FALSE = 0)
ALL_OUT	all points of waveform outside mask (TRUE = 1, FALSE = 0)	SOME_OUT	some points of waveform outside mask (TRUE = 1, FALSE = 0)

### QUERY SYNTAX

`<trace> : PParameter_VAlue? [<parameter>,...,<parameter>]`

`<trace> := {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, D, C1, C2, C3, C4}`. TA through TD are for backward compatibility, and are not returned by queries.

**NOTE: When PAVA? Is used to query a Custom parameter, the prefix is returned for consistency. However, the source for the measurement is the one configured using the PACU command.**

`<parameter> :=` See table of parameters.

Alternative forms of query for mask tests:

`<trace> : PParameter_VAlue? <mask_test>, <mask>`

`<mask_test> := {ALL_IN, SOME_IN, ALL_OUT, SOME_OUT}`

`<mask> := {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD}`

TA through TD are included for backward compatibility with software designed for earlier LeCroy DSOs. These four mnemonics will not be returned in response to queries.

### RESPONSE FORMAT

`<trace> : PParameter_VAlue <parameter> , <value> ,`

`<state> [ , ... , <parameter> , <value> , <state> ]`

`<value> :=` A decimal numeric value

`<state> := {OK, AV, PT, IV, NP, GT, LT, OF, UF, OU}`

**NOTE: If <parameter> is not specified, or is equal to ALL, all standard voltage and time parameters are returned followed by their values and states.**

### AVAILABILITY

`<trace> : {C3, C4}` only available on four-channel oscilloscopes.

### EXAMPLE (GPIB)

The following query reads the rise time of Trace B (TB):

```
CMD$="F2:PAVA? RISE": CALL
```

```
IBWRT(SCOPE%, CMD$):
```

```
CALL IBRD(SCOPE%, RD$): PRINT RD$
```

Response message:

TB:PAVA RISE,3.6E-9S,OK

### RELATED COMMANDS

CURSOR\_MEASURE, CURSOR\_SET, PARAMETER\_CUSTOM,  
PARAMETER\_STATISTICS

### MISCELLANEOUS

**PASS\_FAIL, PF**  
Command/Query

#### DESCRIPTION

The PASS\_FAIL command sets up the pass/fail system.

#### COMMAND SYNTAX

PASS\_FAIL <state>[,<logic>[,<stop after>]]

<state>:={ON,OFF}

<logic>:={AllTrue,AllFalse,AnyTrue,AnyFalse,AllQ1ToQ4OrQ5ToQ8,AnyQ1ToQ4AndAnyQ5ToQ8}

<stop after>:={ }

#### QUERY SYNTAX

PASS\_FAIL? Or PF?

#### EXAMPLE (GPIB)

The following sets the pass/fail system to all false, and to stop after 20 acquisitions:

```
CMD$="PF ON,ALLFALSE,20":CALL  
IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

CRMS

### ***CURSOR***

### **PASS\_FAIL\_DO, PFDO**

Command/Query

#### **DESCRIPTION**

The PASS\_FAIL\_DO command defines the desired outcome and the actions that have to be performed after a Pass/Fail test. The PASS\_FAIL\_D? query indicates which actions are currently selected. Note that BEEP, PULS, SCDP and STO are provided for backward compatibility with existing software for earlier DSOs.

#### **NOTATION**

ALARM or BEEP	Emit a beep.
PRINT or SCDP	Make a hard copy.
PULSE or PULS	Emit a pulse on the Aux out connector.
SAVE or STO	Store in memory or media.
STOP	Stop acquisition.

#### **COMMAND SYNTAX**

```
Pass_Fail_DO [<outcome>[,<act>[,<act> ...]]]  
<outcome>:= {PASS, FAIL}  
<act>:= {ALARM, BEEP, PRINT, SCDP, PULSE, PULS, SAVE,  
STO, STOP}
```

#### **QUERY SYNTAX**

```
Pass_Fail_DO?
```

#### **RESPONSE SYNTAX**

```
Pass_Fail_DO [<pass_fail>[,<act>[,<act>  
...]]]  
<pass_fail>:= {PASS, FAIL}  
<act>:= {ALARM, PRINT, PULSE, SAVE, STOP}
```

#### **EXAMPLE (GPIB)**

The following forces the DSO to stop acquiring when the test passes:

```
CMD$="PFDO PASS, STOP  
CALL IBWRT(SCOPE%, CMD$)
```

#### **RELATED COMMANDS**

BUZZER, CURSOR\_MEASURE, CURSOR\_SET, INR,  
PARAMETER\_VALUE, PASS\_FAIL\_MASK

### **DISPLAY**

**PERSIST, PERS**  
Command/Query

#### **DESCRIPTION**

The PERSIST command enables or disables the persistence display mode.

#### **COMMAND SYNTAX**

PERSist <mode>  
<mode> := {ON, OFF}

#### **QUERY SYNTAX**

PERSist?

#### **RESPONSE FORMAT**

PERSist <mode>

#### **EXAMPLE (GPIB)**

The following turns the persistence display ON:

```
CMD$="PERS ON": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

PERSIST\_COLOR, PERSIST\_LAST, PERSIST\_SAT,  
PERSIST\_SETUP

### **DISPLAY**

**PERSIST\_COLOR, PECL**  
Command/Query

#### **DESCRIPTION**

The PERSIST\_COLOR command controls the color rendering method of persistence traces.

The response to the PERSIST\_COLOR? query indicates the color rendering method, Analog Persistence™ or Color Graded Persistence.

See the Operator's Manual.

#### **COMMAND SYNTAX**

Persist\_Color <state>  
<state> := {ANALOG, COLOR\_GRADED, 3D}

#### **QUERY SYNTAX**

Persist\_Color?

#### **RESPONSE FORMAT**

Persist\_Color <state>

#### **EXAMPLE (GPIB)**

The following sets the persistence trace color to an intensity-graded range of the selected trace color:

```
CMD$="PECL ANALOG": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

PERSIST, PERSIST\_LAST, PERSIST\_SAT,  
PERSIST\_SETUP

## CURSOR

PER\_CURSOR\_SET, PECS

Command/Query

### DESCRIPTION

The PER\_CURSOR\_SET command allows you to position one or more of the six independent cursors at a given screen location. The position can be set or queried even if the cursor is currently invisible.

### NOTATION

HABS	Horizontal absolute	VABS	Vertical absolute
HDIF	Horizontal difference	VDIF	Vertical difference
HREF	Horizontal reference	VREF	Vertical reference

### COMMAND SYNTAX

```
<trace>:Per_Cursor_Set
<cursor>,<position>[,<cursor>,<position>,...,<cursor>,<position>]
```

<trace>:={C1, C2, C3, C4, F1 through F8, M1, M2, M3, M4, TA, TB, TC, TD}. TA through TD are included for compatibility with existing software for earlier DSOs. These four mnemonics will not be used in responses to queries.

### QUERY SYNTAX

```
<trace>:Per_Cursor_Set?
```

### RESPONSE FORMAT

```
<trace>:={C1, C2, C3, C4, F1, F2, F3, F4, F5, F6, F7, F8, M1, M2, M3, M4}
```

### EXAMPLE(GPIB)

The following positions the HREF and HDIF cursors at +2.6 DIV and +7.4 DIV respectively, using Channel 2 as a reference:

```
CMD$="C2:PECS HREF,2.6 DIV,HDIF,7.4 DIV"
CALL IBWRT(SCOPE%,CMD$)
```

### RELATED COMMANDS

CURSOR\_MEASURE, CURSOR\_SET, PERSIST, PER\_CURSOR\_VALUE

### **DISPLAY**

**PERSIST\_LAST, PELT**  
Command/Query

#### **DESCRIPTION**

The `PERSIST_LAST` command controls whether or not the last trace drawn in a persistence data map is shown.

The response to the `PERSIST_LAST?` query indicates whether the last trace is shown within its persistence data map.

#### **COMMAND SYNTAX**

```
Persist_Last <state>  
<state> := {ON, OFF}
```

#### **QUERY SYNTAX**

```
Persist_Last?
```

#### **RESPONSE FORMAT**

```
Persist_Last <state>
```

#### **EXAMPLE (GPIB)**

The following ensures the last trace is visible within its persistence data map:

```
CMD$="PELT ON": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

```
PERSIST, PERSIST_COLOR, PERSIST_SAT,  
PERSIST_SETUP
```

### DISPLAY

**PERSIST\_SAT, PESA**  
Command/Query

#### DESCRIPTION

The PERSIST\_SAT command sets the level at which the color spectrum of the persistence display is saturated. The level is specified in terms of percentage (PCT) of the total persistence data map population. A level of 100 PCT corresponds to the color spectrum being spread across the entire depth of the persistence data map. At lower values, the spectrum will saturate (brightest value) at the specified percentage value. The PCT is optional.

The response to the PERSIST\_SAT? query indicates the saturation level of the persistence data maps.

#### COMMAND SYNTAX

Persist\_SAt <trace> , <value> [<trace> , <value>]

<trace> := { C1,C2,C3,C4,F1 , F2 , F3 , F4 , F5 , F6 , F7 , F8 , TA , TB , TC , TD}. TA through TD are for compatibility with existing software with earlier DSOs. These four mnemonics are not returned by queries.

<value> := 0 to 100 PCT

**Note:** The suffix PCT is optional.

#### QUERY SYNTAX

Persist\_SAt?

#### RESPONSE FORMAT

Persist\_SAt <trace> , <value>

#### EXAMPLE (GPIB)

The following sets the saturation level of the persistence data map for channel 3 to be 60 % — 60 % of the data points will be displayed with the color spectrum, with the remaining 40 % saturated in the brightest color:

```
CMD$="PESA C3 , 60 " : CALL IBWRT ( SCOPE% , CMD$ )
```

#### RELATED COMMANDS

PERSIST, PERSIST\_COLOR, PERSIST\_PERS,  
PERSIST\_SETUP

### DISPLAY

**PERSIST\_SETUP, PESU**  
Command/Query

#### DESCRIPTION

The PERSIST\_SETUP command selects the persistence duration of the display, in seconds, in persistence mode. In addition, the persistence can be set either to all traces or only the top two on the screen.

The PERSIST\_SETUP? query indicates the current status of the persistence.

#### COMMAND SYNTAX

Persist\_SetUp <time>,<mode>

<time>:= {0.5, 1, 2, 5, 10, 20, infinite}

<mode>:= {PERTRACE, ALL}

**Note:** This does not support the argument “Top2” of earlier DSOs.

#### QUERY SYNTAX

Persist\_SetUp?

#### RESPONSE FORMAT

Persist\_SetUp <time>,<mode>

#### EXAMPLE (GPIB)

The following sets the variable persistence at 10 seconds on all traces:

```
CMD$="PESU 20,ALL": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

PERSIST, PERSIST\_COLOR, PERSIST\_PERS,  
PERSIST\_SAT

### **STATUS**

**\*PRE**

Command/Query

#### **DESCRIPTION**

The \*PRE command sets the PaRallel poll Enable register (PRE). The lowest eight bits of the Parallel Poll Register (PPR) are composed of the STB bits. \*PRE allows you to specify which bit(s) of the parallel poll register will affect the 'ist' individual status bit.

The \*PRE? query reads the contents of the PRE register. The response is a decimal number which corresponds to the binary sum of the register bits.

#### **COMMAND SYNTAX**

PRE <value>

<value> : = 0 to 65 535

#### **QUERY SYNTAX**

\*PRE?

#### **RESPONSE FORMAT**

\*PRE <value>

#### **EXAMPLE (GPIB)**

The following will cause the 'ist' status bit to become 1 as soon as the MAV bit (bit 4 of STB, i.e. decimal 16) is set, and yields the PRE value 16:

```
CMD$="*PRE 16": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

\*IST

**\*RCL**  
Command

### ***SAVE/RECALL SETUP***

#### **DESCRIPTION**

The \*RCL command sets the state of your instrument, using one of the five non-volatile panel setups, by recalling the complete front panel setup of the oscilloscope. Panel setup 0 corresponds to the default panel setup.

The \*RCL command produces an effect the opposite of the \*SAV command.

If the desired panel setup is not acceptable, the EXecution error status Register (EXR) is set and the EXE bit of the standard Event Status Register (ESR) is set.

#### **COMMAND SYNTAX**

\*RCL <panel\_setup>  
<panel\_setup> : = 0 to 4

#### **EXAMPLE (GPIB)**

The following recalls your instrument setup previously stored in panel setup 3:

```
CMD$="*RCL 3": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

PANEL\_SETUP, \*SAV, EXR

### **SAVE/RECALL SETUP**

**RECALL\_PANEL, RCPN**  
Command

#### **DESCRIPTION**

The RECALL\_PANEL command recalls a front panel setup from the current directory on mass storage.

#### **COMMAND SYNTAX**

ReCall\_PaNel DISK,<device>,FILE,'<filename>'

<device> : = {FLPY, HDD}

<filename> : = A string of up to eight characters with the extension .LSS.

#### **EXAMPLE (GPIB)**

The following recalls the front panel setup from file P012.LSS on the floppy disk:

```
CMD$="RCPN DISK, FLPY, FILE,'P012.LSS':
```

```
CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

PANEL\_SETUP, \*SAV, STORE\_PANEL, \*RCL

**SAVE/RECALL SETUP**

**\*RST**  
Command

**DESCRIPTION**                      The \*RST command initiates a device reset. \*RST sets all eight traces to the GND line and recalls the default setup.

**COMMAND SYNTAX**                      \*RST

**EXAMPLE (GPIB)**                      The following resets the oscilloscope:  
CMD\$="\*RST": CALL IBWRT(SCOPE%,CMD\$)

**RELATED COMMANDS**                      \*CAL, \*RCL

### ACQUISITION

**SAMPLE\_CLOCK, SCLK**  
Command/Query

#### DESCRIPTION

The SAMPLE\_CLOCK command allows you to choose between the internal sample clock and an external sample clock.

#### COMMAND SYNTAX

```
Sample_Clock <state>  
<state>:= {INTERNAL, EXTERNAL}
```

#### QUERY SYNTAX

```
Sample_Clock?
```

#### RESPONSE FORMAT

```
Sample_Clock <state>
```

#### EXAMPLE (GPIB)

The following instruction sets the DSO to use an external clock:

```
CMD$="SCLK EXTERNAL":CALL IBWRT(SCOPE%,CMD$)
```

### SAVE/RECALL SETUP

**\*SAV**  
Command

#### DESCRIPTION

The \*SAV command stores the current state of your instrument in non-volatile internal memory. The \*SAV command stores the complete front panel setup of the oscilloscope at the time the command is issued.

***NOTE: Neither communication parameters (those modified by the commands `COMM_FORMAT`, `COMM_HEADER`, `COMM_HELP`, `COMM_ORDER` and `WAVEFORM_SETUP`), nor enable registers of the status reporting system (\*SRE, \*PRE, \*ESE, INE), are saved when \*SAV is used.***

#### COMMAND SYNTAX

\*SAV <panel\_setup>  
<panel\_setup> : = 1 to 4

#### EXAMPLE (GPIB)

The following saves the current instrument setup in panel setup 3:

```
CMD$="*SAV 3": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

PANEL\_SETUP, \*RCL

### **HARD COPY**

**SCREEN\_DUMP, SCDP**  
Command

#### **DESCRIPTION**

The SCREEN\_DUMP command causes the DSO to send the screen contents to the current hardcopy device. The time-and-date stamp corresponds to the time of the command.

#### **COMMAND SYNTAX**

Screen\_DumP

#### **QUERY SYNTAX**

Screen Dump?

#### **RESPONSE FORMAT**

Screen\_DumP <status>  
<status>:= {OFF}

#### **EXAMPLE (GPIB)**

The following instruction initiates a screen dump:  
CMD\$="SCDP": CALL IBWRT(SCOPE%,CMD\$)

#### **RELATED COMMANDS**

INR, HARDCOPY\_SETUP

ACQUISITION

SEQUENCE, SEQ  
Command/Query

DESCRIPTION

The SEQUENCE command sets the conditions for the sequence mode acquisition. The response to the SEQUENCE? query gives the conditions for the sequence mode acquisition. The argument <max\_size> can be expressed either as numeric fixed point, exponential or using standard suffixes. When the optional suffix NUM is used with the query, the response will be returned in standard numeric format. Sequence mode cannot be used at the same time as random interleaved sampling (RIS). If RIS is on, SEQ ON will turn RIS off.

COMMAND SYNTAX

SEQuence <mode>[, <segments>[, <max\_size>]]  
  
<mode> := {OFF, ON}  
  
<segments> := the right-hand column in the table below:  
  
<max\_size> := {...10e+3, 10.0e+3,...11e+3,...}, for example, in standard numeric format.  
  
Or, alternatively:  
  
= {50, 100, 250, 500, 1000, 2500, 5K, 10K, 25K, 50K, 100K, 250K, 500K, 1M}  
  
However, values not absolutely identical to those listed immediately above will be recognized by the scope as *numerical data* (see the table under this heading in Chapter 1). For example, the scope will recognize 1.0M as 1 millisample. But it will recognize 1.0MA as 1 megasample.

**NOTE: The oscilloscope will adapt the requested <max\_size> to the closest valid value.**

QUERY SYNTAX

SEQuence? [NUM]

RESPONSE FORMAT

SEQuence <mode> , <segments> , <max\_size>  
  
<mode> := {ON, OFF}

### **EXAMPLE (GPIB)**

The following sets the segment count to 43, the maximum segment size to 250 samples, and turns the sequence mode ON:

```
CMD$="SEQ ON,43,250": CALL IBWRT(SCOPE%,CMD$)
```

### **RELATED COMMANDS**

TRIG\_MODE

### **STATUS**

**\*SRE**

Command/Query

#### **DESCRIPTION**

The \*SRE command sets the Service Request Enable register (SRE). This command allows you to specify which summary message bit or bits in the STB register will generate a service request. Refer to the table on page 208 for an overview of the available summary messages.

A summary message bit is enabled by writing a '1' into the corresponding bit location. Conversely, writing a '0' into a given bit location prevents the associated event from generating a service request (SRQ). Clearing the SRE register disables SRQ interrupts.

The \*SRE? query returns a value that, when converted to a binary number, represents the bit settings of the SRE register. Note that bit 6 (MSS) cannot be set and its returned value is always zero.

#### **COMMAND SYNTAX**

\*SRE <value>  
<value> : = 0 to 255

#### **QUERY SYNTAX**

\*SRE?

#### **RESPONSE FORMAT**

\*SRE <value>

#### **EXAMPLE (GPIB)**

The following allows an SRQ to be generated as soon as the MAV summary bit (bit 4, i.e. decimal 16) or the INB summary bit (bit 0, i.e. decimal 1) in the STB register, or both, are set. Summing these two values yields the SRE mask  $16+1 = 17$ .

```
CMD$="*SRE 17": CALL IBWRT(SCOPE%,CMD$)
```

### **STATUS**

**\*STB?**  
Query

#### **DESCRIPTION**

The \*STB? query reads the contents of the 488.1 defined SStatus Byte register (STB), and the Master Summary Status (MSS). The response represents the values of bits 0 to 5 and 7 of the STB register and the MSS summary message.

The response to a \*STB? query is identical to the response of a serial poll except that the MSS summary message appears in bit 6 in place of the RQS message. Refer to the table on page 208 for further details of the status register structure.

#### **QUERY SYNTAX**

\*STB?

#### **RESPONSE FORMAT**

\*STB <value>  
<value> := 0 to 255

#### **EXAMPLE (GPIB)**

The following reads the status byte register:

```
CMD$="*STB?": CALL IBWRT(SCOPE%,CMD$):  
CALL IBRD(SCOPE%,RSP$): PRINT RSP$
```

Response message:

\*STB 0

#### **RELATED COMMANDS**

ALL\_STATUS, \*CLS, \*PRE, \*SRE

## PART TWO: COMMANDS

### ADDITIONAL INFORMATION

STATUS BYTE REGISTER (STB)				
Bit	Bit Value	Bit Name	Description	Note
7	128	DIO7	0   reserved for future use	
6	64	MSS/RQS MSS = 1 RQS = 1	at least 1 bit in STB masked by SRE is 1 service is requested	1. 2.
5	32	ESB	1   an ESR enabled event has occurred	3.
4	16	MAV	1   output queue is not empty	4.
3	8	DIO3	0   reserved	
2	4	VAB	1   a command data value has been adapted	5.
1	2	DIO1	0   reserved	
0	1	INB	1   an enabled internal state change has occurred	6.

#### NOTE: For the above table...

1. **The Master Summary Status (MSS) indicates that the oscilloscope requests service, while the Service Request status — when set — specifies that the oscilloscope issued a service request. Bit position 6 depends on the polling method:**

**Bit 6 = MSS if an \*STB? query is received,  
= RQS if serial polling is conducted.**

2. **Example: If SRE = 10 and STB = 10, then MSS = 1. If SRE = 010 and STB = 100 then MSS = 0.**
3. **The Event Status Bit (ESB) indicates whether or not one or more of the enabled IEEE 488.2 events have occurred since the last reading or clearing of the Standard Event Status Register (ESR). ESB is set if an enabled event becomes true (1).**
4. **The Message Available bit (MAV) indicates whether or not the Output queue is empty. The MAV summary bit is set true (1) whenever a data byte resides in the Output queue.**
5. **The Value Adapted Bit (VAB) is set true (1) whenever a data value in a command has been adapted to the nearest legal value. For instance, the VAB bit would be set if the timebase is redefined as 2.5  $\mu$ s/div since the adapted value is 2  $\mu$ s/div.**
6. **The INternal state Bit (INB) is set true (1) whenever certain enabled internal states are entered. For further information, refer to the INR query.**

### **ACQUISITION**

### **STOP** Command

#### **DESCRIPTION**

The **STOP** command immediately stops the acquisition of a signal. If the trigger mode is **AUTO** or **NORM**, **STOP** will place the oscilloscope in **STOPPED** trigger mode to prevent further acquisition.

#### **COMMAND SYNTAX**

**STOP**

#### **EXAMPLE**

The following stops the acquisition process:

```
CMD$ ="STOP": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

**ARM\_ACQUISITION**, **TRIG\_MODE**, **WAIT**,  
**FORCE\_TRIGGER**

### WAVEFORM TRANSFER

### STORE, STO Command

#### DESCRIPTION

The STORE command stores the contents of the specified trace into one of the internal memories M1 to M4 or to the current directory on mass storage.


#### COMMAND SYNTAX

STOre [<trace>,<dest>]

<trace> := {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD, C1, C2, C3, 4, ALL\_DISPLAYED}. TA through TD are for compatibility with existing software with earlier DSOs. These four mnemonics are not returned by queries.

<dest> := {M1, M2, M3, M4, FILE}

***TIP: If you send the STORE command without an argument, all traces currently enabled in the Store Setup will be stored. Modify this setup using STORE\_SETUP.***



#### AVAILABILITY

<trace> := {C3, C4} only available on four-channel oscilloscopes.

#### EXAMPLE (GPIB)

The following stores the contents of Trace A (TA) into Memory 1 (M1):

```
CMD$="STO F1,M1": CALL IBWRT(SCOPE%,CMD$)
```

The following stores all currently displayed waveforms onto the memory card:

```
CMD$="STO ALL_DISPLAYED, CARD":  
CALL IBWRT(SCOPE%,CMD$)
```

The following executes the storage operation currently defined in the Storage Setup (see command STORE\_SETUP):

```
CMD$="STO": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

STORE\_SETUP

### SAVE/RECALL SETUP

STORE\_PANEL, STPN  
Command

#### DESCRIPTION

The STORE\_PANEL command stores the complete front panel setup of your oscilloscope, at the time the command is issued, into a file on the current directory on mass storage.

**NOTE:** The communication parameters (the parameters modified by commands *COMM\_FORMAT*, *COMM\_HEADER*, *COMM\_HELP*, *COMM\_ORDER* and *WAVEFORM\_SETUP*) and the enable registers associated with the status reporting system (commands *\*SRE*, *\*PRE*, *\*ESE*, *INE*) are not saved by this command.

*If no filename (or an empty string) is supplied, the oscilloscope generates a filename according to its internal rules.*

#### COMMAND SYNTAX

```
STore_PaNel DISK, <device>, FILE, '<filename>'
<device> := {FLPY, HDD}
<filename> := A string of up to eight characters with the extension
.LSS.
```

#### EXAMPLE (GPIB)

The following saves the current oscilloscope setup to floppy disk in a file called "DIODE.LSS":

```
CMD$="STPN DISK, FLPY, FILE, 'DIODE.LSS' ":
CALL IBWRT(SCOPE%, CMD$)
```

#### RELATED COMMANDS

PNSU, \*SAV, RECALL\_PANEL, \*RCL

### WAVEFORM TRANSFER

### STORE\_SETUP, STST

Command/Query

#### DESCRIPTION

The STORE\_SETUP command controls the way in which traces will be stored. Any one trace, or all displayed traces, can be set up for storage, either by auto-storing or by the STORE command. Using auto-store, two modes are available, FILL, which stops when the storage medium is full, and WRAP, which replaces the oldest trace by the latest one. Wrap mode will overwrite any trace file, whether or not it was made during the current session, and whether or not it records the same trace: no file is safe.

#### COMMAND SYNTAX

```
Store_Setup  
[ <trace> , <dest> ] [ , AUTO , <mode> ] [ , FORMAT , <type>  
> ]
```

<trace>:= {C1, C2, C3, C4, F1, F2, F3, F4, F5, F6, F7, F8, M1, M2, M3, M4, TA, TB, TC, TD, ALL\_DISPLAYED}. TA through TD are for compatibility with existing software with earlier DSOs. These four mnemonics are not returned by queries.

<dest>:= {M1, M2, M3, M4, HDD}

<mode>:= {OFF, FILL, WRAP}

<type>:= {ASCII, BINARY, MATHCAD, MATLAB}

#### QUERY SYNTAX

```
Store_Setup?
```

#### RESPONSE FORMAT

```
Store_Setup <trace> , <dest> , AUTO , <mode>
```

#### EXAMPLE (GPIB)

The following enables autostore for channel 1, to be performed until insufficient space remains for another file.

```
CMD$="STST C1 , CARD , AUTO , FILL"
```

```
CALL IBWRT( "SCOPE% , CMD$ )
```

#### RELATED COMMANDS

STORE, INR

### **WAVEFORM TRANSFER**

**TEMPLATE?, TMPL?**  
Query

#### **DESCRIPTION**

The `TEMPLATE?` query produces a copy of the template which formally describes the various logical entities making up a complete waveform. In particular, the template describes in full detail the variables contained in the descriptor part of a waveform.

See Chapter 4 for more on the waveform template, and Appendix II for a copy of the template itself.

#### **QUERY SYNTAX**

`TeMPLate?`

#### **RESPONSE FORMAT**

`TeMPLate "<template>"`

`<template>` := A variable length string detailing the structure of a waveform.

#### **RELATED COMMANDS**

`INSPECT`

### ACQUISITION

**TIME\_DIV, TDIV**  
Command/Query

#### DESCRIPTION

The `TIME_DIV` command modifies the timebase setting. The new timebase setting can be specified with suffixes: NS for nanoseconds, US for microseconds, MS for milliseconds, S for seconds, or KS for kiloseconds. An out-of-range value causes the VAB bit (bit 2) in the STB register (see table on page 208) to be set.

The `TIME_DIV?` query returns the current timebase setting.

#### COMMAND SYNTAX

`Time_DIV <value>`

`<value> :=` See the Operator's Manual for specifications.

The suffix S (seconds) is optional.

#### QUERY SYNTAX

`Time_DIV?`

#### RESPONSE FORMAT

`Time_DIV <value>`

#### EXAMPLE (GPIB)

The following sets the time base to 500  $\mu$ s/div:

```
CMD$="TDIV 500US": CALL IBWRT(SCOPE%,CMD$)
```

The following sets the time base to 2 msec/div:

```
CMD$="TDIV 0.002": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

`TRIG_DELAY, TRIG_MODE`

### **DISPLAY**

**TRACE, TRA**  
Command/Query

#### **DESCRIPTION**

The TRACE command enables or disables the display of a trace. An environment error (see table on page 141) is set if an attempt is made to display more than four waveforms.

The TRACE? query indicates whether the specified trace is displayed or not.

#### **COMMAND SYNTAX**

<trace> : TRAcE <mode>

<trace> : = {C1, C2, C3, C4, F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD}. TA through TD are for compatibility with existing software with earlier DSOs. These four mnemonics are not returned by queries.

<mode> : = {ON, OFF}

#### **QUERY SYNTAX**

<trace> : TRAcE?

#### **RESPONSE FORMAT**

<trace> : TRAcE <mode>

#### **AVAILABILITY**

<trace> := {C3, C4} only on four-channel instruments.

#### **EXAMPLE (GPIB)**

The following displays Trace F1:

```
CMD$="F1:TRA ON": CALL IBWRT(SCOPE%,CMD$)
```

### WAVEFORM STORAGE

**TRANSFER\_FILE, TRFL**  
Command/Query

#### DESCRIPTION

This command allows you to transfer files to and from storage media, or between scope and computer. The command format is used to transfer files from the computer to storage media. The query format is used to transfer files from storage media to computer.

#### COMMAND SYNTAX

TRansfer\_FiLe,<device>,FILE,'name.ext',#9nnnnnnnnnn  
<data><crc>

<device> : = {FLPY,HDD}

<n . . . n> : = file size in bytes

<data> : = file data (arbitrary data block)

<crc> : = 32-bit cyclic redundancy check of <data>

**Note:** Files are read from or written into the current directory only.

#### QUERY SYNTAX

TRFL? DISK,<device>,FILE,'name.ext'

#### RESPONSE FORMAT

TRFL #9nnnnnnnnnn<file content><crc>

#### EXAMPLE (GPIB)

The following instruction reads the file FAVORITE.DSO from the floppy disk:

CMD\$=TRFL,DISK,FLPY,FILE,'FAVORITE.DSO'

CALL IBWRT(SCOPE%,CMD\$)

#### RELATED COMMANDS

### **ACQUISITION**

**\*TRG**  
Command

#### **DESCRIPTION**

The \*TRG command executes an ARM command. \*TRG is the equivalent of the 488.1 GET (Group Execute Trigger) message.

#### **COMMAND SYNTAX**

\*TRG

#### **EXAMPLE (GPIB)**

The following enables signal acquisition:

```
CMD$="*TRG": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

ARM\_ACQUISITION, STOP, WAIT, FORCE\_TRIGGER

### ACQUISITION

**TRIG\_COUPLING, TRCP**  
Command/Query

#### DESCRIPTION

The TRIG\_COUPLING command sets the coupling mode of the specified trigger source.

The TRIG\_COUPLING? query returns the trigger coupling of the selected source.

#### COMMAND SYNTAX

```
<trig_source>:TRig_CouPling <trig_coupling>  
<trig_source>:= {C1, C2, C3,C4,EX, EX10, ETM10}  
<trig_coupling>:= {DC}
```

#### QUERY SYNTAX

```
<trig_source>:TRig_CouPling?
```

#### RESPONSE FORMAT

```
<trig_source>:TRig_CouPling <trig_coupling>
```

#### AVAILABILITY

```
<trig_source>:= {C3, C4} only on four-channel instruments.
```

#### EXAMPLE (GPIB)

The following sets the coupling mode of the trigger source Channel 2 to AC:

```
CMD$="C2:TRCP AC": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

TRIG\_COUPLING, TRIG\_DELAY, TRIG\_LEVEL,  
TRIG\_MODE, TRIG\_SELECT, TRIG\_SLOPE,  
TRIG\_WINDOW

### ACQUISITION

**TRIG\_DELAY, TRDL**  
Command/Query

#### DESCRIPTION

The TRIG\_DELAY command sets the time at which the trigger is to occur in respect of the first acquired data point (displayed at the left-hand edge of the screen).

Positive trigger delays are to be expressed as a percentage of the full horizontal screen. This mode is called pre-trigger acquisition, as data are acquired before the trigger occurs. Negative trigger delays must be given in seconds. This mode is called post-trigger acquisition, as the data are acquired after the trigger has occurred.

If a value outside the range  $-10\,000 \text{ div} \times \text{time/div}$  and 100 % is specified, the trigger time will be set to the nearest limit and the VAB bit (bit 2) will be set in the STB register.

The response to the TRIG\_DELAY? query indicates the trigger time with respect to the first acquired data point. Positive times are expressed as a percentage of the full horizontal screen and negative times in seconds.

#### COMMAND SYNTAX

TRig\_DeLay <value>

<value> : = 0.00 PCT to 100.00 PCT (pretrigger)

–20 PS to –10 MAS (post-trigger)

**NOTE:** The suffix is optional. For positive numbers, the suffix PCT is assumed. For negative numbers, the suffix S is assumed. MAS is the suffix for Ms (megaseconds), useful only for extremely large delays at very slow timebases.

#### QUERY SYNTAX

TRig\_DeLay?

#### RESPONSE FORMAT

TRig\_DeLay <value>

### EXAMPLE (GPIB)

The following sets the trigger delay to  $-20$  s (post-trigger):

```
CMD$="TRDL -20S": CALL IBWRT(SCOPE%,CMD$)
```

### RELATED COMMANDS

```
TIME_DIV, TRIG_COUPLING, TRIG_LEVEL,  
TRIG_MODE, TRIG_SELECT, TRIG_SLOPE
```

### ACQUISITION

**TRIG\_LEVEL, TRLV**  
Command/Query

#### DESCRIPTION

The TRIG\_LEVEL command adjusts the trigger level of the specified trigger source. An out-of-range value will be adjusted to the closest legal value and will cause the VAB bit (bit 2) in the STB register to be set.

The TRIG\_LEVEL? query returns the current trigger level.

#### COMMAND SYNTAX

<trig\_source> : TRig\_LeVel <trig\_level>

<trig\_source> := {C1, C2, C3, C4, EX, EX10, ETM10}

**NOTE:** The suffix *v* is optional.

#### QUERY SYNTAX

<trig\_source> : TRig\_LeVel?

#### RESPONSE FORMAT

<trig\_source> : TRig\_LeVel <trig\_level>

#### AVAILABILITY

<trig\_source> := {C3, C4} only on four-channel instruments.

#### EXAMPLE (GPIB)

The following adjusts the trigger level of Channel 2 to -3.4 V:

```
CMD$="C2:TRLV -3.4V": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

TRIG\_COUPLING, TRIG\_DELAY, TRIG\_MODE,  
TRIG\_SELECT, TRIG\_SLOPE

### ACQUISITION

**TRIG\_MODE, TRMD**  
Command/Query

#### DESCRIPTION

The TRIG\_MODE command specifies the trigger mode.  
The TRIG\_MODE? query returns the current trigger mode.

#### COMMAND SYNTAX

TRig\_MoDe <mode>  
<mode> := {AUTO, NORM, SINGLE, STOP}

#### QUERY SYNTAX

TRig\_MoDe?

#### RESPONSE FORMAT

TRig\_MoDe <mode>

#### EXAMPLE (GPIB)

The following selects the normal mode:  
CMD\$="TRMD NORM": CALL IBWRT(SCOPE%, CMD\$)

#### RELATED COMMANDS

ARM\_ACQUISITION, STOP, TRIG\_SELECT, SEQUENCE,  
TRIG\_COUPLING, TRIG\_LEVEL, TRIG\_SLOPE

## ACQUISITION

**TRIG\_PATTERN, TRPA**  
Command/Query

### DESCRIPTION

The TRIG\_PATTERN command defines a trigger pattern. The command specifies the logic composition of the pattern sources (Channel 1, Channel 2, Channel 3 and Channel 4), as well as the conditions under which a trigger can occur. Note that this command can be used even if the complex trigger mode has not been activated.

<b>L</b>	<b>LOW</b>	<b>H</b>	<b>HIGH</b>
<b>PR</b>	<b>PATTERN PRESENT</b>	<b>AB</b>	<b>PATTERN ABSENT</b>
<b>EN</b>	<b>PATTERN ENTERED</b>	<b>EX</b>	<b>PATTERN EXITED</b>

The TRIG\_PATTERN? query returns the current trigger pattern.

Note: PR and EN, and AB and EX are equivalent.

### COMMAND SYNTAX

TRig\_PAttern [<source>,<state>,...<source>,<state>],STATE,<trigger\_condition>

<source> := {C1, C2, C3, C4, EX}

<state> := {L, H}

<trigger\_condition> := {AB, PR, EX, EN}

Note: If a source state is not specified in the command, the source will be set to the X (= don't care) state. The response sends back only the source states that are either H (= high) or L (= low), ignoring the X states.

### QUERY SYNTAX

TRig\_PAttern? [<source>,<state>,...<source>,<state>],STATE,<trigger\_condition>

<source> := {C1, C2, C3, C4, EX}

<state> := {L, H}

### RESPONSE FORMAT

TRig\_PAttern  
[<source>,<state>,...<source>,<state>],STATE,<trigger\_condition>

### EXAMPLE (GPIB)

The following configures the logic state of the pattern as HLXH (CH 1 = H, CH 2 = L, CH 3 = X, CH 4 = H) and defines the trigger condition as pattern absent (AB).

CMD\$="TRPA C1,H,C2,L,C4,H,STATE,AB":  
CALL IBWRT(SCOPE%,CMD\$)

### RELATED COMMANDS

TRIG\_COUPLING, TRIG\_DELAY, TRIG\_LEVEL,  
TRIG\_MODE, TRIG\_SELECT, TRIG\_SLOPE

ACQUISITION

TRIG\_SELECT, TRSE  
Command/Query

DESCRIPTION

The TRIG\_SELECT command selects the condition that will trigger the acquisition of waveforms. Depending on the trigger type, additional parameters must be specified. These additional parameters are grouped in pairs. The first in the pair names the variable to be modified, while the second gives the new value to be assigned. Pairs can be given in any order and restricted to those variables to be changed.

The TRIG\_SELECT? query returns the current trigger condition.

TRIGGER NOTATION			
DROP	Dropout	QL	Qualifier
EDGE	Edge	SNG	Single source
EV	Event	SQ	State-Qualified
GLIT	Glitch	SR	Source
HT	Hold type	STD	*Standard (Edge Trigger)
HV	Hold value	TEQ	Edge-Qualified
HV2	Second hold value	TI	Time
IL	Interval larger	TL	Time within
INTV	Interval		
IS	Interval smaller		
I2	Interval-width window		
OFF	No hold-off on wait		
PL	Pulse larger		
PS	Pulse smaller		
P2	Pulse-width window		

\* HT and HV do not apply to the Standard Trigger.

### COMMAND SYNTAX

TRig\_SELECT <trig\_type> , SR , <source> , QL , <source> ,

HT , <hold\_type> , HV , <hold\_value> , HV2 , <hold value>

<trig\_type> := {DROP, EDGE, GLIT, INTV, STD, SNG, SQ, TEQ}

<source> := {C1, C2, C3, C4, LINE, EX, EX10, PA, ETM10}

<hold\_type> := {TI, TL, EV, PS, PL, IS, IL, P2, I2, OFF}

<hold\_value> := See Operator's Manual for valid values

**NOTE: The suffix S (seconds) is optional.**

### QUERY SYNTAX

TRig\_SELECT?

### RESPONSE FORMAT

TRig\_SELECT  
<trig\_type> , SR , <source> , HT , <hold\_type> , HV ,  
<hold\_value> , <hold\_value>

➤ HV2 only returned if <hold\_type> is P2 or I2

### AVAILABILITY

<source> : {C3, C4} only available on four-channel instruments.

### EXAMPLE (GPIB)

The following selects the single-source trigger with Channel 1 as trigger source. Hold type and hold value are chosen as “pulse smaller” than 20 ns:

CMD\$="TRSE SNG,SR,C1,HT,PS,HV,20 NS":

CALL IBWRT(SCOPE%,CMD\$)

### ACQUISITION

**TRIG\_SLOPE, TRSL**  
Command/Query

#### DESCRIPTION

The TRIG\_SLOPE command sets the trigger slope of the specified trigger source.

The TRIG\_SLOPE? query returns the trigger slope of the selected source.

#### COMMAND SYNTAX

```
<trig_source> : TRig_SLope <trig_slope>
<trig_source> := {C1, C2, C3, C4, LINE, EX, EX10}
<trig_slope>  := {NEG, POS, ETM10}
```

#### QUERY SYNTAX

```
<trig_source> : TRig_SLope?
```

#### RESPONSE FORMAT

```
<trig_source> : TRig_SLope <trig_slope>
```

#### AVAILABILITY

<trig\_source> := {C3, C4} only available on four-channel oscilloscopes.

#### EXAMPLE (GPIB)

The following sets the trigger slope of Channel 2 to negative:

```
CMD$="C2:TRSL NEG": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

TRIG\_COUPLING, TRIG\_DELAY, TRIG\_LEVEL,  
TRIG\_MODE, TRIG\_SELECT, TRIG\_SLOPE

### **AUTOMATION**

**VBS , VBS**  
Command/Query

#### **DESCRIPTION**

The VBS command allows Automation commands to be sent in the context of an existing program.

The Automation command must be placed within single quotation marks.

The “=” sign may be flanked by spaces for clarity, but these spaces are optional.

#### **COMMAND SYNTAX**

VBS <automation command>

VBS? <Return=automation command>

#### **QUERY SYNTAX**

#### **EXAMPLES with “GPIB” Equivalents**

The following instruction sets Channel 1 vertical scale to 50 mV/Div, the time per division to 500 ns, and the grid mode to Dual:

CMD\$=“VBS ‘app.Acquisition.C1.VerScale=0.05’”

CMD\$ “C1:VDIV 50 MV” (GPIB equivalent)

CMD\$=“VBS ‘app.Horizontal.HorScale = 500e-9’”

CMD\$ “TDIV 0.5e-6” (GPIB equivalent)

CMD\$=“VBS ‘app.Display.GridMode=”Dual”’

CMD\$ “GRID DUAL” (GPIB equivalent)

#### **QUERY SYNTAX**

CMD\$ = “VBS? ‘Return=app.Acquisition.C1.VerScale’”

### **DISPLAY**

### **VERT\_MAGNIFY, VMAG**

Command/Query

#### **DESCRIPTION**

The VERT\_MAGNIFY command vertically expands the specified trace. The command is executed even if the trace is not displayed.

The maximum magnification allowed depends on the number of significant bits associated with the data of the trace.

The VERT\_MAGNIFY? query returns the magnification factor of the specified trace.

#### **COMMAND SYNTAX**

`<trace>:Vert_MAGnify <factor>`

`<trace> := {F1,F2,F3,F4,F5,F6,F7,F8,TA,TB,TC,TD}` TA through TD are included for backward compatibility with software designed for earlier LeCroy DSOs. These four mnemonics will not be returned in response to queries.

`<factor> := 4.0E-3 to 50 (maximum)`

#### **QUERY SYNTAX**

`<trace>:Vert_MAGnify?`

#### **RESPONSE FORMAT**

`<trace>:Vert_MAGnify <factor>`

#### **EXAMPLE**

The following enlarges the vertical amplitude of Trace A by a factor of 3.45 with respect to its original amplitude:

```
CMD$="TA:VMAG 3.45": CALL IBWRT(SCOPE%,CMD$)
```

#### **RELATED COMMANDS**

VERT\_POSITION

### DISPLAY

VERT\_POSITION, VPOS  
Command/Query

#### DESCRIPTION

The VERT\_POSITION command adjusts the vertical position of the specified trace on the screen. It does not affect the original offset value obtained at acquisition time.

The VERT\_POSITION? query returns the current vertical position of the specified trace.

#### COMMAND SYNTAX

<trace> : Vert\_POSITION <display\_offset>

<trace> : = {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD}  
} TA through TD are included for backward compatibility with software designed for earlier LeCroy DSOs. These four mnemonics will not be returned in response to queries.

<display\_offset> : = -5900 to +5900 DIV

**NOTE: The suffix DIV is optional. The limits depend on the current magnification factor, the number of grids on the display, and the initial position of the trace.**

#### QUERY SYNTAX

<trace> : Vert\_POSition?

#### RESPONSE FORMAT

<trace> : Vert\_POSITION <display\_offset>

#### EXAMPLE

The following shifts Trace A (TA) upwards by +3 divisions relative to the position at the time of acquisition:

```
CMD$="F1:VPOS 3DIV": CALL IBWRT(SCOPE%,CMD$)
```

#### RELATED COMMANDS

VERT\_MAGNIFY

### ACQUISITION

**VOLT\_DIV, VDIV**  
Command/Query

#### DESCRIPTION

The VOLT\_DIV command sets the vertical sensitivity in Volts/div. The VAB bit (bit 2) in the STB register (see table on page 208) is set if an out-of-range value is entered.

The probe attenuation factor is not taken into account for adjusting vertical sensitivity.

The VOLT\_DIV? query returns the vertical sensitivity of the specified channel.

#### COMMAND SYNTAX

<channel> : Volt\_DIV <v\_gain>

<channel> := {C1, C2, C3, C4}

<v\_gain> := See Operator's Manual for specifications.

**NOTE:** The prefix V is optional.

#### QUERY SYNTAX

<channel> : Volt\_DIV?

#### RESPONSE FORMAT

<channel> : Volt\_DIV <v\_gain>

#### AVAILABILITY

<channel> := {C3, C4} only available on four-channel oscilloscopes.

#### EXAMPLE

The following sets the vertical sensitivity of channel 1 to 50 mV/div:

```
CMD$="C1:VDIV 50MV": CALL IBWRT(SCOPE%,CMD$)
```

### **STATUS**

**\*WAI**  
Command

### **DESCRIPTION**

The \*WAI (WAI to continue) command, required by the IEEE 488.2 standard, has no effect on the oscilloscope, as the instrument only starts processing a command when the previous command has been entirely executed.

### **COMMAND SYNTAX**

\*WAI

### **RELATED COMMANDS**

\*OPC

### ACQUISITION

### WAIT Command

#### DESCRIPTION

The WAIT command prevents your instrument from analyzing new commands until the oscilloscope has completed the current acquisition. The optional argument specifies the timeout (in seconds) after which the scope will stop waiting for new acquisitions. If <t> is not given, or if <t> = 0.0, the scope will wait indefinitely.

#### COMMAND SYNTAX

WAIT [<t>]

<t>:=timeout in seconds (0.0 to 1000.0, default is indefinite)

#### EXAMPLE (GPIB)

```
send: "TRMD SINGLE"  
loop {send: "ARM;WAIT;C1:PAVA?MAX"  
read response  
process response  
}
```

This example finds the maximum amplitudes of several signals acquired one after another. ARM starts a new data acquisition. The WAIT command ensures that the maximum is evaluated for the newly acquired waveform.

C1:PAVA?MAX instructs the oscilloscope to evaluate the maximum data value in the Channel 1 waveform.

#### RELATED COMMANDS

\*TRG, TRIG\_MODE, ARM

### WAVEFORM TRANSFER

**WAVEFORM, WF**  
Command/Query

#### DESCRIPTION

A **WAVEFORM** command transfers a waveform from the controller to the oscilloscope, whereas a **WAVEFORM?** query transfers a waveform from the oscilloscope to the controller.

**WAVEFORM** stores an external waveform back into the oscilloscope's internal memory. A waveform consists of several distinct entities:

the descriptor (**DESC**)

the user text (**TEXT**)

the time (**TIME**) descriptor

the data (**DAT1**) block, and, optionally

a second block of data (**DAT2**).

See Chapter 4 for further information on waveform structure.

**NOTE: You can restore to the oscilloscope only complete waveforms queried with **WAVEFORM? ALL**.**

The **WAVEFORM?** query instructs the oscilloscope to transmit a waveform to the controller. The entities can be queried independently. If the “**ALL**” parameter is specified, all four or five entities are transmitted in one block in the order enumerated above.

**NOTE: The format of the waveform data depends on the current settings specified by the last **WAVEFORM\_SETUP**, **COMM\_ORDER** and **COMM\_FORMAT** commands.**

#### COMMAND SYNTAX

<memory> : WaveForm ALL <waveform\_data\_block>

<memory> : = {M1, M2, M3, M4}

<waveform\_data\_block> : = Arbitrary data block (see Chapter 5).

## PART TWO: COMMANDS

---

### QUERY SYNTAX

<trace> : WaveForm? <block>

<trace> := {F1, F2, F3, F4, F5, F6, F7, F8, TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, C3, C4}. TA through TD are for compatibility with existing software with earlier DSOs. These four mnemonics are not returned by queries.


<block> := {DESC, TEXT, TIME, DAT1, DAT2, ALL}

If you do not give a parameter, ALL will be assumed.

### RESPONSE FORMAT

<trace> : WaveForm <block> , <waveform\_data\_block>

***TIP: It may be convenient to disable the response header if the waveform is to be restored. See the COMM\_HEADER command for further details.***



### AVAILABILITY

<trace> := {C3, C4} only available on four-channel oscilloscopes.

### EXAMPLES (GPIB)

The following reads the block DAT1 from Memory 1 and saves it in the file "MEM1.DAT". The path header "M1:" is saved together with the data.

```
FILE$ = "MEM1.DAT"
CMD$ = "M1:WF? DAT1"
CALL IBWRT(SCOPE%, CMD$)
CALL IBRDF(SCOPE%, FILE$)
```

In the following example, the entire contents of Channel 1 are saved in the file "CHAN1.DAT". The path header "C1:" is skipped to ensure that the data can later be recalled into the oscilloscope.

```
FILE$="CHAN1.DAT":RD$=SPACE$(3)
CMD$="CHDR SHORT; C1:WF?"
CALL IBWRT(SCOPE%, CMD$)
CALL IBRD(SCOPE%, RD$) Skip first 3 characters "C1:"
CALL IBRDF(SCOPE%, FILE$) Save data in file
"CHAN1.DAT"
```

The following illustrates how the waveform data saved in the preceding example can be recalled into Memory 1:

```
FILE$ = "CHAN1.DAT"
CMD$ = "M1:"
CALL IBEOT(SCOPE%, 0) disable EOI
```

```
CALL IBWRT( SCOPE% , CMD$ )  
CALL IBEOT( SCOPE% , 1 ) re-enable EOI  
CALL IBWRTF( SCOPE% , FILE$ )
```

The “M1:” command ensures that the active waveform is “M1”. When the data file is sent to the oscilloscope, it first sees the header “WF” (the characters “C1:” having been skipped when reading the file) and assumes the default destination “M1”.

### **RELATED COMMANDS**

```
INSPECT, COMM_FORMAT, COMM_ORDER,  
FUNCTION_STATE, TEMPLATE, WAVEFORM_SETUP,  
WAVEFORM_TEXT
```

WAVEFORM TRANSFER

WAVEFORM\_SETUP, WFSU  
Command/Query

**DESCRIPTION** The WAVEFORM\_SETUP command specifies the amount of data in a waveform to be transmitted to the controller. The command controls the settings of the parameters listed below.

NOTATION			
FP	first point	NP	number of points
SN	segment number	SP	Sparsing

**Sparsing (SP):** The sparsing parameter defines the interval between data points. For example:

- SP = 0 sends all data points
- SP = 1 sends all data points
- SP = 4 sends every 4th data point

**Number of points (NP):** The number of points parameter indicates how many points should be transmitted. For example:

- NP = 0 sends all data points
- NP = 1 sends 1 data point
- NP = 50 sends a maximum of 50 data points
- NP = 1001 sends a maximum of 1001 data points

**First point (FP):** The first point parameter specifies the address of the first data point to be sent. For waveforms acquired in sequence mode, this refers to the relative address in the given segment. For example:

- FP = 0 corresponds to the first data point
- FP = 1 corresponds to the second data point
- FP = 5000 corresponds to data point 5001

### Segment number (SN):

The segment number parameter indicates which segment should be sent if the waveform was acquired in sequence mode. This parameter is ignored for non-segmented waveforms. For example:

SN = 0          all segments  
SN = 1          first segment  
SN = 23        segment 23

The WAVEFORM\_SETUP? query returns the transfer parameters currently in use.

### COMMAND SYNTAX

WaveForm\_SetUp  
SP, <sparsing>, NP, <number>, FP, <point>, SN, <segment>

**NOTE:** After power-on, all values are set to 0 (i.e. entire waveforms will be transmitted without sparsing).

**Parameters are grouped in pairs. The first of the pair names the variable to be modified, while the second gives the new value to be assigned. Pairs can be given in any order and restricted to those variables to be changed.**

### QUERY SYNTAX

WaveForm\_SetUp?

### RESPONSE FORMAT

WaveForm\_SetUp  
SP, <sparsing>, NP, <number>, FP, <point>, SN, <segment>

### EXAMPLE (GPIB)

The following instructs every 3rd data point (SP=3) starting at address 200 to be transferred:

```
CMD$="WFSU SP,3,FP,200": CALL  
IBWRT(SCOPE%,CMD$)
```

### RELATED COMMANDS

INSPECT, WAVEFORM, TEMPLATE

§ § §