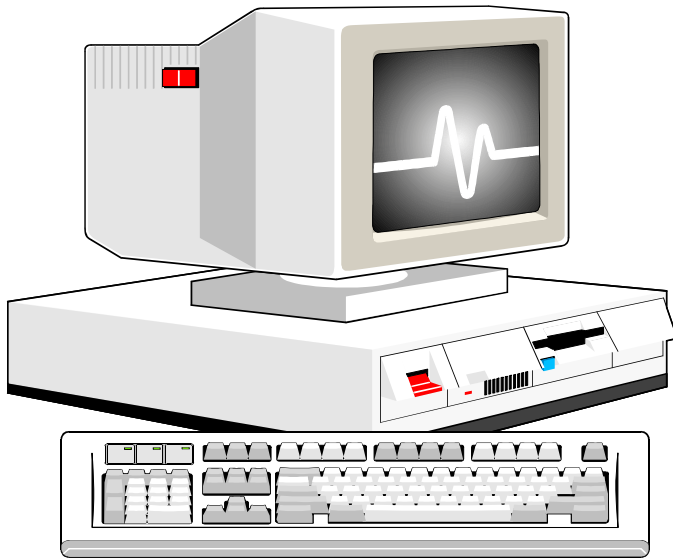
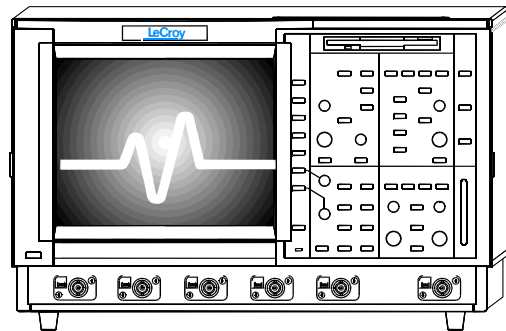




# **Remote Control Manual LeCroy 9300 & LC Oscilloscopes**



**Revision P**

**LeCroy Corporation**

700 Chestnut Ridge Road  
Chestnut Ridge, NY 10977-6499  
Tel: (845) 578 6020, Fax: (845) 578 5985

**Internet:** [www.lecroy.com](http://www.lecroy.com)

© 2001 by LeCroy Corporation. All rights reserved. Information in this publication supersedes all earlier versions. Specifications subject to change.

LeCroy, ProBus and SMART Trigger are registered trademarks of LeCroy Corporation. Centronics is a registered trademark of Data Computer Corp. Epson is a registered trademark of Epson America Inc. I<sup>2</sup>C is a trademark of Philips. Mathcad is a registered trademark of MATHSOFT Inc. MATLAB is a registered trademark of The MathWorks, Inc. Microsoft, MS and Microsoft Access are registered trademarks, and Windows and NT trademarks, of Microsoft Corporation. PowerPC is a registered trademark of IBM Microelectronics. DeskJet, ThinkJet, QuietJet, LaserJet, PaintJet, HP 7470 and HP 7550 are registered trademarks of Hewlett-Packard Company.

Manufactured under an ISO 9000  
Registered Quality Management  
System  
Visit [www.lecroy.com](http://www.lecroy.com) to view the  
certificate.



This electronic product is subject to disposal and recycling regulations that vary by country and region. Many countries prohibit the disposal of waste electronic equipment in standard waste receptacles. For more information about proper disposal and recycling of your LeCroy product, please visit [www.lecroy.com/recycle](http://www.lecroy.com/recycle).

## Contents

---

### Chapter 1 — Overview of Remote Control

|                                                |     |
|------------------------------------------------|-----|
| How to Operate the Oscilloscope Remotely ..... | 1-1 |
|------------------------------------------------|-----|

### Chapter 2 — GPIB

|                                           |      |
|-------------------------------------------|------|
| Communication via the GPIB Bus.....       | 2-1  |
| Instrument Polls .....                    | 2-13 |
| Driving Hardcopy Devices on the GPIB..... | 2-19 |
| Printing by GPIB Controller.....          | 2-20 |

### Chapter 3 — RS-232-C

|                                              |     |
|----------------------------------------------|-----|
| Using the RS-232-C Port.....                 | 3-1 |
| Simulating GPIB Commands with RS-232-C ..... | 3-6 |

### Chapter 4 — Waveform

|                                               |      |
|-----------------------------------------------|------|
| Understanding Waveforms.....                  | 4-1  |
| Using the INSPECT? Query.....                 | 4-4  |
| WAVEFORM?, Related Commands, and Blocks ..... | 4-6  |
| High-Speed Waveform Transfer.....             | 4-15 |

### Chapter 5 — Status Registers

|                             |     |
|-----------------------------|-----|
| Using Status Registers..... | 5-1 |
|-----------------------------|-----|

### SYSTEM COMMANDS

|                                       |    |
|---------------------------------------|----|
| About These Commands & Queries.....   | 1  |
| ... Tabled By Long Form .....         | 3  |
| ... Tabled By Subsystem .....         | 8  |
| <i>The Commands and Queries</i> ..... | 12 |

### Appendix A — GPIB Program Examples

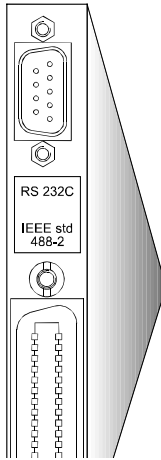
### Appendix B — Waveform Template

### Index



BLANK PAGE

## How to Operate the Oscilloscope Remotely



**Illustrated above and below-right, the GPIB (IEEE Std 488-2) and RS-232-C ports found on the back of LeCroy oscilloscopes, used for connecting the instrument to an external controller. They are vertically or horizontally arranged on the back panel according to model.**

### GPIB Standard

Your LeCroy oscilloscope can of course be operated *manually*, using the front-panel controls (see the *accompanying Operator's Manual*). But it can also be operated *remotely* by means of an external controller. Normally, this controller will be a computer. However, it may be a simple terminal.

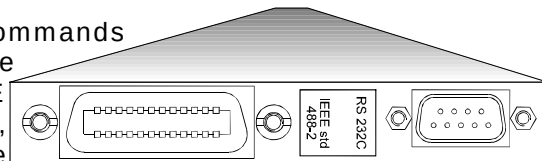
The present manual describes how to remotely control the oscilloscope. Its main section provides the system commands for executing the instrument's functions from an external controller.

Remote control is done using either the GPIB (General Purpose Interface Bus) — labeled "IEEE Std 488-2" — or the RS-232-C communication port on the rear panel of the oscilloscope. The instrument can be fully controlled in remote mode, the only actions not able to be performed being the powering-on of the oscilloscope and the setting of remote addresses.

In this chapter, the basic remote control concepts common to both GPIB and RS-232-C are introduced. Also presented is a brief description of the remote control messages. The following two chapters set out how to send program messages over the GPIB and RS-232-C interfaces, respectively. Chapter 4 offers a detailed description and run-through of the transfer and formatting of waveforms. While Chapter 5 explains the use of status bytes for error reporting.

The special *System Commands* section provides a complete directory and description of the system commands. And the Appendices offer GPIB Program Examples (Appendix A) and a Waveform Template (Appendix B).

The remote commands conform to the GPIB IEEE 488.2\* standard, which may be



\* ANSI/IEEE Std. 488.2-1987, *IEEE Standard Codes, Formats, Protocols, and Common Commands*. The Institute of Electrical and Electronics Engineers Inc., 345 East 47th Street, New York, NY 10017, USA.



## About Remote Control

---

considered as an extension of the IEEE 488.1 standard, dealing mainly with electrical and mechanical issues. The IEEE 488.2 recommendations have also been adopted for RS-232-C communications wherever applicable.

### Program Messages

To control the oscilloscope remotely, the program messages sent from the external controller must conform to precise format structures. The oscilloscope will execute all program messages sent in the correct form, but will ignore those where errors are detected.

Warning or error messages are normally not reported unless the controller explicitly examines the relevant status register. Or if the status-enable registers have been set so that the controller can be interrupted when an error occurs.

During the development of the control program it is possible to observe all remote control transactions, including error messages, on an external monitor connected to the RS-232-C port. *Refer to the command "COMM\_HELP" for further details.*

### Commands and Queries

Program messages consist of either one or several commands or queries.

Whereas the command directs the instrument to change its state — its timebase or vertical sensitivity, for example — the query asks the instrument about that state. Very often, the same mnemonic is used for a command and a query, the query being identified by a <?> after the last character.

For example, to change the timebase to 2 ms/div, the controller sends the following command to the instrument:

**TIME\_DIV 2 MS**

To ask the instrument about its timebase, this query should be sent:

**TIME\_DIV?**

A query causes the instrument to send a response message. The control program should read this message with a 'read' instruction to the GPIB or RS-232-C interface of the controller. The response message to the query above might be:

**TIME\_DIV 10 NS**

The portion of the query preceding the question mark is repeated as part of the response message. If desired, this text may be suppressed with the command "COMM\_HEADER".

Depending on the state of the instrument and the computation to be done, up to several seconds may pass before a response is received. Command interpretation does not have priority over other oscilloscope activities. It is therefore judicious to set the controller IO timeout conditions to three or more seconds. In addition, it should always be remembered that an incorrect query message will not generate a response message.

### Program Message Form

An instrument is remotely controlled with program messages that consist of one or several commands or queries, separated by semicolons <;> and ending in a terminator:

```
<command/query>;.....;<command/query>  
<terminator>
```

Upper or lower-case characters or both can be used in program messages.

The instrument does not decode incoming program messages before receiving a terminator. The exception to this is when the program message is longer than the 256 byte input buffer: the oscilloscope will start analyzing the message when the buffer is full. Commands and queries are executed in the order in which they are transmitted.

In GPIB mode, the following are valid terminators:

|            |                                                                                    |
|------------|------------------------------------------------------------------------------------|
| <NL>       | New-line character (i.e. the ASCII new-line character, whose decimal value is 10). |
| <NL> <EOI> | New-line character with a simultaneous <EOI> signal.                               |
| <EOI>      | <EOI> signal together with the last character of the program message.              |

**Note: The <EOI> signal is a dedicated GPIB interface line which can be set with a special call to the GPIB interface driver. Refer to the GPIB interface manufacturer's manual and support programs.**

## About Remote Control



The <NL> <EOI> terminator is always used in response messages sent by the instrument to the controller.

In RS-232-C, the terminator may be defined by the user with the command "COMM\_RS232". The default value is <CR>, i.e. the ASCII carriage return character, the decimal value of which is 13.

### Examples

#### GRID DUAL

This program message consists of a single command that instructs the instrument to display a dual grid. The terminator is not shown, as it is usually automatically added by the interface driver routine writing to the GPIB (or RS-232).

#### DZOM ON; DISPLAY OFF; DATE?

This program message consists of two commands, followed by a query. They instruct the instrument to turn on the multi-zoom mode, turn off the display, and then ask for the current date. Again, the terminator is not shown.

### Command/Query Form

The general form of a command or a query consists of a command header <header> optionally followed by one or several parameters <data> separated by commas:

<header>[?] <data>, ..., <data>

The notation [?] shows that the question mark is optional (turning the command into a query). The detailed listing of all commands in *System Commands* indicates which may also be queries. There is a space between the header and the first parameter. There are commas between parameters.

### Example

#### DATE 15, JAN, 1993, 13, 21, 16

This command instructs the oscilloscope to set its date and time to 15 JAN 1993, 13:21:16. The command header "DATE" indicates the action, the 6 data values specify it in detail.

### Header

The header is the mnemonic form of the operation to be performed by the oscilloscope. All command mnemonics are listed in alphabetic order in the System Commands section.

The majority of the command/query headers have a long form for optimum legibility and a short form for better transfer and decoding speed. The two forms are fully equivalent and can be



used interchangeably. For example, the following two commands for switching to the automatic trigger mode are fully equivalent:

**TRIG\_MODE AUTO** and **TRMD AUTO**

Some command/query mnemonics are imposed by the IEEE 488.2 standard. They are standardized so that different instruments present the same programming interface for similar functions. All these mnemonics begin with an asterisk <\*>. For example, the command “\*RST” is the IEEE 488.2 imposed mnemonic for resetting the instrument, whereas “\*TST?” instructs the instrument to perform an internal self-test and to report the outcome.

## Header path

Some commands or queries apply to a sub-section of the oscilloscope — for example, a single input channel or a trace on the display. In such cases, the header must be preceded by a path name that indicates the channel or trace to which the command applies. The header path normally consists of a two-letter path name followed by a colon <:> immediately preceding the command header.

One of the waveform traces can usually be specified in the header path (refer to the individual commands listed in *System Commands* for details of the values applying to given command headers):

|                       |                               |
|-----------------------|-------------------------------|
| <b>C1, C2</b>         | Channels 1 and 2              |
| <b>C3, C4</b>         | Channels 3 and 4 <sup>†</sup> |
| <b>M1, M2, M3, M4</b> | Memories 1, 2, 3, 4           |
| <b>TA, TB, TC, TD</b> | Traces A, B, C and D          |
| <b>EX, EX10, EX5</b>  | External trigger              |
| <b>LINE</b>           | LINE source for trigger       |

## Example

**C1:0FST -300 MV**

Commands to set the offset of Channel 1 to –300 mV.

Header paths need only be specified once. Subsequent commands with header destinations not indicated are assumed to refer to the last defined path. For example, the following commands are identical:

---

<sup>†</sup> On four-channel instruments only.



## About Remote Control

---

**C2:VDIV?; C2:OFST?** What is the vertical sensitivity and the offset of channel 2?

**C2:VDIV?; OFST?** Same as above, without repeating the path.

### Data

Whenever a command/query uses additional data values, the values are expressed in terms of ASCII characters. There is a single exception: the transfer of waveforms with the command/query "WAVEFORM", where the waveform may be expressed as a sequence of binary data values. *Chapter 4 gives a detailed explanation of waveform format.*

ASCII data can have the form of character, numeric, string or block data.

### Character data

These are simple words or abbreviations for the indication of a specific action.

#### Example

##### **DUAL\_ZOOM ON**

Here, the data value "ON" commands that the dual-zoom mode be turned on (the data value "OFF" in such a case will obviously have the opposite effect).

However, this can become more complex. In some commands, where as many as a dozen different parameters are able to be specified, or where not all the parameters are applicable at the same time, the format requires pairs of data values. The first value names the parameter to be modified, while the second gives its value. Only those parameter pairs changed need indicating:

#### Example

##### **HARDCOPY\_SETUP DEV, EPSON, PORT, GPIB**

Here, two pairs of parameters are specified. The first specifies the device as the EPSON printer (or compatible) and the second indicates the GPIB port. While the command "HARDCOPY\_SETUP" allows many more parameters, they are either not relevant for printers or are left unchanged.

### Numeric Data

The numeric data type is used to enter quantitative information. Numbers can be entered as integers or fractions, or in exponential representation:

## 9300 &amp; LC Series

|                     |                                                                  |
|---------------------|------------------------------------------------------------------|
| <b>TA:VPOS -5</b>   | Move the displayed trace of Trace A downwards by five divisions. |
| <b>C2:OFST 3.56</b> | Set the DC offset of Channel 2 to 3.56 V.                        |
| <b>TDIV 5.0E-6</b>  | Adjust the timebase to 5 $\mu$ s/div.                            |

**Note:** Numeric values may be followed by multipliers and units, modifying the value of the numerical expression. The following mnemonics are recognized:

|    |       |        |    |       |        |
|----|-------|--------|----|-------|--------|
| EX | 1E18  | Exa-   | PE | 1E15  | Peta-  |
| T  | 1E12  | Tera-  | G  | 1E9   | Giga-  |
| MA | 1E6   | Mega-  | K  | 1E3   | kilo-  |
| M  | 1E-3  | milli- | U  | 1E-6  | micro- |
| N  | 1E-9  | nano-  | PI | 1E-12 | pico-  |
| F  | 1E-15 | femto- | A  | 1E-18 | atto-  |

**Examples**

There are many ways of setting the timebase of the instrument to 5  $\mu$ s/div:

**TDIV 5E-6** Exponential notation, without any suffix.

**TDIV 5 US** Suffix multiplier "U" for 1E-6, with the (optional) suffix "S" for seconds.

or

**TDIV 5000 NS**

**TDIV 5000E-3 US**

**String Data**

This data type enables the transfer of a (long) string of characters as a single parameter. String data are formed by simply enclosing any sequence of ASCII characters between single or double quotation marks:



## About Remote Control

---

### **MESSAGE 'Connect probe to point J3'**

The instrument displays this message in the Message field above the grid.

### **Block Data**

These are binary data values coded in hexadecimal ASCII, i.e. 4-bit nibbles translated into the digits 0,...9, A,...F and transmitted as ASCII characters. They are used only for the transfer of waveforms (Command "WAVEFORM") and of the instrument configuration (Command "PANEL\_SETUP")

### **Response Message Form**

The instrument sends a response message to the controller, as an answer to a query. The format of such messages is the same as that of program messages, i.e. individual responses in the format of commands, separated by semicolons <;> and ending with a terminator. They can be sent back to the instrument in the form in which they are received, to be accepted as valid commands. In GPIB response messages, the <NL> <EOI> terminator is always used.

For instance, if the controller sends the program message:

**TIME\_DIV?;TRIG\_MODE NORM;C1:COUPLING?**  
(terminator not shown).

The instrument might respond as follows:

**TIME\_DIV 50 NS;C1:COUPLING D50** (terminator not shown).

The response message refers only to the queries: "TRIG\_MODE" is left out. If this response is sent back to the instrument, it is a valid program message for setting its timebase to 50 ns/div and the input coupling of Channel 1 to 50 W.

Whenever a response is expected from the instrument, the control program must instruct the GPIB or RS-232-C interface to read from the instrument. If the controller sends another program message without reading the response to the previous one, the response message in the output buffer of the instrument is discarded.

The instrument uses somewhat stricter rules for response messages than for the acceptance of program messages. Whereas the controller may send program messages in upper or lower case characters, response messages are always returned

in upper case. Program messages may contain extraneous spaces or tabs (white space): response messages will not. And while program messages may contain a mixture of short and long command/query headers, response messages always use short headers by default.

However, the instrument can be forced, using the command "COMM\_HEADER", to use long headers, or no headers at all. If the response header is omitted, the response transfer time is minimized, but such a response will not be able to be sent back to the instrument. Suffix units are also suppressed in the response.

If the trigger slope of Channel 1 is set to negative, the query "C1:TRSL?" might yield the following responses:

|                          |                      |
|--------------------------|----------------------|
| <b>C1:TRIG_SLOPE NEG</b> | header format: long  |
| <b>C1:TRSL NEG</b>       | header format: short |
| <b>NEG</b>               | header format: off   |

Waveforms which are obtained from the instrument using the query "WAVEFORM?" constitute a special kind of response message. Their exact format can be controlled via the "COMM\_FORMAT" and "COMM\_ORDER" commands.

## Communication via the GPIB Bus

This chapter describes how to remotely control the oscilloscope using the General Purpose Interface Bus (GPIB). Discussed are interface capabilities, addressing, standard bus commands, and polling schemes. *See also the “Utilities” chapter in the accompanying Operator’s Manual and “Hands-On Guide”.*

### GPIB Structure

GPIB is similar to a standard computer bus. But whereas a computer interconnects circuit cards via a backplane bus, the GPIB interconnects independent devices by means of a cable bus. GPIB also carries both program and interface messages.

**Program messages**, often called device-dependent messages, contain programming instructions, measurement results, instrument status and waveform data. *Their general form is described in the previous chapter.*

**Interface messages** manage the bus itself. They perform functions such as its initialization, the addressing and “unaddressing” of devices, and the setting of remote and local modes.

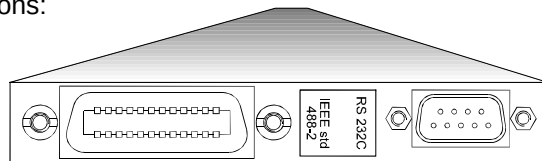
### Talkers and Listeners

Devices connected by GPIB can be listeners, talkers, or controllers. A talker sends program messages to one or more listeners, while the controller manages the flow of information on the bus by sending interface messages to the devices.

The oscilloscope can be a talker or listener, but *not* a controller. The host computer must be able to play all three roles.\*

### Interface Capabilities

The interface capabilities of the oscilloscope include the following IEEE 488.1 definitions:



\* For details of how the controller configures the GPIB for specific functions, refer to the GPIB interface manufacturer's manual.

|            |                                          |
|------------|------------------------------------------|
| <b>AH1</b> | Complete Acceptor Handshake              |
| <b>SH1</b> | Complete Source Handshake                |
| <b>L4</b>  | Partial Listener Function                |
| <b>T5</b>  | Complete Talker Function                 |
| <b>SR1</b> | Complete Service Request Function        |
| <b>RL1</b> | Complete Remote/Local Function           |
| <b>DC1</b> | Complete Device Clear Function           |
| <b>DT1</b> | Complete Device Trigger                  |
| <b>PP1</b> | Parallel Polling: remote configurability |
| <b>C0</b>  | No Controller Functions                  |
| <b>E2</b>  | Tri-state Drivers                        |

## Addressing

Every device on the GPIB has an address. When the remote control port is set to "GPIB", using the oscilloscope's front-panel-operated "UTILITIES" menus, the instrument can be controlled via GPIB. When the remote control port is set to "RS-232" by the same means, the instrument will execute solely "talk-only" operations over the GPIB, such as driving a printer. Setting the oscilloscope to "RS-232" enables the instrument to be controlled via the RS-232-C port (*see next chapter*).

If the oscilloscope is addressed to talk, it will remain thus configured until receiving a universal untalk command (UNT), its own listen address (MLA), or another instrument's talk address.

Similarly, if the oscilloscope is addressed to listen, it will remain configured to listen until a universal *unlisten* command (UNL), or its own talker address (MTA), is received.

## GPIB Signals

The bus system consists of 16 signal lines and eight ground or shield lines. The signal lines are divided into three groups:

1. **Data Lines:** These eight lines, usually called DI01 through to DI08, carry both program and interface messages. Most of the messages use the 7-bit ASCII code, in which case DI08 is unused.

## 9300 & LC Series

---

**2. Handshake Lines:** These three lines control the transfer of message bytes between devices. The process is called a three-wire interlocked handshake and it guarantees that the message bytes on the data lines are sent and received without transmission error.

**3. Interface Management Lines:** These five lines manage the flow of information across the interface:

**ATN (ATteNtion):** The controller drives the ATN line true when it uses the data lines to send interface messages such as talk and listen addresses or a device clear (DCL) message. When ATN is false, the bus is in data mode for the transfer of program messages from talkers to listeners.

**IFC (InterFace Clear):** The controller sets the IFC line true to initialize the bus.

**REN (Remote ENable):** The controller uses this line to place devices in remote or local program mode.

**SRQ (Service ReQuest):** Any device can drive the SRQ line true to asynchronously request service from the controller. This is the equivalent of a single interrupt line on a computer bus.

**EOI (End Or Identify):** This line has two purposes. The talker uses it to mark the end of a message string. The controller uses it to tell devices to identify their response in a parallel poll (discussed later in this section).

### I/O Buffers

The instrument has 256-byte input and output buffers. An incoming program message is not decoded before a message terminator has been received. However, if the input buffer becomes full (because the program message is longer than the buffer), the instrument starts analyzing the message. In this case data transmission is temporarily halted, and the controller may generate a timeout if the limit was set too low.

### IEEE 488.1 Standard Messages

The IEEE 488.1 standard specifies not only the mechanical and electrical aspects of the GPIB, but also the low-level transfer protocol — for instance, it defines how a controller addresses devices, turns them into talkers or listeners, resets them or puts





**Note:** In addition to the IEEE 488.1 interface message standards, the IEEE 488.2 standard specifies certain standardized program messages, i.e. command headers. They are identified with a leading asterisk <\*> and are listed in the System Commands section.

them in the remote state. Such interface messages are executed with the interface management lines of the GPIB, usually with ATN true.

All of these messages (except GET) are executed immediately upon reception and not, with normal commands, in chronological order.

The command list in *System Commands* does not contain a command for clearing the input or output buffers, nor for setting the instrument to the remote state. This is because such commands are already specified as IEEE 488.1 standard messages. Refer to the GPIB interface manual of the host controller as well as to its support programs, which should contain special calls for the execution of these messages.

The following description covers those IEEE 488.1 standard messages which go beyond mere reconfiguration of the bus and have an effect on the operation of the instrument.

#### Device CLeAr

In response to a universal Device CLeAr (DCL) or a Selected Device Clear message (SDC), the oscilloscope clears the input or output buffers, aborts the interpretation of the current command (if any) and clears pending commands. However, status registers and status-enable registers are *not* cleared. Although DCL has an immediate effect it can take several seconds to execute if the instrument is busy.

#### Group Execute Trigger

The Group Execute Trigger message (GET) causes the oscilloscope to arm the trigger system. It is functionally identical to the “\*TRG” command.

#### Remote ENable

This interface message is executed when the controller holds the Remote ENable control line (REN) true and configures the instrument as a listener. All the front-panel controls except the menu buttons are disabled. The menu indications on the right-hand side of the screen no longer appear, since menus cannot

now be operated manually. Instead, the text REMOTE ENABLE appears at the top of the menu field to indicate that the instrument is set to the remote mode. Whenever the controller returns the REN line to false, all instruments on the bus return to GO TO LOCAL. Individual instruments can be returned to LOCAL with the Go To Local message (see *below*).

Local front-panel control may be regained by pressing the GO TO LOCAL menu button, unless the instrument has been placed in Local LOCKout (LLO) mode.

### **Local LOCKout**

The Local LOCKout command (LLO) causes the GO TO LOCAL menu to disappear. The LLO command can be sent in local or remote mode but only becomes effective once the instrument has been set to the remote mode.

### **Go To Local**

The Go To Local message (GTL) causes the instrument to return to local mode. All front-panel controls become active and the normal menus reappear. Thereafter, whenever the instrument is addressed as a listener it will be immediately reset to the remote state — except when the LLO command has been sent.

When Local Lockout is activated the scope can only be returned to its local state by the controller returning the LLO to false. And whenever the instrument returns to the remote state the local lockout mode will immediately become effective again.

### **InterFace Clear**

The InterFace Clear message (IFC) initializes the GPIB but has no effect on the operation of the oscilloscope.

### **Programming GPIB Transfers**

To illustrate the GPIB programming concepts a number of examples written in BASICA are included here. It is assumed that the controller is IBM-PC compatible, running under DOS, and that it is equipped with a National Instruments<sup>†</sup> GPIB interface card. Nevertheless, GPIB programming with other languages such as C or Pascal is quite similar to this.

*If using another type of computer or GPIB interface, refer to the interface manual for installation procedures and subroutine calls.*

---

<sup>†</sup>National Instruments Corporation, 12109 Technology Boulevard, Austin, Texas 78727, USA.



## Configuring the GPIB Driver Hardware

Check that the GPIB interface is properly installed in the computer. If found that it is not, follow the interface manufacturer's installation instructions. In the case of the National Instruments interface, it is possible to modify the base I/O address of the board, the DMA channel number and the interrupt line setting using switches and jumpers.

In these program examples, default positions are assumed.

Connect the oscilloscope to the computer with a GPIB interface cable. Set the GPIB address to the required value. The program examples assume a setting of '4'.

## Configuring the GPIB Driver Software

The host computer requires an interface driver that handles the transactions between the operator's programs and the interface board. In the case of the National Instruments interface, the installation procedure copies the GPIB handler GPIB.COM into the boot directory, modifies the DOS system configuration file CONFIG.SYS to declare the presence of the GPIB handler, creates a sub-directory GPIB-PC, and installs in GPIB-PC a number of files and programs useful for testing and reconfiguring the system and for writing user programs.

The following files in the sub-directory GPIB-PC are particularly useful:

**IBIC.EXE** allows interactive control of the GPIB via functions entered at the keyboard. Use of this program is highly recommended to anyone unfamiliar with GPIB programming or the oscilloscope's remote commands.

**DECL.BAS** is a declaration file that contains code to be included at the beginning of any BASICA application program. Simple application programs can be quickly written by appending the operator's instructions to DECL.BAS and executing the complete file.

**IBCONF.EXE** is an interactive program that allows inspection or modification of the current settings of the GPIB handler. *To run IBCONF.EXE, refer to the National Instruments manual.*

## 9300 &amp; LC Series

In the program examples in this section, it is assumed that the National Instruments GPIB driver GPIB.COM is in its default state — i.e. that the user has not modified it with IBCONF.EXE. This means that the interface board can be referred to by the symbolic name 'GPIB0' and that devices on the GPIB bus with addresses between 1 and 16 can be called by the symbolic names 'DEV1' to 'DEV16'.

**Note:**

*In the program examples in this section, it is assumed that the National Instruments GPIB driver GPIB.COM is in its default state — i.e. that the user has not modified it with IBCONF.EXE. This means that the interface board can be referred to by the symbolic name 'GPIB0' and that devices on the GPIB bus with addresses between 1 and 16 can be called by the symbolic names 'DEV1' to 'DEV16'.*

*If you have a National Instruments PC2 interface card rather than PC2A, you must run IBCONF to declare the presence of this card rather than the default PC2A.*

**Simple Transfers**

For a large number of remote control operations it is sufficient to use just three different subroutines (IBFIND, IBRD and IBWRT) provided by National Instruments. The following complete program reads the timebase setting of the oscilloscope and displays it on the terminal:

```
1-99  <DECL.BAS>
100   DEV$="DEV4"
110   CALL IBFIND(DEV$,SCOPE%)
120   CMD$="TDIV?"
130   CALL IBWRT(SCOPE%,CMD$)
140   CALL IBRD(SCOPE%,RD$)
150   PRINT RD$
160   END
```

Lines 1-99 are a copy of the file DECL.BAS supplied by National Instruments. The first six lines are required for the initialization of the GPIB handler. The other lines are declarations which may be useful for larger programs, but are not really required code. The



sample program above only uses the strings CMD\$ and RD\$ which are declared in DECL.BAS as arrays of 255 characters.

**Note:**

***DECL.BAS requires access to the file BIB.M during the GPIB initialization. BIB.M is one of the files supplied by National Instruments, and it must exist in the directory currently in use.***

***The first two lines of DECL.BAS both contain a string "XXXXX" which must be replaced by the number of bytes which determine the maximum workspace for BASICA (computed by subtracting the size of BIB.M from the space currently available in BASICA). For example, if the size of BIB.M is 1200 bytes and when BASICA is loaded it reports "60200 bytes free", you should replace "XXXXX" by the value 59 000 or less.***

Lines 100 and 110 open the device "DEV4" and associate with it the descriptor "SCOPE%". All I/O calls after that will refer to "SCOPE%". The default configuration of the GPIB handler recognizes "DEV4" and associates with it a device with GPIB address 4.

Lines 120 and 130 prepare the command string TDIV? and transfer it to the instrument. The command instructs the instrument to respond with the current setting of the timebase.

Lines 140 and 150 reads the response of the instrument and places it into the character string RD\$.

Line 170 displays the response on the terminal.

When running this sample program, the oscilloscope will automatically be set to the remote state when IBWRT is executed, and will remain in that state. Pressing the LOCAL menu button will return the oscilloscope to local mode if the GPIB handler was modified to inhibit Local Lockout (LLO).

Here is a slightly modified version of the sample program which checks if any error occurred during GPIB operation:

## 9300 &amp; LC Series

```
1-99 <DECL.BAS>
100 DEV$="DEV4"
110 CALL IBFIND(DEV$,SCOPE%)
120 CMD$="TDIV?"
130 CALL IBWRT(SCOPE%,CMD$)
140 IF ISTA% < 0 THEN GOTO 200
150 CALL IBRD(SCOPE%,RD$)
160 IF ISTA% < 0 THEN GOTO 250
170 PRINT RD$
180 IBLOC(SCOPE%)
190 END
200 PRINT "WRITE ERROR =";IBERR%
210 END
250 PRINT "READ ERROR =";IBERR%
260 END
```

The GPIB status word ISTA%, the GPIB error variable IBERR% and the count variable IBCNT% are defined by the GPIB handler and are updated with every GPIB function call. *Refer to the National Instruments manual for details.* The sample program above would report if the GPIB address of the instrument was set to a value other than 4. Line 180 resets the instrument to local with a call to the GPIB routine IBLOC.

**Additional Driver Calls**

**IBLOC** is used to execute the IEEE 488.1 standard message Go To Local (GTL), i.e. it returns the instrument to the local state. The programming example above illustrates its use.

**IBCLR** executes the IEEE 488.1 standard message Selected Device Clear (SDC).

**IBRDF** and **IBWRTF**, respectively, allow data to be read from GPIB to a file, and written from a file to GPIB. Transferring data directly to or from a storage device does not limit the size of the data block, but may be slower than transferring to the computer memory.

**IBRDI** and **IBWRTI**, respectively, allow data to be read from GPIB to an integer array, and written from integer array to GPIB. Since the integer array allows storage of up to 64 kilobytes (in BASIC), IBRDI and IBWRTI should be used for the transfer of large data blocks to the computer memory, rather than IBRD or



IBWRT, which are limited to 256 bytes by the BASIC string length. Note that IBRDI and IBWRTI only exist for BASIC, since for more modern programming languages, such as C, the function calls IBRD and IBWRT are far less limited in terms of data-block size.

**IBTMO** can be used to change the timeout value during program execution. The default value of the GPIB driver is 10 seconds — for example, if the instrument does not respond to an IBRD call, IBRD will return with an error after the specified time.

**IBTRG** executes the IEEE 488.1 standard message Group Execute Trigger (GET), which causes the oscilloscope to arm the trigger system.

National Instruments supply a number of additional function calls. In particular, it is possible to use the so-called board level calls which allow a very detailed control of the GPIB.

## Service Requests

When an oscilloscope is used in a remote application, events often occur asynchronously — at times that are unpredictable for the host computer. The most common example of this is the wait of a trigger after the arming of the instrument: the controller must wait until the acquisition is finished before it can read the acquired waveform. The simplest way of checking if a certain event has occurred is by either continuously or periodically reading the status bit associated with it until the required transition is detected. *Continuous status bit polling is described in more detail below. For a complete explanation of status bytes refer to Chapter 5.*

A potentially more efficient way of detecting events occurring in the instrument is the use of the Service Request (SRQ). This GPIB interrupt line can be used to interrupt program execution in the controller. The controller can then execute other programs while waiting for the instrument. Unfortunately, not all interface manufacturers support the programming of interrupt service routines. In particular, National Instruments supports only the SRQ bit within the ISTA% status word. This requires the user to continuously or periodically check this word, either explicitly or with the function call IBWAIT. In the absence of real interrupt service routines the use of SRQ may not be very advantageous.

## 9300 &amp; LC Series

In the default state, after power-on, the Service ReQuest is disabled. The SRQ is enabled by setting the Service Request Enable register with the command “\*SRE” and specifying which event should generate an SRQ. The oscilloscope will interrupt the controller as soon as the selected event(s) occur by asserting the SRQ interface line. If several devices are connected to the GPIB, the controller may be required to identify which instrument caused the interrupt by serial polling the various devices.

**Note:** The SRQ bit is latched until the controller reads the SStatus Byte Register (STB). The action of reading the STB with the command “\*STB?” clears the register contents except the MAV bit (bit 4) until a new event occurs. Service requesting may be disabled by clearing the SRE register (“\*SRE 0”).

**Example 1**

To assert SRQ in response to the events “new signal acquired” or “return-to-local” (pressing the soft key/menu button for GO TO LOCAL).

These events are tracked by the INR register which is reflected in the SRE register as the INB summary bit in position 0. Since the bit position 0 has the value 1, the command “\*SRE 1” enables the generation of SRQ whenever the INB summary bit is set.

In addition, the events of the INR register that may be

**Note on Terms:** The term “soft-key”, used here in reference to remote operations, is synonymous with “menu button”, used exclusively in the accompanying Operator’s Manual for front-panel operations. Both terms refer to the column of seven buttons running parallel to the screen on the oscilloscope front panel and the functions they control.

summarized in the INB bit must be specified. The event “new signal acquired” corresponds to INE bit 0 (value 1) while the event “return-to-local” is assigned to INE bit 2 (value 4). The total sum is  $1 + 4 = 5$ . Thus the command “INE 5” is needed:

**CMD\$=“INE 5;\*SRE 1”**





**CALL IBWRT(SCOPE%, CMD\$)**

***Example 2***

**To assert SRQ when soft key 4 (fourth menu button from top of screen) is pressed.**

The event “soft key 4 pressed” is tracked by the URR register. Since the URR register is not directly reflected in STB but only in the ESR register (URR, bit position 6), the ESE enable register must be set first with the command “\*ESE 64” to allow the URQ setting to be reported in STB. An SRQ request will now be generated provided that the ESB summary bit (bit position 5) in the SRE enable register is set (“\*SRE 32”):

**CMD\$=“\*ESE 64; \*SRE 32”**

**CALL IBWRT(SCOPE%, CMD\$)**

## Instrument Polls

State transitions occurring within the instrument can be remotely monitored by polling selected internal status registers (see, too, Chapter 5). Four basic polling methods can be used to detect the occurrence of a given event. These are continuous, serial, parallel and \*IST.

By far the simplest of these is continuous polling. The other three are appropriate only when interrupt service routines (servicing the SRQ line) are supported, or multiple devices on GPIB require constant monitoring. Emphasizing the differences between these methods, which are described below, the same example — determining whether a new acquisition has taken place — is presented in respect of each.

### Continuous Poll

Here, a status register is continuously monitored until a transition is observed. This is the most straightforward method for detecting state changes, but may be impracticable in certain situations, especially with multiple device configurations.

In the following example, the event “new signal acquired” is observed by continuously polling the INternal state change Register (INR) until the corresponding bit (in this case bit 0, i.e. value 1) is non-zero, indicating a new waveform has been acquired. Reading INR clears this at the same time, obviating the need for an additional clearing action after a non-zero value has been detected. The command “CHDR OFF” instructs the instrument to omit any command headers when responding to a query, simplifying the decoding of the response. The instrument will then send “1” instead of “INR 1”:

```
CMD$="CHDR OFF"
CALL IBWRT(SCOPE%,CMD$)
MASK% = 1 'New Signal Bit has value 1'
LOOP% = 1
WHILE LOOP%
  CMD$="INR?"
  CALL IBWRT(SCOPE%,CMD$)
  CALL IBRD(SCOPE%,RD$)
  NEWSIG% = VAL(RD$) AND MASK%
```

```
IF NEWSIG% = MASK% THEN LOOP% = 0  
WEND
```

## **Serial Poll**

Serial polling takes place once the SRQ interrupt line has been asserted, and is only advantageous when there are several instruments involved. The controller finds which device of a number has generated the interrupt by inspecting the SRQ bit in the STB register of each instrument. Because the service request is based on an interrupt mechanism, serial polling offers a reasonable compromise in terms of servicing speed in multiple-device configurations.

In the following example, the command "INE 1" enables the event "new signal acquired" to be reported in the INR to the INB bit of the status byte STB. The command "\*SRE 1" enables the INB of the status byte to generate an SRQ whenever it is set. The function call IBWAIT instructs the computer to wait until one of three conditions occurs: &H8000 in the mask (MASK%) corresponds to a GPIB error, &H4000 to a timeout error, and &H0800 to the detection of RQS (ReQuest for Service) generated by the SRQ bit.

Whenever IBWAIT detects RQS it automatically performs a serial poll to find out which instrument generated the interrupt. It will only exit if there was a timeout or if the instrument "SCOPE%" generated SRQ. The additional function call IBRSP fetches the value of the status byte which may be further interpreted. For this to function properly the value of "Disable Auto Serial Polling" must be set to "off" in the GPIB handler (use IBCONF.EXE to check):

```
CMD$="*CLS; INE 1; *SRE 1"  
CALL IBWRT(SCOPE%,CMD$)  
MASK% = &HC800  
CALL IBWAIT(SCOPE%,MASK%)  
IF (IBSTA% AND &HC000) <> 0 THEN PRINT "GPIB  
or Timeout Error" : STOP  
CALL IBRSP(SCOPE%,SPR%)  
PRINT "Status Byte =.", SPR%
```

Board-level function calls can deal simultaneously with several instruments attached to the same interface board. Refer to the National Instruments manual.

**Note:** After the serial poll is completed, the RQS bit in the STB status register is cleared. Note that the other STB register bits remain set until they are cleared by means of a “\*CLS” command or the instrument is reset. If these bits are not cleared, they cannot generate another interrupt.

## Parallel Poll

Like serial polling, this is only advantageous when there are several instruments. In parallel polling, the controller simultaneously reads the Individual Status bit (IST) of all the instruments to determine which needs service. Because this method allows up to eight different instruments to be polled at the same time, it is the fastest way to identify state changes in instruments with this capability.

When a parallel poll is initiated, each instrument returns a status bit over one of the DIO data lines. Devices may respond either individually, using a separate DIO line, or collectively on a single data line. Data-line assignments are made by the controller using a Parallel Poll Configure (PPC) sequence.

In the following example, the command “INE 1” enables the event “new signal acquired” in the INR to be reported to the INB bit of the status byte STB. The PaRallel poll Enable register (PRE) determines which events will be summarized in the IST status bit. The command “\*PRE 1” enables the INB bit to set the IST bit whenever it is itself set. Once parallel polling has been established, the parallel-poll status is examined until a change on data bus line DI02 takes place.

**Stage 1:** Enable the INE and PRE registers, configure the controller for parallel poll and instruct the oscilloscope to respond on data line 2 (DI02)

```
CMD1$="?_@$"  
CALL IBCMD(BRD0%, CMD1$)  
CMD$="INE 1; *PRE 1"  
CALL IBWRT(BRD0%, CMD$)  
CMD4$=CHR$(&H5)+CHR$(&H69)+"?"  
CALL IBCMD(BRD0%, CMD4$)
```

**Stage 2:** Parallel poll the instrument until DI02 is set

```

LOOP% = 1
WHILE LOOP%
CALL IBRPP (BRD0%, PPR%)
IF (PPR% AND &H2) = 2 THEN LOOP% = 0
WEND

```

**Stage 3:** Disable parallel polling (hex 15) and clear the parallel poll register

```

CMD5$=CHR$(&H15)
CALL IBCMD (BRD0%, CMD5$)
CALL IBCMD (BRD0%, CMD1$)
CMD$="*PRE 0"
CALL IBWRT (BRD0%, CMD$) :

```

**Note:** In the example above, board-level GPIB function calls are used. It is assumed that the controller (board) and scope (device) are respectively located at addresses 0 and 4. The listener and talker addresses for the controller and oscilloscope are

| Logic Device        | Listener Address   | Talker Address    |
|---------------------|--------------------|-------------------|
| External Controller | 32 ASCII<space>    | 64 (ASCII @)      |
| Oscilloscope        | 32+4=36 (ASCII \$) | 64+4=68 (ASCII D) |

**\*IST Poll**

The state of the Individual Status bit (IST) returned in parallel polling can also be read by sending the “\*IST?” query. To enable this poll mode, the oscilloscope must be initialized as for parallel polling by writing into the PRE register. Since \*IST polling emulates parallel polling, this method is applicable in all instances where parallel polling is not supported by the controller.

**Note:**

*The characters “?” and “\_” appearing in the command strings stand for unlisten and untalk respectively. They are used to set the devices to a “known” state.*

*To shorten the size of the program examples, device talking and listening initialization instructions have been grouped into character chains. They are:*

*CMD1\$ = “?\_@\$” Unlisten, Untalk, PC talker, DSO listener*

*The remote message code for executing a parallel response in binary form is 01101PPP, where PPP specifies the data line. Since data line 2 is selected, the identification code is 001, which results in the code 01101001 (binary) or &H69 (hex). See Table 38 of the IEEE 488-1978 Standard for further details.*

In the following example, the command “INE 1” enables the event “new signal acquired” in the INR to be reported to the INB bit of the status byte STB. The command “\*PRE 1” enables the INB bit to set the IST bit whenever it is set. The command “CHDR OFF” suppresses the command header in the instrument’s response, simplifying the interpretation. The status of the IST bit is then continuously monitored until set by the instrument:



```
CMD$="CHDR OFF; INE 1; *PRE 1"  
CALL IBWRT(SCOPE%, CMD$)  
LOOP% = 1  
WHILE LOOP%  
    CMD$="*IST?"  
    CALL IBWRT(SCOPE%, CMD$)  
    CALL IBRD(SCOPE%, RD$)  
    IF VAL(RD$) = 1 THEN LOOP% = 0  
WEND
```

## Driving Hardcopy Devices on the GPIB

The oscilloscope can be interfaced with a wide range of hardcopy devices for copying the screen contents to them. The devices supported are listed with the command "HARDCOPY\_SETUP" (see *System Commands*).

With a hardcopy device connected to the GPIB, one of two basic configurations can be used. When only the oscilloscope and a hardcopy device such as a printer are connected to the GPIB, the oscilloscope must be configured as talker-only, and the hardcopy device as listener-only, to ensure proper data transfer (see *below*).

However, when an external controller is connected to the GPIB, it must be used to supervise the data transfers. A variety of schemes can then be used to transfer the oscilloscope screen contents (see *next page*).

### DSO & Printer Only

The oscilloscope is configured as talker-only using the "GPIB & RS232" menus selected using the instrument's front-panel controls (see the sections "GPIB/RS232 Setup" and "Hardcopy Setup" in the *UTILITIES Chapter of the Operator's Manual*). The hardcopy device manufacturer usually specifies an address that forces the instrument into listening mode, and this as well as the other necessary settings can be selected using the same menus.

**To configure the oscilloscope as talker-only:**

1. Press  to select "Remote Control from RS-232" from the ("UTILITIES" menu).
2. Select "GPIB" from the "output to" menu in the "HARDCOPY" group.
3. Put the hardcopy device in listener-only mode.

4. Press    
 SCREEN  
 DUMP



## Printing by GPIB Controller

The following schemes can be used for driving hardcopy devices by remote control using the GPIB.

### Data read by controller

The controller reads the data into internal memory, then sends them to the printer. This alternative can be done with simple high-level GPIB function calls. The controller stores the full set of printer instructions and afterwards sends them to the graphics device. This method is the most straightforward way to transfer screen contents, but requires a large amount of buffer storage:

```
CMD$ = "SCDP"
CALL IBWRT(SCOPE%, CMD$)
FILE$="PRINT.DAT"
CALL IBRDF(SCOPE%, FILE$)
CALL IBWRTF(PRINTER%, FILE$)
```

### DSO sends data to both

The oscilloscope sends data to both controller and printer. The oscilloscope puts the printer instructions onto the bus. The data is directly put out and saved in scratch memory in the controller. The contents of the scratch file can later be deleted.

**Stage 1:** Controller talker, oscilloscope listener. Issue the screen dump command

```
CMD1$="?" @$: CALL IBCMD(BRD0%, CMD1$)
CMD$="SCDP": CALL IBWRT(BRD0%, CMD$)
```

**Stage 2:** Oscilloscope talker, controller and printer listeners. Print data while storing data in scratch file SCRATCH.DAT

```
CMD2$="?" D%: CALL IBCMD(BRD0%, CMD2$)
FILE$="SCRATCH.DAT": CALL IBRDF(BRD0%, FILE$)
```

### DSO talks directly to printer

The controller goes into a standby state. The oscilloscope becomes a talker and sends data directly to the printer. The controller goes into stand-by and resumes GPIB operations once the data have been printed, i.e. when an EOI is detected:

## 9300 &amp; LC Series

**Stage 1:** Controller talker, oscilloscope listener. Issue the screen dump command

```
CMD1$="?_@$": CALL IBCMD(BRD0%,CMD1$)
CMD$="SCDP": CALL IBWRT(BRD0%,CMD$)
```

**Stage 2:** Oscilloscope talker, printer listener. Put controller in stand-by

```
CMD2$="?_D%": CALL IBCMD(BRD0%,CMD2$)
V%=1: CALL IBGTS(BRD0%,V%)
```

**Note:**

*In Schemes 2 and 3, board-level GPIB function calls are used. It is assumed that the controller (board), the scope and the printer are respectively located at addresses 0, 4 and 5. The listener and talker addresses for the controller, oscilloscope and printer are*

| Logic Device | Listener Address   | Talker Address    |
|--------------|--------------------|-------------------|
| Controller   | 32 (ASCII<space>)  | 64 (ASCII @)      |
| Oscilloscope | 32+4=36 (ASCII \$) | 64+4=68 (ASCII D) |
| Printer      | 32+5=37 (ASCII %)  | 64+5=69 (ASCII E) |

*The characters “?” and “\_” appearing in the command strings stand for unlisten and untalk respectively. They are used to set the devices to a “known” state.*

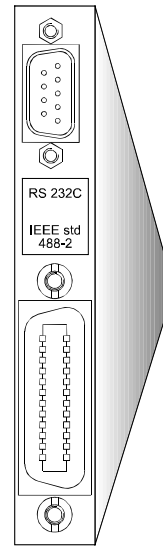
*To shorten the size of the program examples, device talking and listening initialization instructions have been grouped into character chains. They are:*

```
CMD1$ = "?_@$" Unlisten, Untalk, PC talker, DSO
            listener
CMD2$ = "?_D%" Unlisten, Untalk, PC listener, DSO
            talker
```

## Using the RS-232-C Port

This chapter describes remote control of the oscilloscope when the RS-232-C port is used. It concludes with definitions of the special RS-232-C commands used for configuring the oscilloscope, or for simulating GPIB 488.1 messages such as those for placing the oscilloscope in either remote or local mode.

The port supports all commands described in the *System Commands* section of this manual. However, waveform transfer is only possible in HEX mode, while the default value for COMM\_FORMAT is set appropriately. The syntax of the response to WF? is identical to that for GPIB.



### Notation

In this chapter, characters that cannot be printed in ASCII are represented by their mnemonics, for example:

- <LF> ASCII line feed character whose decimal value is 10.
- <BS> ASCII backspace character whose decimal value is 8.
- CTRL\_U means that the control key and the U key are pressed simultaneously.

**Connector Pin Assignments** See the section "GPIB/RS232 Setup" in the UTILITIES Chapter of the oscilloscope Operator's Manual.

**RS-232-C Configuration** The RS-232-C port is full-duplex configured. This means that both sides, the external controller *and* the oscilloscope, can send and receive messages at the same time.

Nevertheless, when the oscilloscope receives a new command, it stops outputting.

Transmission of long messages to the oscilloscope should be done while it is in a triggered mode, and *not* while an acquisition is in progress. This is especially important when sending waveforms or front-panel setups into the oscilloscope.

The behavior of the RS-232-C port may be set according to the operator's needs. In addition to the basic setup on the front-panel



menu, there are “immediate commands”, as well as a special command “COMM\_RS232”. Immediate commands consist of the ASCII ESCape character <ESC> (whose decimal value is 27), followed by another character. These commands are interpreted as soon as the second character has been received.

***Note: The RS-232-C baud rate, parity, character length and number of stop bits are among the parameters that are saved or recalled by the front-panel “SAVE” or “RECALL” buttons, or by the remote commands “\*SAV”, “\*RCL” or “PANEL\_SETUP”. When recalling, care must be taken to ensure that these parameters are set at the same value as the actual ones. Otherwise, the host may no longer be able to communicate with the oscilloscope and a manual reconfiguration would be necessary.***

## Echo

The serial port may echo the received characters. Echo is useful if the oscilloscope is attached to a terminal. Echoing can be turned on or off by sending the two-character sequence <ESC>[ or <ESC>[ respectively. Echoing is on by default.

## Handshake Control

When the oscilloscope intake buffer becomes nearly full, the instrument sends a handshake signal to the host telling it to stop transmitting. When this buffer has enough room to receive more characters, another handshake signal is sent. These handshake signals are either the CTRL-S (or <XOFF>) and CTRL-Q (<XON>) characters, or a signal level on the RTS line. They are selected by sending the two-character sequence <ESC>[ for XON/XOFF handshake — the default — or <ESC>[ for RTS handshake.

The flow of characters coming from the oscilloscope may be controlled either by a signal level on the CTS line or by the <XON>/<XOFF> pair of characters.

***Note: The host must NOT echo characters received from the oscilloscope.***

**9300 & LC Series**

---

**Editing Features**

When the oscilloscope is directly connected to a terminal, the following will facilitate the correction of typing errors:

|                  |                            |
|------------------|----------------------------|
| <BS> or <DELETE> | Delete the last character. |
| CTRL_U           | Delete the last line.      |

**Message Terminators**

"Message terminators" are markers that indicate to the receiver that a message has been completed.

On input to the oscilloscope, the Program Message Terminator is a character that could be selected. A good choice is a character never used for anything else. Such a character is chosen using the command COMM\_RS232 and the keyword EI. The default Program Message Terminator is the ASCII character <CR>, whose decimal value is 13.

The oscilloscope appends a Response Message Terminator to the end of each of its responses. This is a string, similar to a computer prompt, which is chosen by the user. This string must not be empty. The default Response Message Terminator is "\n\r", which means <LF><CR>.

**Examples****COMM\_RS232 EI, 3**

This command informs the oscilloscope that each message it receives will be terminated with the ASCII character <ETX> which corresponds to 3 in decimal.

**COMM\_RS232 E0, "\r\nEND\r\n"**

This command indicates to the oscilloscope that it must append the string "\r\nEND\r\n" to each response.

After these settings, a host command will look like:

**TDIV?<ETX>**

The oscilloscope responds:

**TDIV 1.S  
END**



**Note:** Having sent a `COMM_RS232` command, the host must wait for the oscilloscope to change its behavior before sending a command in the new mode. A sure way to do this is to include a query on the line that contains the `COMM_RS232` command and wait until the response is received. For example: `COMM_RS232 EI, 3; *STB?`

## SRQ Message

Each time the Master Summary Status (MSS) bit of the Status Byte (STB) is set, the SRQ message (a string of characters) is sent to the host to indicate that the oscilloscope requests service. The RS-232-C SRQ message has the same meaning as the GPIB SRQ message. If the string is empty, no message will be sent. This is the default setting. Note that no response message terminator is added at the end of the SRQ message.

### Example

**COMM\_RS232 SRQ, "\r\n\nSRQ\r\na"**

When the MSS bit is set, the oscilloscope will send:

a <CR> followed by two <LF>

SRQ

a <CR> followed by one <LF>

and the buzzer will sound.

## Long Line Splitting

Line splitting is a feature provided for hosts that cannot accept lines with more than a certain number of characters. The oscilloscope may be configured to split responses into many lines. This feature is very useful for waveform or front-panel setup transfers although it is applicable to all response messages. Two parameters control this feature:

### Line Separator: Off

— messages will not be split into lines.

<CR>, <LF> or <CR><LF>

— possible line terminators.

**Line Length:** the maximum number of characters to a line.

### Example

**COMM\_RS232 LS, LF, LL, 40**

**9300 & LC Series**

---

The line separator is the ASCII character <LF>, the line is a maximum of 40 characters long (excluding the line separator).

If the oscilloscope receives the command PNSU?, it may answer:

```
PNSU#9000001496  
AAAA5555000655AA4030005800190000000000001  
00000000000000000000000000000000C1B0100580000  
0000000000000000000000000000000000000000000000000  
. . .
```

**Remarks**

Long commands sent to the oscilloscope may not be split into lines. If a command sent to the oscilloscope is the response to a previous query, the line-split characters (<LF> and/or <CR>) must be removed.

This also applies to line-split characters inside strings sent to the oscilloscope.

However, hex-ASCII data sent to the oscilloscope may contain line-split characters. If you wish to use line splitting, ensure that neither the input message terminator characters nor the line-split characters occur in the data.



## Simulating GPIB Commands with RS-232-C

These special RS-232-C commands can be used for simulating GPIB 488.1 messages.

**<ESC>C or <ESC>c**  
Device Clear Command

**Clears the input and output buffers.** This command has the same meaning as the GPIB DCL or SDC interface messages.

**<ESC>R or <ESC>r**  
Set to Remote Command  
(REN)

**Places the oscilloscope into the remote mode.** This command's function is the same as the GPIB command asserting the REN line and setting the oscilloscope to listener.

**<ESC>L or <ESC>l**  
Set to Local Command

**Places the oscilloscope into local mode.** The command clears local lockout (*see below*). It has the same function as GPIB setting the REN line to false.

**<ESC>F or <ESC>f**  
Set to Local Lockout  
Command

**Disables the front-panel "LOCAL" button**, either immediately, if the oscilloscope is already in remote mode, or later, when the oscilloscope is next set to remote.

This disabling of the menu "LOCAL" button is called "Local Lockout" and can only be cancelled with the <ESC>L command. <ESC>F has the same meaning as the GPIB LLO interface message.

**<ESC>T or <ESC>t**  
Trigger Command (GET)

**Rearms the oscilloscope while it is in the "STOP" mode**, but only while the oscilloscope is in remote mode. This command has the same meaning as the "\*TRG" command, as well as having the same meaning as the GPIB GET interface message.



## Understanding Waveforms

**This chapter covers the reading and writing of waveforms in remote control, and attempts to explain their structure and content.**

### Basic Structure

Waveforms can be divided into two basic entities. One is the basic data array: the raw data values from the oscilloscope's ADCs (Analog-to-Digital Converters) in the acquisition. The other is the accompanying descriptive information, such as vertical and horizontal scale, and time of day, necessary for a full understanding of the information contained in the waveform.

This information can be accessed by remote control using the `INSPECT?` Query, which interprets it in an easily understood ASCII text form. It can be more rapidly transferred using the `WAVEFORM?` query, or written back into the instrument with the `WAVEFORM` command.

The oscilloscope itself contains a data structure, or *template*, which provides a detailed description of how the waveform's information is organized.

Waveforms can also be stored in pre-formatted ASCII output\*, for popular spreadsheets and math processing packages, using the `STORE` and `STORE_SETUP` commands.

### Waveform Template

This gives a detailed description of the form and content of the logical data blocks of a waveform, and is provided as a reference. Although a sample template is given elsewhere in this manual (see *Appendix B*), it is suggested that the `TEMPLATE?` query and the actual instrument template be used. The template may change as the instrument's firmware is enhanced, and it will help provide backward compatibility for the interpretation of waveforms.

### Logical Data Blocks

A waveform normally contains a waveform descriptor block and a data array block. However, in more complicated cases, one or more other blocks will be present.

---

\* See *Appendix E* of the Operator's Manual for these formats.



## Waveform Structure

---

Waveform Descriptor block (WAVEDESC): This block includes all the information necessary to reconstitute the display of the waveform from the data. This includes:

- hardware settings at the time of acquisition
- the exact time of the event
- the kinds of processing that have been performed
- the name and serial number of the instrument
- the encoding format used for the data blocks
- miscellaneous constants.

Optional User-provided Text block (USERTEXT): The WFTX command can be used to put a title or description of a waveform into this block. The WFTX? query command gives an alternative way to read it. This text block can hold up to 160 characters. They can be displayed in the TEXT + TIMES status menu as four lines of 40 characters.

Sequence Acquisition Times block (TRIGTIME): This block is needed for sequence acquisitions to record the exact timing information for each segment. It contains the time of each trigger relative to the trigger of the first segment, as well as the time of the first data point of each segment relative to its trigger.

Random interleaved sampling times block (RISTIME): This block is needed for RIS acquisitions to record the exact timing information for each segment.

First data array block (SIMPLE or DATA\_ARRAY\_1): This is the basic integer data of the waveform. It can be raw or corrected ADC data or the integer result of waveform processing.

Second data array block (DATA\_ARRAY\_2): This second data array is needed to hold the results of processing functions such as the Extrema (WP01 option) or Complex FFT (WP02 option). In such cases, the data arrays contain:

## 9300 &amp; LC Series

|              | Extrema     | FFT            |
|--------------|-------------|----------------|
| DATA_ARRAY_1 | Roof trace  | Real part      |
| DATA_ARRAY_2 | Floor trace | Imaginary part |

***Note: The Template also describes an array named DUAL. This is simply a way to allow the INSPECT? command to examine the two data arrays together.***



## Using the INSPECT? Query

The query **INSPECT?** is a simple way to examine the contents of a waveform in remote control.

Usable on both the data and descriptive parts, its most basic form is:

**INSPECT? "name"**

where the template gives the name of a descriptor item or data block. The answer is returned as a single string, but may span many lines. Some typical dialogue:

*question*  
*response*  
*question*  
*response*

**C1:INSPECT? "VERTICAL\_OFFSET"**

**C1:INSP "VERTICAL\_OFFSET: 1.5625e-03"**

**C1:INSPECT? "TRIGGER\_TIME"**

**C1:INSP "TRIGGER\_TIME: Date = FEB 17, 1994,  
Time = 4: 4:29.5580"**

**INSPECT?** can also be used to provide a readable translation of the full waveform descriptor block with:

**INSPECT? "WAVEDESC"**

The template dump will give details of the interpretation of each of the parameters. **INSPECT?** is also used to examine the measured data values of a waveform using:

**INSPECT? "SIMPLE"**

For example, for an acquisition with 52 points:

```
INSPECT? "SIMPLE"
C1:INSP "
0.0005225 0.0006475 -0.00029 -0.000915 2.25001E-0 0.000835
0.0001475 -0.0013525 -0.00204 -4E-05 0.0011475 0.0011475
-0.000915 -0.00179 -0.0002275 0.0011475 0.001085 -0.00079
-0.00179 -0.0002275 0.00071 0.00096 -0.0003525 -0.00104
0.0002725 0.0007725 0.00071 -0.0003525 -0.00129 -0.0002275
0.0005225 0.00046 -0.00104 -0.00154 0.0005225 0.0012725
0.001335 -0.0009775 -0.001915 -0.000165 0.0012725 0.00096
-0.000665 -0.001665 -0.0001025 0.0010225 0.00096 -0.0003525
-0.000915 8.50001E-0 0.000835 0.0005225
"
```

---

9300 & LC Series

---

These numbers are the fully converted measurements in volts. Of course, when the data block contains thousands of items the string will contain a great many lines.

Depending on the application, the data may be preferred in its raw form as either a BYTE (8 bits) or a WORD (16 bits) for each data value. In this case the relations given below must be used in association with WAVEFORM? to interpret the measurement. It might then say:

**INSPECT? "SIMPLE", BYTE**

The examination of data values for waveforms with two data arrays can be performed as follows:

|                                |                                              |
|--------------------------------|----------------------------------------------|
| <b>INSPECT? "DUAL"</b>         | to get pairs of data values on a single line |
| <b>INSPECT? "DATA_ARRAY_1"</b> | to get the values of the first data array    |
| <b>INSPECT? "DATA_ARRAY_2"</b> | to get the values of the second data array.  |

**Finally...**

INSPECT? is useful, but it is also a rather verbose way to send information. As a query form only, INSPECT? cannot be used to send a waveform back into the oscilloscope. Users who require this capability or speed or both should instead use the WAVEFORM query or commands. It is possible to examine just a part of the waveform or a sparsed form of it, using the WAVEFORM\_SETUP command covered later in this chapter.

BASIC users might find it convenient, too, to combine the capabilities of the inspect facility with the waveform query command in order to construct files containing a human and BASIC readable version of the waveform descriptor together with the full waveform in a format suitable for retransmission to the instrument. This can be done for a waveform in a memory location by sending the command

**MC:INSPECT? "WAVEDESC";WAVEFORM?**

and putting the response directly into a disk file.



## WAVEFORM?, Related Commands, and Blocks

Using the **WAVEFORM?** query is an effective way to transfer waveform data using the block formats defined in the IEEE-488.2 standard. Responses can then be downloaded back into the instrument using the **WAVEFORM** command.

All of a waveform's logical blocks can be read with the single query:

### **C1:WAVEFORM?**

This is the preferred form for most applications due to its completeness. Time and space are the advantages when reading many waveforms with the same acquisition conditions, or when the interest is only in large amounts of raw integer data.

And any single block can be chosen for reading with a query such as:

### **C1:WAVEFORM? DAT1**

The description in the *System Commands* section provides the various block names.

**Note:** A waveform query response can easily be a block containing over 16 million bytes if it is in binary format and twice as much if the HEX option is used.

### Interpreting the Waveform Descriptor

The binary response to a query of the form:

### **C1:WAVEFORM? or C1:WAVEFORM? ALL**

can be placed in a disk file and then dumped to show the following hexadecimal and ASCII form<sup>†</sup>:

---

<sup>†</sup> Done using the GPIB bus with default settings.

## 9300 &amp; LC Series

| Byte Offset #    | Binary Contents in Hexadecimal |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | ASCII Translation<br>(.= uninteresting) |
|------------------|--------------------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----------------------------------------|
| 0                | 43                             | 31 | 3A | 57 | 46 | 20 | 41 | 4C | 4C | 2C | 23 | 39 | 30 | 30 | 30 | 30 | C1:WFALL, #90000                        |
| 16               | 30                             | 30 | 34 | 35 | 30 |    |    |    |    |    |    |    |    |    |    |    | 00450                                   |
| 0                |                                |    |    |    |    | 57 | 41 | 56 | 45 | 44 | 45 | 53 | 43 | 00 | 00 | 00 | WAVEDESC...                             |
| 32               | 11                             | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 01 | 5A | 00 | 00 | .....LECROY 2 2.                        |
| 48               | 27                             | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | .....                                   |
| 64               | 43                             | 00 | 00 | 00 | 00 | 68 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | .....                                   |
| 80               | 59                             | 00 | 4C | 45 | 43 | 52 | 4F | 59 | 39 | 33 | 37 | 34 | 4C | 00 | 00 | 00 | .....LECROY9374L.....                   |
| 96               | 75                             | 00 | 37 | 84 | 09 | 40 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |                                         |
| 112              | 91                             | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 34 | 00 | 00 | 00 | 34 | 00 | 00 |                                         |
| 128              | 107                            | 32 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 33 | 00 | 00 | 00 | 00 | 00 | 00 |                                         |
| 144              | 123                            | 01 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 01 | 00 | 00 | 00 | 01 | 00 | 00 |                                         |
| 160              | 139                            | 00 | 34 | 83 | 12 | 6F | 3A | 0D | 8E | C9 | 46 | FE | 00 | 00 | C7 | 00 |                                         |
| 176              | 155                            | 00 | 00 | 08 | 00 | 01 | 32 | 2B | CC | 77 | BE | 6B | A4 | BB | 51 | A0 | 69                                      |
| 192              | 171                            | BB | BE | 6A | D7 | F2 | A0 | 00 | 00 | 00 | 56 | 00 | 00 | 00 | 00 | 00 | 00                                      |
| 208              | 187                            | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00                                      |
| 224              | 203                            | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00                                      |
| 240              | 219                            | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00                                      |
| 256              | 235                            | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 40 | 3B | 00                                      |
| 272              | 251                            | 00 | 00 | 00 | 00 | 00 | 17 | 0A | 05 | 02 | 07 | C8 | 00 | 00 | 00 | 00 | 00                                      |
| 288              | 267                            | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 01 | 00 | 0E | 00 | 04 | 3F | 80 | 00 | 00                                      |
| 304              | 283                            | 00 | 00 | 0A | 00 | 00 | 3F | 80 | 00 | 00 | 3A | 0D | 8E | C9 | 00 | 00 | 00                                      |
| 320              | 299                            |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 11                                      |
| 336              | 315                            | 00 | 13 | 00 | 04 | 00 | FA | 00 | 09 | 00 | 16 | 00 | 0B | 00 | F3 | 00 | E8                                      |
| 352              | 331                            | 00 | 08 | 00 | 1B | 00 | 1B | 00 | FA | 00 | EC | 00 | 05 | 00 | 1B | 00 | 1A                                      |
|                  |                                | 00 | FC | 00 | EC | 00 | 05 | 00 | 14 | 00 | 18 | 00 | 03 | 00 | F8 | 00 | 0D                                      |
| 367              | 0                              | 00 | 15 | 00 | 14 | 00 | 03 | 00 | F4 | 00 | 05 | 00 | 11 | 00 | 10 | 00 | F8                                      |
| 368              | 1                              | 00 | F0 | 00 | 11 | 00 | 1D | 00 | 1E | 00 | F9 | 00 | EA | 00 | 06 | 00 | 1D                                      |
| 384              | 17                             | 00 | 18 | 00 | FE | 00 | EE | 00 | 07 | 00 | 19 | 00 | 18 | 00 | 03 | 00 | FA                                      |
| 400              | 33                             | 00 | 0A | 00 | 16 | 00 | 11 | 00 |    |    |    |    |    |    |    |    |                                         |
| 416              | 49                             |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |                                         |
| 432              | 65                             |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |                                         |
| 448              | 81                             |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |                                         |
| 464              | 97                             |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |                                         |
| 471 (Terminator) |                                |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0A                                      |

Here, in order to illustrate the contents of the logical blocks, the relevant parts (see explanations next page) have been separated. In addition, to facilitate counting, the corresponding Byte Offset numbering has been restarted each time a new block begins. The ASCII translation, only part of which is shown, has been similarly split and highlighted, showing how its parts correspond to the binary contents, highlighted in the same fashion.



## Waveform Structure

**On the preceding page...** The first 10 bytes translate into ASCII and resemble the simple beginning of a query response. These are followed by the string “#9000000450”, the beginning of a binary block in which nine ASCII integers are used to give the length of the block (450 bytes). The waveform itself starts immediately after this, at Byte number 21. The very first byte is Byte #0, as it is for the first byte in each block (at the head of each of the three Byte Offset columns illustrated).

The first object<sup>‡</sup> is a DESCRIPTOR\_NAME, a string of 16 characters with the value **WAVEDESC**.

Then, 16 bytes after the beginning of the descriptor (or at Byte #37, counting from the very start and referring to the numbers in the first Byte Offset column), we find the beginning of the next string: the TEMPLATE\_NAME with the value **LECROY\_2\_2**.

Several other parameters follow. The INSTRUMENT\_NAME, 76 bytes from the descriptor start (Byte #97), is easily recognizable. On the preceding line, at 38 bytes after the descriptor (Byte #59), a four-byte-long integer gives the length of the descriptor:

WAVE\_DESCRIPTOR = 00 00 01 5A (hex) = 346.

At 60 bytes from the descriptor start (or Byte #81) we find another four-byte integer giving the length of the data array:

WAVE\_ARRAY\_1 = 00 00 00 68 (hex) = 104.

And at 116 bytes after the descriptor (Byte #137), yet another four-byte integer gives the number of data points:

WAVE\_ARRAY\_COUNT = 00 0000 34 (hex) = 52.

Now we know that the data will start at 346 bytes from the descriptor's beginning (Byte #367), and that each of the 52 data points will be represented by two bytes. The waveform has a total length of 346 + 104, which is the same as the ASCII string indicated at the beginning of the block. The final 0A at Byte #471 is the NL character associated with the GPIB message terminator <NL><EOI>.

---

<sup>‡</sup> The waveform descriptor can be deciphered with the aid of the waveform template (see Appendix B).



As the example was taken using an oscilloscope with an eight-bit ADC, we see the eight bits followed by a 0 byte for each data point. However, for many other kinds of waveform this second byte will not be zero and will contain significant information. The data is coded in signed form (two's complement) with values ranging from  $-32768 = 8000$  (hex) to  $32767 = 7FFF$  (hex). If we had chosen to use the BYTE option for the data format the values would have been signed integers in the range  $-128 = 80$  (hex) to  $127 = 7F$  (hex). These ADC values are mapped to the display grid in the following way:

0 is located on the grid's center axis

127 (BYTE format) or 32767 (WORD format) is located at the top of the grid

$-128$  (BYTE format) or  $-32768$  (WORD format) is located at the bottom of the grid.

### Interpreting Vertical Data

Now that we know how to decipher the data it would be useful to convert it to the appropriate measured values.

The vertical reading for each data point depends on the vertical gain and the vertical offset given in the descriptor. For acquisition waveforms this corresponds to the volts/div and voltage offset selected after conversion for the data representation being used. The template tells us that the vertical gain and offset can be found at bytes 156 and 160 respectively of the descriptor start and that they are stored as floating point numbers in the IEEE 32-bit format. An ASCII string giving the vertical unit is to be found in VERTUNIT, Byte #196. The vertical value is given by the relationship:

$$\text{value} = \text{VERTICAL\_GAIN} \times \text{data} - \text{VERTICAL\_OFFSET}$$

In the case of the data shown above we find:

VERTICAL\_GAIN = 2.44141e-07 from the floating point number 3483 126f at byte 177

VERTICAL\_OFFSET = 0.00054 from the floating point number 3A0D 8EC9 at byte 181

VERTICAL\_UNIT = V = volts from the string 5600 ... at byte 217

and therefore:



## Waveform Structure

since  $\text{data}[4] = \text{FA00} = 64000$  from the hexadecimal word FA00 at byte 371. Overflows the maximum. 16 bit value of 32767, so must be a negative value. Using the two's complement conversion  $64000 - 2^{16} = -1536$

$\text{value}[4] = -0.000915 \text{ V}$  as stated in the inspect command.

If the computer or the software available is not able to understand the IEEE floating point values, a description is to be found in the template.

The data values in a waveform may not all correspond to measured points. FIRST\_VALID\_PNT and LAST\_VALID\_PNT give the necessary information. The descriptor also records the SPARSING\_FACTOR, the FIRST\_POINT, and the SEGMENT\_INDEX to aid interpretation if the options of the WAVEFORM\_SETUP command have been used.

For sequence acquisitions the data values for each segment are given in their normal order and the segments are read out one after the other. The important descriptor parameters are the WAVE\_ARRAY\_COUNT and the SUBARRAY\_COUNT, giving the total number of points and the number of segments.

For waveforms such as the extrema and the complex FFT there will be two arrays — one after the other — for the two of the result.

### Calculating a Data Point's Horizontal Position

Each vertical data value has a corresponding horizontal position, usually measured in time or frequency units. The calculation of this position depends on the type of waveform being examined. We will treat separately the single sweep, the sequence, and the interleaved (RIS) waveform. Each data value has a position,  $i$ , in the original waveform, with  $i = 0$  corresponding to the first data point acquired. The descriptor parameter HORUNIT gives a string with the name of the horizontal unit.

**Single-Sweep Waveforms**

$$x[i] = \text{HORIZ\_INTERVAL} \times i + \text{HORIZ\_OFFSET}$$

For acquisition waveforms this time is from the trigger to the data point in question. It will be different from acquisition to acquisition since the HORIZ\_OFFSET is measured for each trigger.

In the case of the data shown above this means:

HORIZ\_INTERVAL = 1e-08 from the floating point number 322b cc77 at byte 194

HORIZ\_OFFSET = -5.149e-08 from the double precision floating point number be6b a4bb 51a0 69bb at byte 198

HORUNIT = S = seconds from the string 5300 ... at byte 262

which gives:

$$x[0] = -5.149e-08 \text{ S}$$

$$x[1] = -4.149e-08 \text{ S.}$$

**Sequence Waveforms**

Since sequence waveforms are really many independent acquisitions, each segment will have its own horizontal offset. These can be found in the TRIGTIME array. For the n'th segment

$$x[i,n] = \text{HORIZ\_INTERVAL} \times i + \text{TRIGGER\_OFFSET}[n].$$

The TRIGTIME array can contain up to 200 segments of timing information with two eight-byte double precision floating point numbers for each segment.

**Interleaved (RIS) Waveforms**

These are composed of many acquisitions interleaved together. The descriptor parameter, RIS\_SWEEPS gives the number of acquisitions. The i'th point will belong to the m'th segment where:

$$m = i \text{ modulo } (\text{RIS\_SWEEPS})$$

will have a value between 0 and RIS\_SWEEPS -1.

Then with:

$$j = i - m$$

$$x[i] = x[j,m] = \text{HORIZ\_INTERVAL} \times j + \text{RIS\_OFFSET}[m],$$

where the RIS\_OFFSETs can be found in the RISTIME array. There can be up to 100 eight-byte double precision floating point numbers in this block. The instrument tries to get segments with times such that:



## Waveform Structure

$$\text{RIS\_OFFSET}[i] \cong \text{PIXEL\_OFFSET} + (i - 0.5) \times \text{HORIZ\_INTERVAL}$$

Thus, taking as an example a RIS with  $\text{RIS\_SWEEPS} = 10$ ,  $\text{HORIZ\_INTERVAL} = 1 \text{ ns}$ , and  $\text{PIXEL\_OFFSET} = 0.0$ , we might find for a particular event that:

$$\begin{aligned} \text{RIS\_OFFSET}[0] &= -0.5 \text{ ns} & \text{RIS\_OFFSET}[1] &= 0.4 \text{ ns} \\ \text{RIS\_OFFSET}[2] &= 1.6 \text{ ns} & \text{RIS\_OFFSET}[3] &= 2.6 \text{ ns} \\ \text{RIS\_OFFSET}[4] &= 3.4 \text{ ns} & \text{RIS\_OFFSET}[5] &= 4.5 \text{ ns} \\ \text{RIS\_OFFSET}[6] &= 5.6 \text{ ns} & \text{RIS\_OFFSET}[7] &= 6.4 \text{ ns} \\ \text{RIS\_OFFSET}[8] &= 7.6 \text{ ns} & \text{RIS\_OFFSET}[9] &= 8.5 \text{ ns} \end{aligned}$$

and therefore:

$$\begin{aligned} x[0] &= \text{RIS\_OFFSET}[0] = -0.5 \text{ ns} \\ x[1] &= \text{RIS\_OFFSET}[1] = 0.4 \text{ ns} \\ &\dots \\ x[9] &= \text{RIS\_OFFSET}[9] = 8.5 \text{ ns} \\ x[10] &= 1 \text{ ns} \times 10 + (-0.5) = 9.5 \text{ ns} \\ x[11] &= 1 \text{ ns} \times 10 + 0.4 = 10.4 \text{ ns} \\ &\dots \\ x[19] &= 1 \text{ ns} \times 20 + 8.5 = 18.5 \text{ ns} \\ x[20] &= 1 \text{ ns} \times 20 + (-0.5) = 19.5 \text{ ns} \\ &\dots \end{aligned}$$

### WAVEFORM Commands

Waveforms that have been read in their entirety with the WAVEFORM? query can be sent back into the instrument using WAVEFORM and other, related commands. Since the descriptor contains all of the necessary information, care need not be taken with any of the communication format parameters. The instrument can learn all it needs to know from the waveform.

When synthesizing waveforms for display or comparison, in order

**Note: Waveforms can only be sent back to memory traces (M1, M2, M3, M4). This means possibly removing or changing the prefix (C1 or CHANNEL 1) in the response to the WF? query. See the System Commands for examples.**

to ensure that the descriptor is coherent, read out a waveform of

## 9300 & LC Series

---

the appropriate size and then replace the data with the desired values.

There are many ways to use WAVEFORM and related commands to simplify or speed up work. Among them:

**Partial Waveform Readout:** The WAVEFORM\_SETUP command allows specification of a short part of a waveform for readout, as well as selection of a sparsing factor for reading only every n'th data point.

**Byte Swapping:** The COMM\_ORDER command allows the swapping of the two bytes of data presented in 16-bit word format (can be in the descriptor or in the data/time arrays), when sending the data over the GPIB or RS-232-C remote-control ports. This allows easier data interpretation, depending on the computer system used:

Intel-based computers — the data should be sent with the LSB first, and the command should be CORD LO.

Motorola-based computers — the data should be sent with the MSB first (CORD HI). This is the default at power-up.

***Note: Data written to the scope's Hard Disk Drive, Floppy Drive, or to Memory Card, will always remain in LSB first format, as this is the default DOS format. The CORD command cannot be applied here, as it is only applicable for data sent over the GPIB or RS-232-C ports***

**Data Length, Block Format, and Encoding:** The COMM\_FORMAT command gives control over these parameters. If the extra precision of the lower order byte of the standard data value is not needed, the BYTE option allows a saving of a factor of two on the amount of data to be transmitted or stored. If the computer being used is unable to read binary data, the HEX option allows a response form where the value of each byte is given by a pair of hexadecimal digits.



## Waveform Structure

---

**Data-Only Transfers:** The COMM\_HEADER OFF mode enables a response to WF? DAT1 with the data only (the C1:WF DAT1 will disappear).

If COMM\_FORMAT OFF,BYTE,BIN has also been specified, the response will be mere data bytes (the #90000nnnnn will disappear).

**Formatting for RS-232-C Users:** The COMM\_RS232 command can assist by splitting the very long WF? response into individual lines.

## High-Speed Waveform Transfer

Several important factors need to be taken into account for achieving maximum continuous-data-transfer rates from oscilloscope to controller.

The single most important of these is the limiting of work done in the computer. This effectively means avoiding writing data to disk wherever possible, as well as minimizing operations such as per-data-point computations and reducing the number of calls to the IO system. Ways of doing this include:

Reducing the number of points to be transferred and the number of data bytes per point. The pulse parameter capability and the processing functions can save a great deal of computing and a lot of data transfer time if employed creatively.

Attempting to overlap waveform acquisition with waveform transfer. The oscilloscope is capable of transferring an already acquired or processed waveform after a new acquisition has been started. The total time that the oscilloscope will be able to acquire events will be considerably increased if it is obliged to wait for triggers (livelime).

Minimizing the number of waveform transfers by using Sequence Mode to accumulate many triggers for each transfer. This is preferable to using the WAVEFORM\_SETUP command to reduce the number of data points to be transferred. It also significantly reduces oscilloscope transfer overhead.

### **Example**

The desirable type of command is:

**ARM; WAIT; C1:WF?** to wait for the event, transfer the data, and then start a new acquisition.

This line can be “looped” in the program as soon as it has finished reading the waveform.

## Using Status Registers

**A wide range of status registers allows the oscilloscope's internal processing status to be determined quickly at any time. These registers and the instrument's status reporting system are designed to comply with IEEE 488.2 recommendations. *Following an overview, starting this page, each of the registers and their roles are described.***

Related functions are grouped together in common status registers. Some, such as the Status Byte Register (STB) or the Standard Event Status Register (ESR), are required by the IEEE 488.2 Standard. Other registers are device-specific, and include the Command Error Register (CMR) and Execution Error Register (EXR). Those commands associated with IEEE 488.2 mandatory status registers are preceded by an asterisk <\*>.

### Overview

The Standard Event Status Bit (ESB) and the Internal Status Change Bit (INB) in the Status Byte Register are summary bits of the Standard Event Status Register (ESR) and the Internal State Change Register (INR). The Message Available Bit (MAV) is set whenever there are data bytes in the output queue. The Value Adapted Bit (VAB) indicates that a parameter value was adapted during a previous command interpretation (for example, if the command "TDIV 2.5 US" is received, the timebase is set to 2 m s/div along with the VAB bit).

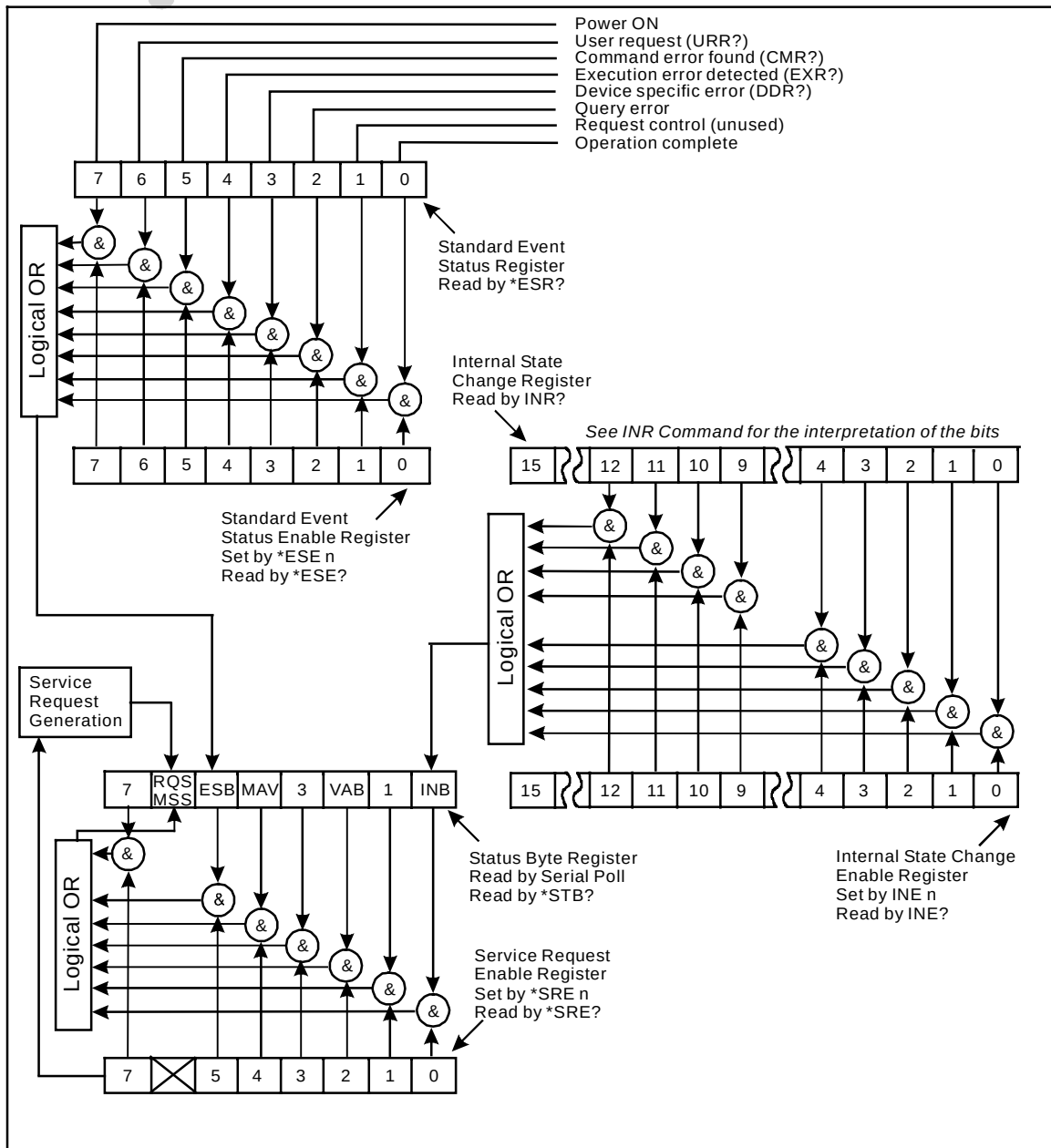
The Master Summary Status bit (MSS) indicates a request for service from the instrument. The MSS bit can only be set if one or more of the other bits of STB are enabled with the Service Request Enable Register (SRE).

All Enable registers (SRE, ESE and INE) are used to generate a bit-wise AND with their associated status registers. The logical OR of this operation is reported to the STB register. At power-on, all Enable registers are zero, inhibiting any reporting to the STB.

The Standard Event Status Register (ESR) primarily summarizes errors, whereas the Internal State Change Register (INR) reports internal changes to the instrument. Additional details of errors reported by ESR can be obtained with the queries "CMR?", "DDR?", "EXR?" and "URR?".



## Status Registers



Status Register Structure

The register structure contains one additional register, not shown in the figure on the previous page. This is the Parallel Poll Enable Register (PRE), which behaves exactly like the Service Request Enable Register (SRE), but sets the “ist” bit (also not shown in the figure), used in the Parallel Poll. The “ist” bit can also be read with the “\*IST?” query.

**Example**

If an erroneous remote command — “TRIG\_MAKE SINGLE”, for example — is transmitted to the instrument, it rejects the command and sets the Command Error Register (CMR) to the value 1 (unrecognized command/query header). The non-zero value of CMR is reported to Bit 5 of the Standard Event Status Register (ESR), which is then set.

Nothing further occurs unless the corresponding Bit 5 of the Standard Event Status Enable Register (ESE) is set (with the command “\*ESE 32”), enabling Bit 5 of ESR to be set for reporting to the summary bit ESB of the Status Byte Register (STB).

If setting of the ESB summary bit in STB is enabled, again nothing occurs unless further reporting is enabled by setting the corresponding bit in the Service Request Enable Register (with the command “\*SRE 32”). In this case, the generation of a non-zero value of CMR ripples through to the Master Summary Status bit (MSS), generating a Service Request (SRQ).

The value of CMR can be read and simultaneously reset to zero at any time with the command “CMR?”. The occurrence of a command error can also be detected by analyzing the response to “\*ESR?”. However, if several types of potential errors must be surveyed, it is usually far more efficient to enable propagation of the errors of interest into the STB with the enable registers ESE and INE.

**Summary**

A command error (CMR) sets Bit 5 of ESR if:

- Bit 5 of ESE is set, ESB of STB is also set, or

- Bit 5 of SRE is set, MSS/RQS of STB is also set and a Service Request is generated.



## Status Registers

---

### Status Byte Register (STB)

The Status Byte Register (STB) is the instrument's central reporting structure. The STB is composed of eight single-bit summary messages (of which three are unused), which reflect the current status of the associated data structures implemented in the instrument:

**Bit 0** is the summary bit INB of the Internal State Change Register. It is set if any of the bits of the INR are set, provided they are enabled by the corresponding bit of the INE register.

**Bit 2** is the Value Adapted bit, indicating that a parameter value was adapted during a previous command interpretation.

**Bit 4** is the Message Available (MAV) bit, indicating that the interface output queue is not empty.

**Bit 5** is the summary bit ESB of the Standard Event Status Register. It is set if any of the bits of the ESR are set, provided they are enabled by the corresponding bit of the ESE register.

**Bit 6** is either the Master Summary Status bit (MSS) or the Request for Service bit (RQS), owing to the STB being able to be read in two different ways. The command “\*STB?” reads and clears the STB in the query mode, in which case Bit 6 is the MSS bit, and indicates whether the instrument has any reason for requesting service.

The Status Byte Register can be read using the query “\*STB?”. The response represents the binary weighted sum of the register bits. The register is cleared by “\*STB?”, “ALST?”, “\*CLS”, or after the instrument has been powered up.

Another way of reading the STB is using the serial poll (see “*Instrument Polls*”, Chapter 2). In this case, Bit 6 is the RQS bit, indicating that the instrument has activated the SRQ line on the GPIB. The serial poll only clears the RQS bit. Therefore, the MSS bit of the STB (and any other bits which caused MSS to be set) will stay set after a serial poll. These bits must be reset.

**Standard Event  
Status Register (ESR)**

The Standard Event Status Register (ESR) is a 16-bit register reflecting the occurrence of events. The ESR bit assignments have been standardized by IEEE 488.2. Only the lower eight bits are currently in use.

The ESR is read using the query “\*ESR?”. The response is the binary weighted sum of the register bits. The register is cleared with an “\*ESR?” or “ALST?” query, a “\*CLS” command or after power-on.

**Example**

The response message “\*ESR 160” indicates that a command error occurred and that the ESR is being read for the first time after power-on. The value 160 can be broken down into 128 (Bit 7) plus 32 (bit 5). *See the table on the same page as the ESR command description for the conditions corresponding to the bits set.*

The “Power ON” bit appears only on the first “\*ESR?” query after power-on because the query clears the register. This type of command error can be determined by reading the Command Error Status Register with the query “CMR?”. Note that it is not necessary to read (nor simultaneously clear) this register in order to set the CMR bit in the ESR on the next command error.

**Standard Event Status  
Enable Register (ESE)**

The Standard Event Status Enable Register (ESE) allows one or more events in the Standard Event Status Register to be reported to the ESB summary bit in the STB.

The ESE is modified with the command “\*ESE” and cleared with the command “\*ESE 0”, or after power-on. It is read with the query “\*ESE?”.

**Example**

“\*ESE 4” sets bit 2 (binary 4) of the ESE Register, enabling query errors to be reported.

**Service Request  
Enable Register (SRE)**

The Service Request Enable Register (SRE) specifies which summary bit(s) in the Status Byte Register will bring about a service request. The SRE consists of eight bits. Setting a bit in this register allows the summary bit located at the same bit position in the Status Byte Register to generate a service request, provided that the associated event becomes true. Bit 6 (MSS) cannot be set and is always reported as zero in response to the query “\*SRE?”.

## Status Registers

### Parallel Poll Enable Register (PRE)

The SRE is modified with the command “\*SRE” and cleared with the command “\*SRE 0”, or after power-on. It may be read with the query “\*SRE?”.

The Parallel Poll Enable Register (PRE) specifies which summary bit(s) in the Status Byte Register will set the “ist” individual local message. This register is quite similar to the Service Request Enable Register (SRE), but is used to set the parallel poll “ist” bit rather than MSS.

The value of the “ist” may also be read without a Parallel Poll via the query “\*IST?”. The response indicates whether or not the “ist” message has been set (values are 1 or 0).

The PRE is modified with the command “\*PRE” and cleared with the command “\*PRE 0”, or after power-on. It is read with the query “\*PRE?”. (See Chapter 2 “Instrument Polls”.)

#### Example

“\*PRE 5” sets bits 2 and 0 (decimal 4 and 1) of the Parallel Poll Enable Register.

### Internal State Change Status Register (INR)

The Internal State Change Status Register (INR) reports the completion of a number of internal operations (*the events tracked by this 16-bit-wide register are listed with the “INR?” query in the System Commands section*).

The INR is read using the query “INR?”. The response is the binary-weighted sum of the register bits. The register is cleared with an “INR?” or “ALST?” query, a “\*CLS” command, or after power-on.

### Internal State Change Enable Register (INE)

The Internal State Change (INE) allows one or more events in the Internal State Change Status Register to be reported to the INB summary bit in the STB.

The INE is modified with the command “INE” and cleared with the command “INE 0”, or after power-on. It is read with the query “INE?”.

### Command Error Status Register (CMR)

The Command Error Status register contains the code of the last command error detected by the instrument. Command error codes are listed with the command “CMR?”.

The Command Error Status Register may be read using the query “CMR?”. The response is the error code. The register is cleared with a “CMR?” or “ALST?” query, a “\*CLS” command, or after power-on.

**Device Dependent Error  
Status Register (DDR)**

The Device Dependent Error Status Register (DDR) indicates the type of hardware errors affecting the instrument. Individual bits in this register report specific hardware failures. They are listed with the command "DDR?".

The DDR is read using the "DDR?" query. The response is the binary weighted sum of the error bits. The register is cleared with a "DDR?" or "ALST?" query, a "\*CLS" command, or after power-on.

**Execution Error Status  
Register (EXR)**

The Execution Error Status Register (EXR) contains the code of the last execution error detected by the instrument. Execution error codes are listed with the command "EXR?".

The EXR is read using the "EXR?" query. The response is the error code. The register is cleared with an "EXR?" or "ALST?" query, a "\*CLS" command, or after power-on.

**User Request Status  
Register (URR)**

The User Request Status Register (URR) contains the identification code of the last menu button pressed. The codes are listed with the command "URR?".

The URR is read using the query "URR?". The response is the decimal code associated with the selected menu button. The register is cleared with a "URR?" or "ALST?" query, a "\*CLS" command, or after power-on.

## About These Commands & Queries

This section lists and describes the remote control commands and queries recognized by the instrument. All commands and queries can be executed in either local or remote state. Where not included here, those for special options can be found in the options' dedicated Operator's Manuals.

The description for each command or query, with syntax and other information, begins on a new page. The name (header) is given in both long and short form at the top of the page, and the subject is indicated as a command or query or both. Queries perform actions such as obtaining information, and are recognized by the question mark (?) following the header.

### How They are Listed

The descriptions are listed in alphabetical order according to their *long* form. Thus the description of ATTENUATION, whose short form is ATTN, is listed before that of AUTO SETUP, whose short form is ASET. The two special indexes at the beginning of this section are designed as reference aids for quickly finding commands and queries. One lists the commands and queries in alphabetical order according to long form, while the other groups them according to subsystem or category.

### How They are Described

In the descriptions themselves, a brief explanation of the function performed is given. This is followed by a presentation of the formal syntax, with the header given in **Upper-and-Lower-Case** characters and the short form derived from it in **ALL UPPER-CASE** characters. Where applicable, the syntax of the query is given with the format of its response.

A short example illustrating a typical use is also presented. The GPIB examples assume that the controller is equipped with a *National Instruments* interface board, which shows calls to the related interface subroutines in BASIC. The device name of the instrument is defined as SCOPE%.

### When Can They be Used?

The commands and queries listed here can be used with all LeCroy digital instruments. However, the raised hand symbol G indicates a note on availability for a particular model or option.

## 9300 &amp; LC Series

## 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 may be used in a command instead of the placeholder.
- { } Braces enclose a list of choices, one of which one must be made.
- [ ] Square brackets enclose optional items.
- ... An ellipsis indicates that the items both to its left and right may be repeated a number of times.

As an example, consider the syntax notation for the command to set the vertical input sensitivity:

<channel> : **VOLT\_DIV** <v\_gain>

<channel> : = {**C1**, **C2**}

<v\_gain> : = 5.0 mV to 2.5 V

The first line shows the formal appearance of the command, with <channel> denoting the placeholder for the header path and <v\_gain> the placeholder for the data parameter specifying the desired vertical gain value. The second line indicates that either **C1** or **C2** must be chosen for the header path. And the third explains that the actual vertical gain can be set to any value between 5 mV and 2.5 V.

*Refer to Chapter 1 for an overview of the command functions and notation used.*



## Commands & Queries Tabled by Long Form

| <b>Short</b> | <b>Long Form</b> | <b>Subsystem</b> | <b>What the Command/Query Does</b>                        |
|--------------|------------------|------------------|-----------------------------------------------------------|
| ALST?        | ALL_STATUS?      | STATUS           | Reads and clears the contents of all status registers.    |
| ARM          | ARM_ACQUISITION  | ACQUISITION      | Changes acquisition state from "stopped" to "single".     |
| ATTN         | ATTENUATION      | ACQUISITION      | Selects the vertical attenuation factor of the probe.     |
| ACAL         | AUTO_CALIBRATE   | MISCELLANEOUS    | Enables or disables automatic calibration.                |
| ASCR         | AUTO_SCROLL      | DISPLAY          | Controls the Auto Scroll viewing feature.                 |
| ASET         | AUTO_SETUP       | ACQUISITION      | Adjusts vertical, timebase and trigger parameters.        |
| BWL          | BANDWIDTH_LIMIT  | ACQUISITION      | Enables/disables the bandwidth-limiting low-pass filter.  |
| BUZZ         | BUZZER           | MISCELLANEOUS    | Controls the built-in piezo-electric buzzer.              |
| *CAL?        | *CAL?            | MISCELLANEOUS    | Performs complete internal calibration of the instrument. |
| COUT         | CAL_OUTPUT       | MISCELLANEOUS    | Sets signal type put out at the CAL BNC connector.        |
| CHST         | CALL_HOST        | DISPLAY          | Allows manual generation of a service request (SRQ).      |
| CLM          | CLEAR_MEMORY     | FUNCTION         | Clears the specified memory.                              |
| CLSW         | CLEAR_SWEEPS     | FUNCTION         | Restarts the cumulative processing functions.             |
| *CLS         | *CLS             | STATUS           | Clears all status data registers.                         |
| CMR?         | CMR?             | STATUS           | Reads and clears the Command error Register (CMR).        |
| COLR         | COLOR            | DISPLAY          | Selects color of individual on-screen objects             |
| CSCH         | COLOR_SCHEME     | DISPLAY          | Selects the display color scheme.                         |
| COMB         | COMBINE_CHANNELS | ACQUISITION      | Controls the channel interleaving function.               |
| 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.             |
| CONET        | COMM_NET         | COMMUNICATION    | Specifies network addresses of scope and printers.        |
| CORD         | COMM_ORDER       | COMMUNICATION    | Controls the byte order of waveform data transfers.       |
| CORS         | COMM_RS232       | COMMUNICATION    | Sets remote control parameters of the RS-232-C port.      |
| CPL          | COUPLING         | ACQUISITION      | Selects the specified input channel's coupling mode.      |
| CRMS         | CURSOR_MEASURE   | CURSOR           | Specifies the type of cursor/parameter measurement.       |
| CRRD         | CURSOR_READOUT   | CURSOR           | Sets the cursor amplitude in volts or dBm.                |
| CRST?        | CURSOR_SET?      | CURSOR           | Allows positioning of any one of eight cursors.           |
| CRVA?        | CURSOR_VALUE?    | CURSOR           | Returns trace values measured by specified cursors.       |
| DPNT         | DATA_POINTS      | DISPLAY          | Controls bold/single pixel display of sample points.      |
| DATE         | DATE             | MISCELLANEOUS    | Changes the date/time of the internal real-time clock.    |
| DDR?         | DDR?             | STATUS           | Reads, clears the Device Dependent Register (DDR).        |
| DEF          | DEFINE           | FUNCTION         | Specifies math expression for function evaluation.        |

## 9300 &amp; LC Series

| <b>Short</b>  | <b>Long Form</b>         | <b>Subsystem</b> | <b>What the Command/Query Does</b>                      |
|---------------|--------------------------|------------------|---------------------------------------------------------|
| <b>DELF</b>   | <b>DELETE_FILE</b>       | MASS STORAGE     | Deletes files from mass storage.                        |
| <b>DIR</b>    | <b>DIRECTORY</b>         | MASS STORAGE     | Creates and deletes file directories.                   |
| <b>DISP</b>   | <b>DISPLAY</b>           | DISPLAY          | Controls the display screen.                            |
| <b>DTJN</b>   | <b>DOT_JOIN</b>          | DISPLAY          | Controls the interpolation lines between data points.   |
| <b>DZOM</b>   | <b>DUAL_ZOOM</b>         | DISPLAY          | Sets horizontal magnification and positioning.          |
| <b>EKEY</b>   | <b>ENABLE_KEY</b>        | DISPLAY          | Allows use of the KEY command in local mode.            |
| <b>*ESE</b>   | <b>*ESE</b>              | STATUS           | Sets the Standard Event Status Enable register(ESE).    |
| <b>*ESR?</b>  | <b>*ESR?</b>             | STATUS           | Reads, clears the Event Status Register (ESR).          |
| <b>EXR?</b>   | <b>EXR?</b>              | STATUS           | Reads, clears the EXecution error Register (EXR).       |
| <b>FATC</b>   | <b>FAT_CURSOR</b>        | DISPLAY          | Controls width of cursors.                              |
| <b>FLNM</b>   | <b>FILENAME</b>          | MASS STORAGE     | Changes default filenames.                              |
| <b>FCR</b>    | <b>FIND_CTR_RANGE</b>    | FUNCTION         | Automatically sets the center and width of a histogram. |
| <b>FCRD</b>   | <b>FORMAT_CARD</b>       | MISCELLANEOUS    | Formats the memory card.                                |
| <b>FFLP</b>   | <b>FORMAT_FLOPPY</b>     | MISCELLANEOUS    | Formats a floppy disk.                                  |
| <b>FHDD</b>   | <b>FORMAT_HDD</b>        | MASS STORAGE     | Formats the removable hard disk.                        |
| <b>FVDISK</b> | <b>FORMAT_VDISK</b>      | MASS STORAGE     | Formats the non-volatile RAM (virtual disk).            |
| <b>FSCR</b>   | <b>FULL_SCREEN</b>       | DISPLAY          | Selects magnified view format for the grid.             |
| <b>FRST</b>   | <b>FUNCTION_RESET</b>    | FUNCTION         | Resets a waveform-processing function.                  |
| <b>GBWL</b>   | <b>GLOBAL_BWL</b>        | ACQUISITION      | Enables/disables the Global Bandwidth Limit.            |
| <b>GRID</b>   | <b>GRID</b>              | DISPLAY          | Specifies single-, dual- or quad-mode grid display.     |
| <b>HCSU</b>   | <b>HARDCOPY_SETUP</b>    | HARD COPY        | Configures the hard-copy driver.                        |
| <b>HCTR</b>   | <b>HARDCOPY_TRANSMIT</b> | HARD COPY        | Sends string of ASCII characters to hard-copy unit.     |
| <b>HMAG</b>   | <b>HOR_MAGNIFY</b>       | DISPLAY          | Horizontally expands the selected expansion trace.      |
| <b>HPOS</b>   | <b>HOR_POSITION</b>      | DISPLAY          | Horizontally positions intensified zone's center.       |
| <b>*IDN?</b>  | <b>*IDN?</b>             | MISCELLANEOUS    | For identification purposes.                            |
| <b>INE</b>    | <b>INE</b>               | STATUS           | Sets the Internal state change Enable register (INE).   |
| <b>INR?</b>   | <b>INR?</b>              | STATUS           | Reads, clears INternal state change Register (INR).     |
| <b>INSP?</b>  | <b>INSPECT?</b>          | WAVEFORM TRANS.  | Allows acquired waveform parts to be read.              |
| <b>INTS</b>   | <b>INTENSITY</b>         | DISPLAY          | Sets the grid or trace/text intensity level.            |
| <b>ILVD</b>   | <b>INTERLEAVED</b>       | ACQUISITION      | Enables/disables random interleaved sampling (RIS).     |
| <b>IST?</b>   | <b>IST?</b>              | STATUS           | Reads the current state of the IEEE 488.                |
| <b>KEY</b>    | <b>KEY</b>               | DISPLAY          | Displays a string in the menu field.                    |
| <b>LOGO</b>   | <b>LOGO</b>              | DISPLAY          | Displays LeCroy logo at top of grid.                    |
| <b>MASK</b>   | <b>MASK</b>              | CURSOR           | Invokes PolyMask draw and fill tools.                   |
| <b>MLIM</b>   | <b>MATH_LIMITS</b>       | FUNCTIONS        | Limits averaging to only the points of interest.        |
| <b>MGAT</b>   | <b>MEASURE_GATE</b>      | DISPLAY          | Controls highlighting of the measurement gate region.   |
| <b>MSIZ</b>   | <b>MEMORY_SIZE</b>       | ACQUISITION      | Selects max. memory length.                             |

| <b>Short</b> | <b>Long Form</b>      | <b>Subsystem</b> | <b>What the Command/Query Does</b>                         |
|--------------|-----------------------|------------------|------------------------------------------------------------|
| MSG          | MESSAGE               | DISPLAY          | Displays a string of characters in the message field.      |
| MZOM         | MULTI_ZOOM            | DISPLAY          | Sets horizontal magnification and positioning.             |
| OFST         | OFFSET                | ACQUISITION      | Allows output channel vertical offset adjustment.          |
| OFCT         | OFFSET_CONSTANT       | ACQUISITION      | Sets offset in volts or divisions.                         |
| *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.                        |
| 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 current statistics parameter values.               |
| PAVA?        | PARAMETER_VALUE?      | CURSOR           | Returns current parameter, mask test values.               |
| PFCO         | PASS_FAIL_CONDITION   | CURSOR           | Adds a Pass/Fail test condition or custom parameter.       |
| PFCT         | PASS_FAIL_COUNTER     | CURSOR           | Resets the Pass/Fail acquisition counters.                 |
| PFDO         | PASS_FAIL_DO          | CURSOR           | Defines desired outcome, actions after Pass/Fail test.     |
| PFMS         | PASS_FAIL_MASK        | CURSOR           | Generates tolerance mask on a trace and stores it.         |
| PFST?        | PASS_FAIL_STATUS?     | CURSOR           | Returns the Pass/Fail test for a given line number.        |
| PDET         | PEAK_DETECT           | ACQUISITION      | Switches the peak detector ON and OFF.                     |
| PECS         | PER_CURSOR_SET        | CURSOR           | Positions independent cursors.                             |
| PECV?        | PER_CURSOR_VALUE?     | CURSOR           | Returns values measured by cursors.                        |
| PERS         | PERSIST               | DISPLAY          | Enables or disables the persistence display mode.          |
| PECL         | PERSIST_COLOR         | DISPLAY          | Controls color rendering method of persistence traces.     |
| 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).              |
| PRCA?        | PROBE_CAL?            | PROBES           | Performs auto-calibration of connected current probe.      |
| PRDG?        | PROBE_DEGAUSS?        | PROBES           | Degausses and calibrates the connected current probe.      |
| PRIT?        | PROBE_INFOTEXT        | PROBES           | Returns the connected probe's informative text.            |
| PRNA         | PROBE_NAME?           | PROBES           | Names the probe connected to the instrument.               |
| *RCL         | *RCL                  | SAVE/RECALL      | Recalls one of five non-volatile panel setups.             |
| ROUT         | REAR_OUTPUT           | MISCELLANEOUS    | Sets the type of signal put out at the rear BNC connector. |
| REC          | RECALL                | WAVEFORM TRANS.  | Recalls a file from mass storage to internal memory.       |
| RCPN         | RECALL_PANEL          | SAVE/RECALL      | Recalls a front-panel setup from mass storage.             |
| RCLK         | REFERENCE_CLOCK       | ACQUISITION      | Selects the system clock source: internal or external.     |
| *RST         | *RST                  | SAVE/RECALL      | The *RST command initiates a device reset.                 |
| SCLK         | SAMPLE_CLOCK          | ACQUISITION      | Allows control of an external timebase.                    |

## 9300 &amp; LC Series

| <b>Short</b>  | <b>Long Form</b>      | <b>Subsystem</b>  | <b>What the Command/Query Does</b>                                              |
|---------------|-----------------------|-------------------|---------------------------------------------------------------------------------|
| <b>*SAV</b>   | <b>*SAV</b>           | SAVE/RECALL       | Stores current state in non-volatile internal memory.                           |
| <b>SCREEN</b> | <b>SCREEN</b>         | DISPLAY           | Turns the screen of or off.                                                     |
| <b>SCDP</b>   | <b>SCREEN_DUMP</b>    | HARD COPY         | Causes a screen dump to the hard-copy device.                                   |
| <b>SCSV</b>   | <b>SCREEN_SAVE</b>    | DISPLAY           | Controls the automatic screen saver.                                            |
| <b>SEL</b>    | <b>SELECT</b>         | DISPLAY           | Selects the specified trace for manual display control.                         |
| <b>SEQ</b>    | <b>SEQUENCE</b>       | ACQUISITION       | Sets the conditions for the sequence mode acquisition.                          |
| <b>SKEY</b>   | <b>SKEY</b>           | DISPLAY           | Allows you to assign text, borders, and files to Custom DSO menu soft keys.     |
| <b>SLEEP</b>  | <b>SLEEP</b>          | MISCELLANEOUS     | Makes the scope wait before it interprets new commands.                         |
| <b>*SRE</b>   | <b>*SRE</b>           | STATUS            | Sets the Service Request Enable register (SRE).                                 |
| <b>*STB?</b>  | <b>*STB?</b>          | STATUS            | Reads the contents of IEEE 488.                                                 |
| <b>STITLE</b> | <b>STITLE</b>         | DISPLAY           | Allows you to title a panel of menus.                                           |
| <b>STOP</b>   | <b>STOP</b>           | ACQUISITION       | Immediately stops signal acquisition.                                           |
| <b>STO</b>    | <b>STORE</b>          | WAVEFORM TRANS.   | Stores a trace in internal memory or mass storage.                              |
| <b>STPN</b>   | <b>STORE_PANEL</b>    | SAVE/RECALL       | Stores front-panel setup to mass storage.                                       |
| <b>STST</b>   | <b>STORE_SETUP</b>    | WAVEFORM TRANS.   | Controls the way in which traces are stored.                                    |
| <b>STTM</b>   | <b>STORE_TEMPLATE</b> | WAVEFORM TRANS.   | Stores the waveform template to mass storage.                                   |
| <b>TMPL?</b>  | <b>TEMPLATE?</b>      | WAVEFORM TRANS.   | Produces a complete waveform template copy.                                     |
| <b>TDISP</b>  | <b>TIME_DISPLAY</b>   | MISCELLANEOUS     | Changes time display from current to trigger or none.                           |
| <b>TDIV</b>   | <b>TIME_DIV</b>       | ACQUISITION       | Modifies the timebase setting.                                                  |
| <b>TRA</b>    | <b>TRACE</b>          | DISPLAY           | Enables or disables the display of a trace.                                     |
| <b>TRLB</b>   | <b>TRACE_LABEL</b>    | DISPLAY           | Allows you to enter a label for a trace.                                        |
| <b>TOPA</b>   | <b>TRACE_OPACITY</b>  | DISPLAY           | Controls the opacity of the trace color.                                        |
| <b>TORD</b>   | <b>TRACE_ORDER</b>    | DISPLAY           | Allows you to specify the display order of traces.                              |
| <b>TRFL</b>   | <b>TRANSFER_FILE</b>  | WAVEFORM TRANSFER | Transfers ASCII files to and from storage media, or between scope and computer. |
| <b>*TRG</b>   | <b>*TRG</b>           | ACQUISITION       | Executes an ARM command.                                                        |
| <b>TRCP</b>   | <b>TRIG_COUPLING</b>  | ACQUISITION       | Sets the coupling mode of the specified trigger source.                         |
| <b>TRDL</b>   | <b>TRIG_DELAY</b>     | ACQUISITION       | Sets the time at which the trigger is to occur.                                 |
| <b>TRLV</b>   | <b>TRIG_LEVEL</b>     | ACQUISITION       | Adjusts the trigger level of the specified trigger source.                      |
| <b>TRLV2</b>  | <b>TRIG_LEVEL_2</b>   | ACQUISITION       | Adjusts the second trigger threshold reference level.                           |
| <b>TRMD</b>   | <b>TRIG_MODE</b>      | ACQUISITION       | Specifies the trigger mode.                                                     |
| <b>TRPA</b>   | <b>TRIG_PATTERN</b>   | ACQUISITION       | Defines a trigger pattern.                                                      |
| <b>TRSE</b>   | <b>TRIG_SELECT</b>    | ACQUISITION       | Selects the condition that will trigger acquisition.                            |
| <b>TRSL</b>   | <b>TRIG_SLOPE</b>     | ACQUISITION       | Sets the trigger slope of the specified trigger source.                         |
| <b>TRWI</b>   | <b>TRIG_WINDOW</b>    | ACQUISITION       | Sets window amplitude on current Edge trigger source.                           |
| <b>*TST?</b>  | <b>*TST?</b>          | MISCELLANEOUS     | Performs an internal self-test.                                                 |
| <b>URR?</b>   | <b>URR?</b>           | STATUS            | Reads, clears User Request status Register (URR).                               |

| <b>Short</b> | <b>Long Form</b>        | <b>Subsystem</b> | <b>What the Command/Query Does</b>                     |
|--------------|-------------------------|------------------|--------------------------------------------------------|
| <b>VMAG</b>  | <b>VERT_MAGNIFY</b>     | DISPLAY          | Vertically expands the specified trace.                |
| <b>VPOS</b>  | <b>VERT_POSITION</b>    | DISPLAY          | Adjusts the vertical position of the specified trace.  |
| <b>VDIV</b>  | <b>VOLT_DIV</b>         | ACQUISITION      | Sets the vertical sensitivity.                         |
| <b>*WAI</b>  | <b>*WAI</b>             | STATUS           | Required by the IEEE 488.                              |
| <b>WAIT</b>  | <b>WAIT</b>             | ACQUISITION      | Prevents new analysis until current is completed.      |
| <b>WF</b>    | <b>WAVEFORM</b>         | WAVEFORM TRANS.  | Transfers a waveform from controller to scope.         |
| <b>WFSU</b>  | <b>WAVEFORM_SETUP</b>   | WAVEFORM TRANS.  | Specifies amount of waveform data to go to controller. |
| <b>WFTX</b>  | <b>WAVEFORM_TEXT</b>    | WAVEFORM TRANS.  | Documents acquisition conditions.                      |
| <b>XYAS?</b> | <b>XY_ASSIGN?</b>       | DISPLAY          | Returns traces currently assigned to the XY display.   |
| <b>XYCO</b>  | <b>XY_CURSOR_ORIGIN</b> | CURSOR           | Sets origin position of absolute cursor measurements.  |
| <b>XYCS</b>  | <b>XY_CURSOR_SET</b>    | CURSOR           | Allows positioning of XY voltage cursors.              |
| <b>XYCV?</b> | <b>XY_CURSOR_VALUE?</b> | CURSOR           | Returns the current values of the X vs. Y cursors.     |
| <b>XYDS</b>  | <b>XY_DISPLAY</b>       | DISPLAY          | Enables or disables the XY display mode.               |
| <b>XYRD</b>  | <b>XY_RENDER</b>        | DISPLAY          | Controls XY plot: smooth or disconnected points.       |
| <b>XYSA</b>  | <b>XY_SATURATION</b>    | DISPLAY          | Sets persistence color saturation level in XY display. |

## Commands & Queries Tabled by Subsystem

| <b>Short</b>                                                        | <b>Long Form</b>        | <b>What the Command/Query Does</b>                                  |
|---------------------------------------------------------------------|-------------------------|---------------------------------------------------------------------|
| <b><i>ACQUISITION — Controlling Waveform Acquisition</i></b>        |                         |                                                                     |
| <b>ARM</b>                                                          | <b>ARM_ACQUISITION</b>  | Changes acquisition state from “stopped” to “single”.               |
| <b>ASET</b>                                                         | <b>AUTO_SETUP</b>       | Adjusts vertical, timebase and trigger parameters for display.      |
| <b>ATTN</b>                                                         | <b>ATTENUATION</b>      | Selects the vertical attenuation factor of the probe.               |
| <b>BWL</b>                                                          | <b>BANDWIDTH_LIMIT</b>  | Enables or disables the bandwidth-limiting low-pass filter.         |
| <b>COMB</b>                                                         | <b>COMBINE_CHANNELS</b> | Controls the acquisition system’s channel-interleaving function.    |
| <b>CPL</b>                                                          | <b>COUPLING</b>         | Selects the coupling mode of the specified input channel.           |
| <b>GBWL</b>                                                         | <b>GLOBAL_BWL</b>       | Enables/disables the Global Bandwidth Limit                         |
| <b>ILVD</b>                                                         | <b>INTERLEAVED</b>      | Enables or disables random interleaved sampling (RIS).              |
| <b>MSIZ</b>                                                         | <b>MEMORY_SIZE</b>      | Allows selection of maximum memory length (M-, L-models only).      |
| <b>OFST</b>                                                         | <b>OFFSET</b>           | Allows vertical offset adjustment of the specified input channel.   |
| <b>OFCT</b>                                                         | <b>OFFSET_CONSTANT</b>  | Sets offset in volts or divisions.                                  |
| <b>PDET</b>                                                         | <b>PEAK_DETECT</b>      | Switches the peak detector ON and OFF.                              |
| <b>RCLK</b>                                                         | <b>REFERENCE_CLOCK</b>  | Selects the system clock source: internal or external.              |
| <b>SCLK</b>                                                         | <b>SAMPLE_CLOCK</b>     | Allows control of an external timebase.                             |
| <b>SEQ</b>                                                          | <b>SEQUENCE</b>         | Sets the conditions for Sequence-mode acquisition.                  |
| <b>STOP</b>                                                         | <b>STOP</b>             | Immediately stops signal acquisition.                               |
| <b>TDIV</b>                                                         | <b>TIME_DIV</b>         | Modifies the timebase setting.                                      |
| <b>TRCP</b>                                                         | <b>TRIG_COUPLING</b>    | Sets the coupling mode of the specified trigger source.             |
| <b>TRDL</b>                                                         | <b>TRIG_DELAY</b>       | Sets the time at which the trigger is to occur.                     |
| <b>*TRG</b>                                                         | <b>*TRG</b>             | Executes an ARM command.                                            |
| <b>TRLV</b>                                                         | <b>TRIG_LEVEL</b>       | Adjusts the level of the specified trigger source.                  |
| <b>TRLV2</b>                                                        | <b>TRIG_LEVEL_2</b>     | Adjusts the second trigger threshold reference level.               |
| <b>TRMD</b>                                                         | <b>TRIG_MODE</b>        | Specifies Trigger mode.                                             |
| <b>TRPA</b>                                                         | <b>TRIG_PATTERN</b>     | Defines a trigger pattern.                                          |
| <b>TRSE</b>                                                         | <b>TRIG_SELECT</b>      | Selects the condition that will trigger acquisition.                |
| <b>TRSL</b>                                                         | <b>TRIG_SLOPE</b>       | Sets the slope of the specified trigger source.                     |
| <b>TRWI</b>                                                         | <b>TRIG_WINDOW</b>      | Sets the window amplitude in volts on current Edge trigger source.  |
| <b>VDIV</b>                                                         | <b>VOLT_DIV</b>         | Sets the vertical sensitivity in volts/div.                         |
| <b>WAIT</b>                                                         | <b>WAIT</b>             | Prevents new command analysis until current acquisition completion. |
| <b><i>COMMUNICATION — Setting Communication Characteristics</i></b> |                         |                                                                     |
| <b>CFMT</b>                                                         | <b>COMM_FORMAT</b>      | Selects the format to be used for sending waveform data.            |
| <b>CHDR</b>                                                         | <b>COMM_HEADER</b>      | Controls formatting of query responses.                             |
| <b>CHLP</b>                                                         | <b>COMM_HELP</b>        | Controls operational level of the RC Assistant.                     |

| <b>Short</b>                                   | <b>Long Form</b>             | <b>What the Command/Query Does</b>                                      |
|------------------------------------------------|------------------------------|-------------------------------------------------------------------------|
| <b>CHL</b>                                     | <b>COMM_HELP_LOG</b>         | Returns the contents of the RC Assistant log.                           |
| <b>CONET</b>                                   | <b>COMM_NET</b>              | Specifies network addresses of scope and printers.                      |
| <b>CORD</b>                                    | <b>COMM_ORDER</b>            | Controls the byte order of waveform data transfers.                     |
| <b>CORS</b>                                    | <b>COMM_RS232</b>            | Sets remote control parameters of the RS-232-C port.                    |
| <b><i>CURSOR — Performing Measurements</i></b> |                              |                                                                         |
| <b>CRMS</b>                                    | <b>CURSOR_MEASURE</b>        | Specifies type of cursor or parameter measurement for display.          |
| <b>CRRD</b>                                    | <b>CURSOR_READ</b>           | Sets the cursor amplitude in volts or dBm.                              |
| <b>CRST?</b>                                   | <b>CURSOR_SET?</b>           | Allows positioning of any one of eight independent cursors.             |
| <b>CRVA?</b>                                   | <b>CURSOR_VALUE?</b>         | Returns values measured by specified cursors for a given trace.         |
| <b>MASK</b>                                    | <b>MASK</b>                  | Invokes PolyMask draw and fill tools.                                   |
| <b>PACL</b>                                    | <b>PARAMETER_CLR</b>         | Clears all current parameters in Custom and Pass/Fail modes.            |
| <b>PADL</b>                                    | <b>PARAMETER_DELETE</b>      | Deletes a specified parameter in Custom and Pass/Fail modes.            |
| <b>PAST?</b>                                   | <b>PARAMETER_STATISTICS?</b> | Returns current statistics values for specified pulse parameter.        |
| <b>PAVA?</b>                                   | <b>PARAMETER_VALUE?</b>      | Returns current value(s) of parameter(s) and mask tests.                |
| <b>PECS</b>                                    | <b>PER_CURSOR_SET</b>        | Allows positioning of any one of six independent cursors.               |
| <b>PECV?</b>                                   | <b>PER_CURSOR_VALUE?</b>     | Returns values measured by specified cursors for a given trace.         |
| <b>PFCO</b>                                    | <b>PASS_FAIL_CONDITION</b>   | Adds a Pass/Fail test condition or custom parameter to display.         |
| <b>PFCT</b>                                    | <b>PASS_FAIL_COUNTER</b>     | Resets the Pass/Fail acquisition counters.                              |
| <b>PFDO</b>                                    | <b>PASS_FAIL_DO</b>          | Defines desired outcome and actions following a Pass/Fail test.         |
| <b>PFMS</b>                                    | <b>PASS_FAIL_MASK</b>        | Generates a tolerance mask around a chosen trace and stores it.         |
| <b>PFST?</b>                                   | <b>PASS_FAIL_STATUS?</b>     | Returns the Pass/Fail test for a given line number.                     |
| <b>XYCO</b>                                    | <b>XY_CURSOR_ORIGIN</b>      | Sets position of origin for absolute cursor measurements on XY display. |
| <b>XYCS</b>                                    | <b>XY_CURSOR_SET</b>         | Allows positioning of any of six independent XY voltage cursors.        |
| <b>XYCV?</b>                                   | <b>XY_CURSOR_VALUE?</b>      | Returns current values of X vs. Y cursors.                              |
| <b><i>DISPLAY — Displaying Waveforms</i></b>   |                              |                                                                         |
| <b>ASCR</b>                                    | <b>AUTO_SCROLL</b>           | Controls the Auto Scroll viewing feature.                               |
| <b>CHST</b>                                    | <b>CALL_HOST</b>             | Allows manual generation of a service request (SRQ).                    |
| <b>COLR</b>                                    | <b>COLOR</b>                 | Selects color of individual objects: traces, grids or cursors.          |
| <b>CSCH</b>                                    | <b>COLOR_SCHEME</b>          | Selects the display color scheme.                                       |
| <b>DPNT</b>                                    | <b>DATA_POINTS</b>           | Controls display of sample points in single display pixels or bold.     |
| <b>DISP</b>                                    | <b>DISPLAY</b>               | Controls the oscilloscope display screen.                               |
| <b>DTJN</b>                                    | <b>DOT_JOIN</b>              | Controls the interpolation lines between data points.                   |
| <b>DZOM</b>                                    | <b>DUAL_ZOOM</b>             | Sets horiz. magnification and positioning for all expanded traces.      |
| <b>EKEY</b>                                    | <b>EKEY</b>                  | Allows use of the KEY command in local mode.                            |
| <b>FATC</b>                                    | <b>FAT_CURSOR</b>            | Controls width of cursors.                                              |
| <b>FSCR</b>                                    | <b>FULL_SCREEN</b>           | Selects magnified view format for the grid.                             |
| <b>GRID</b>                                    | <b>GRID</b>                  | Specifies grid display in single, dual or quad mode.                    |

## 9300 &amp; LC Series

| <b>Short</b>                                                   | <b>Long Form</b> | <b>What the Command/Query Does</b>                                          |
|----------------------------------------------------------------|------------------|-----------------------------------------------------------------------------|
| HMAG                                                           | HOR_MAGNIFY      | Horizontally expands selected expansion trace.                              |
| HPOS                                                           | HOR_POSITION     | Horizontally positions intensified zone's center on source trace.           |
| INTS                                                           | INTENSITY        | Sets grid or trace/text intensity level.                                    |
| KEY                                                            | KEY              | Displays a string in the menu field.                                        |
| LOGO                                                           | LOGO             | Displays the LeCroy logo at the top of the grid.                            |
| MGAT                                                           | MEASURE_GATE     | Controls highlighting of region between parameter cursors.                  |
| MSG                                                            | MESSAGE          | Displays a string of characters in the message field.                       |
| MZOM                                                           | MULTI_ZOOM       | Sets horiz. magnification and positioning for all expanded traces.          |
| 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.                   |
| SCREEN                                                         | SCREEN           | Turns the screen on or off.                                                 |
| SCSV                                                           | SCREEN_SAVE      | Controls the automatic screen saver.                                        |
| SEL                                                            | SELECT           | Selects the specified trace for manual display control.                     |
| SKEY                                                           | SKEY             | Allows you to assign text, borders, and files to Custom DSO menu soft keys. |
| STITLE                                                         | STITLE           | Allows you to title a panel of menus.                                       |
| TRA                                                            | TRACE            | Enables or disables the display of a trace.                                 |
| TRLB                                                           | TRACE_LABEL      | Allows you to enter a label for a trace.                                    |
| TOPA                                                           | TRACE_OPACITY    | Controls the opacity of the trace color.                                    |
| TORD                                                           | TRACE_ORDER      | Allows you to specify the display order of traces.                          |
| VMAG                                                           | VERT_MAGNIFY     | Vertically expands the specified trace.                                     |
| VPOS                                                           | VERT_POSITION    | Adjusts the vertical position of the specified trace.                       |
| XYAS?                                                          | XY_ASSIGN?       | Returns the traces currently assigned to the XY display.                    |
| XYDS                                                           | XY_DISPLAY       | Enables or disables the XY display mode.                                    |
| XYRD                                                           | XY_RENDER        | Controls XY plot: smooth or disconnected points.                            |
| XYSA                                                           | XY_SATURATION    | Sets persistence color saturation level in XY display.                      |
| <b>FUNCTION — Performing Waveform Mathematical Operations</b>  |                  |                                                                             |
| CLM                                                            | CLEAR_MEMORY     | Clears the specified memory.                                                |
| CLSW                                                           | CLEAR_SWEEPS     | Restarts the cumulative processing functions.                               |
| DEF                                                            | DEFINE           | Specifies the math expression to be evaluated by a function.                |
| FRST                                                           | FUNCTION_RESET   | Resets a waveform processing function.                                      |
| MLIM                                                           | MATH_LIMITS      | Limits averaging to only the points of interest.                            |
| <b>HARD COPY — Printing the Contents of the Display Screen</b> |                  |                                                                             |
| HCSU                                                           | HARDCOPY_SETUP   | Configures the hard-copy driver.                                            |



| <b>Short</b>                                                             | <b>Long Form</b>         | <b>What the Command/Query Does</b>                                 |
|--------------------------------------------------------------------------|--------------------------|--------------------------------------------------------------------|
| <b>HCTR</b>                                                              | <b>HARDCOPY_TRANSMIT</b> | Sends string of unmodified ASCII characters to the hard-copy unit. |
| <b>SCDP</b>                                                              | <b>SCREEN_DUMP</b>       | Causes a screen dump to the hard-copy device.                      |
| <b>MASS STORAGE — Creating and Deleting File Directories</b>             |                          |                                                                    |
| <b>DELF</b>                                                              | <b>DELETE_FILE</b>       | Deletes files from currently selected directory on mass storage.   |
| <b>DIR</b>                                                               | <b>DIRECTORY</b>         | Creates and deletes file directories on mass-storage devices.      |
| <b>FCRD</b>                                                              | <b>FORMAT_CARD</b>       | Formats the memory card.                                           |
| <b>FCR</b>                                                               | <b>FIND_CTR_RANGE</b>    | Automatically sets the center and width of a histogram.            |
| <b>FFLP</b>                                                              | <b>FORMAT_FLOPPY</b>     | Formats a floppy disk in the Double- or High-Density format.       |
| <b>FHDD</b>                                                              | <b>FORMAT_HDD</b>        | Formats the removable hard disk.                                   |
| <b>FLNM</b>                                                              | <b>FILENAME</b>          | Changes default filename of any stored trace, setup or hard copy.  |
| <b>FVDISK</b>                                                            | <b>FORMAT_VDISK</b>      | Formats the non-volatile RAM (virtual disk).                       |
| <b>SLEEP</b>                                                             | <b>SLEEP</b>             | Makes the scope wait before it interprets new commands.            |
| <b>MISCELLANEOUS — Calibration and Testing</b>                           |                          |                                                                    |
| <b>ACAL</b>                                                              | <b>AUTO_CALIBRATE</b>    | Enables or disables automatic calibration.                         |
| <b>BUZZ</b>                                                              | <b>BUZZER</b>            | Controls the built-in piezo-electric buzzer.                       |
| <b>*CAL?</b>                                                             | <b>*CAL?</b>             | Performs a complete internal calibration of the instrument.        |
| <b>COUT</b>                                                              | <b>CAL_OUTPUT</b>        | Sets the type of signal put out at the CAL BNC connector.          |
| <b>DATE</b>                                                              | <b>DATE</b>              | Changes the date/time of the internal real-time clock.             |
| <b>*IDN?</b>                                                             | <b>*IDN?</b>             | Used for identification purposes.                                  |
| <b>*OPT?</b>                                                             | <b>*OPT?</b>             | Identifies oscilloscope options.                                   |
| <b>ROUT</b>                                                              | <b>REAR_OUTPUT</b>       | Sets the type of signal put out at the rear BNC connector.         |
| <b>TDISP</b>                                                             | <b>TIME_DISPLAY</b>      | Changes time display from current to trigger or none.              |
| <b>*TST?</b>                                                             | <b>*TST?</b>             | Performs an internal self-test.                                    |
| <b>PROBES — Using Probes</b>                                             |                          |                                                                    |
| <b>PRCA?</b>                                                             | <b>PROBE_CAL?</b>        | Performs a complete calibration of the connected current probe.    |
| <b>PRDG</b>                                                              | <b>PROBE_DEGAUSS?</b>    | Degausses and calibrates the connected current probe.              |
| <b>PRIT?</b>                                                             | <b>PROBE_INFOTEXT</b>    | Returns the connected probe's informative text.                    |
| <b>PRNA</b>                                                              | <b>PROBE_NAME?</b>       | Gives an identification of the probe connected to the instrument.  |
| <b>SAVE/RECALL SETUP — Preserving and Restoring Front-Panel Settings</b> |                          |                                                                    |
| <b>PNSU</b>                                                              | <b>PANEL_SETUP</b>       | Complements the *SAV/*RST commands.                                |
| <b>*RCL</b>                                                              | <b>*RCL</b>              | Recalls one of five non-volatile panel setups.                     |
| <b>RCPN</b>                                                              | <b>RECALL_PANEL</b>      | Recalls a front-panel setup from mass storage.                     |
| <b>*RST</b>                                                              | <b>*RST</b>              | Initiates a device reset.                                          |
| <b>*SAV</b>                                                              | <b>*SAV</b>              | Stores the current state in non-volatile internal memory.          |
| <b>STPN</b>                                                              | <b>STORE_PANEL</b>       | Stores the complete front-panel setup on a mass-storage file.      |

## 9300 &amp; LC Series

| <b>Short</b>                                                                 | <b>Long Form</b>      | <b>What the Command/Query Does</b>                                              |
|------------------------------------------------------------------------------|-----------------------|---------------------------------------------------------------------------------|
| <b>STATUS — Obtaining Status Information and Setting Up Service Requests</b> |                       |                                                                                 |
| <b>ALST?</b>                                                                 | <b>ALL_STATUS?</b>    | Reads, clears contents of all (but one) of the status registers.                |
| <b>*CLS</b>                                                                  | <b>*CLS</b>           | Clears all the status data registers.                                           |
| <b>CMR?</b>                                                                  | <b>CMR?</b>           | Reads and clears contents of CoMmand error Register (CMR).                      |
| <b>DDR?</b>                                                                  | <b>DDR?</b>           | Reads and clears the Device-Dependent error Register (DDR).                     |
| <b>*ESE</b>                                                                  | <b>*ESE</b>           | Sets the standard Event Status Enable (ESE) register.                           |
| <b>*ESR?</b>                                                                 | <b>*ESR?</b>          | Reads and clears the Event Status Register (ESR).                               |
| <b>EXR?</b>                                                                  | <b>EXR?</b>           | Reads and clears the EXecution error Register (EXR).                            |
| <b>INE</b>                                                                   | <b>INE</b>            | Sets the INternal state change Enable register (INE).                           |
| <b>INR?</b>                                                                  | <b>INR?</b>           | Reads and clears the INternal state change Register (INR).                      |
| <b>IST?</b>                                                                  | <b>IST?</b>           | Individual STatus reads the current state of IEEE 488.                          |
| <b>*OPC</b>                                                                  | <b>*OPC</b>           | Sets to true the OPC bit (0) in the Event Status Register (ESR).                |
| <b>*PRE</b>                                                                  | <b>*PRE</b>           | Sets the PaRallel poll Enable register (PRE).                                   |
| <b>*SRE</b>                                                                  | <b>*SRE</b>           | Sets the Service Request Enable register (SRE).                                 |
| <b>*STB?</b>                                                                 | <b>*STB?</b>          | Reads the contents of IEEE 488.                                                 |
| <b>URR?</b>                                                                  | <b>URR?</b>           | Reads and clears the User Request status Register (URR).                        |
| <b>*WAI</b>                                                                  | <b>*WAI</b>           | WAI to continue — required by IEEE 488.                                         |
| <b>WAVEFORM TRANSFER — Preserving and Restoring Waveforms</b>                |                       |                                                                                 |
| <b>INSP?</b>                                                                 | <b>INSPECT?</b>       | Allows acquired waveform parts to be read.                                      |
| <b>REC</b>                                                                   | <b>RECALL</b>         | Recalls waveform file from mass storage to internal memories.                   |
| <b>STO</b>                                                                   | <b>STORE</b>          | Stores a trace in one of the internal memories or mass storage.                 |
| <b>STST</b>                                                                  | <b>STORE_SETUP</b>    | Controls the way in which traces are stored.                                    |
| <b>STTM</b>                                                                  | <b>STORE_TEMPLATE</b> | Stores the waveform template in a mass-storage device.                          |
| <b>TMPL?</b>                                                                 | <b>TEMPLATE?</b>      | Produces a copy of the template describing a complete waveform.                 |
| <b>TRFL</b>                                                                  | <b>TRANSFER_FILE</b>  | Transfers ASCII files to and from storage media, or between scope and computer. |
| <b>WF</b>                                                                    | <b>WAVEFORM</b>       | Transfers a waveform from the controller to the oscilloscope.                   |
| <b>WFSU</b>                                                                  | <b>WAVEFORM_SETUP</b> | Specifies amount of waveform data for transmission to controller.               |
| <b>WFTX</b>                                                                  | <b>WAVEFORM_TEXT</b>  | Documents waveform acquisition conditions.                                      |

| STATUS                  | ALL_STATUS?, ALST?<br>Query                                                                                                                                                                                                                                                                                                                                                  |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>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.</p> <p>The ALL_STATUS? query is useful in a complete overview of the state of the instrument.</p> |
| <b>QUERY SYNTAX</b>     | <b>ALL_Status?</b>                                                                                                                                                                                                                                                                                                                                                           |
| <b>RESPONSE FORMAT</b>  | <p><b>ALL_Status STB, &lt;value&gt;, ESR, &lt;value&gt;, INR, &lt;value&gt;, DDR, &lt;value&gt;, CMR, &lt;value&gt;, EXR, &lt;value&gt;, URR, &lt;value&gt;</b></p> <p>&lt;value&gt; : = 0 to 65535</p>                                                                                                                                                                      |
| <b>EXAMPLE (GPIB)</b>   | <p>The following instruction reads the contents of all the status registers:</p> <p><b>CMD\$="ALST?": CALL IBWRT(SCOPE%, CMD\$):</b><br/> <b>CALL IBRD(SCOPE%, RSP\$): PRINT RSP\$</b></p> <p>Response message:</p> <p><b>ALST TB, 000000, ESR, 000052, INR, 000005, DDR, 000000, CMR, 000004, EXR, 000024, URR, 000000</b></p>                                              |
| <b>RELATED COMMANDS</b> | *CLS, CMR?, DDR?, *ESR?, EXR?, *STB?, URR?                                                                                                                                                                                                                                                                                                                                   |

**ACQUISITION****ARM\_ACQUISITION, ARM  
Command**

|                         |                                                                                                                                                 |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The ARM_ACQUISITION command enables the signal acquisition process by changing the acquisition state (trigger mode) from "stopped" to "single". |
| <b>COMMAND SYNTAX</b>   | <b>ARM_acquisition</b>                                                                                                                          |
| <b>EXAMPLE</b>          | The following command enables signal acquisition:<br><b>CMD\$="ARM": CALL IBWRT(SCOPE%,CMD\$)</b>                                               |
| <b>RELATED COMMANDS</b> | STOP, *TRG, TRIG_MODE, WAIT                                                                                                                     |

## 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 may be specified.

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

### COMMAND SYNTAX

**<channel> : ATTeNuation <attenuation>**

**<channel> : = {C1, C2, C3<sup>G</sup>, C4<sup>G</sup>, EX, EX10<sup>G</sup>, EX5<sup>G</sup>}**

**<attenuation> : = {1, 2, 5, 10, 20, 25, 50, 100, 200, 500, 1000, 10000}**

### QUERY SYNTAX

**<channel> : ATTeNuation?**

### RESPONSE FORMAT

**<channel> : ATTeNuation <attenuation>**

## G AVAILABILITY

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

**<channel> : {EX10}** available on all models except those in the LC564 and LC584 Series.

**<channel> : {EX5}** available only on LC564 and LC584 Series models.

### EXAMPLE (GPIB)

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

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

**MISCELLANEOUS****AUTO\_CALIBRATE, ACAL**  
**Command/Query****DESCRIPTION**

The AUTO\_CALIBRATE command is used to enable or disable the automatic calibration of the instrument. At power-up, auto-calibration is turned ON, i.e. all input channels are periodically calibrated for the current input amplifier and timebase settings.

The automatic calibration may be disabled by issuing the command ACAL OFF. Whenever it is 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.

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

**COMMAND SYNTAX**

**Auto\_CALibrate** <state>  
<state> : = {**ON**, **OFF**}

**QUERY SYNTAX**

**Auto\_CALibrate?**

**RESPONSE FORMAT**

**Auto\_CALibrate** <state>

**EXAMPLE (GPIB)**

The following instruction disables auto-calibration:

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

**RELATED COMMANDS**

\*CAL?

**DISPLAY**

**AUTO\_SCROLL, ASCR**  
Command/Query

|                         |                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The AUTO_SCROLL command and query controls the Auto Scroll feature, accessed through the front-panel using the "MATH SETUP" button and "ZOOM + MATH" menus. This automatically moves the selected trace (or all traces if multi-zoom is on) across the screen. The command turns the scroll on and off and sets the scrolling speed in divisions per second, and the query returns the current scroll rate. |
| <b>COMMAND SYNTAX</b>   | <b>Auto_SCROLL</b> <rate><br><br><rate> : = scroll rate in divisions per second in the range - 10.0 to 10.00. A positive value causes it to scroll to the right while a negative value scrolls to the left. 0 will stop the scrolling.                                                                                                                                                                      |
| <b>QUERY SYNTAX</b>     | <b>Auto_SCROLL?</b>                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>RESPONSE FORMAT</b>  | <b>Auto_SCROLL</b> <rate>                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>EXAMPLE (GPIB)</b>   | The following instruction activates Auto Scroll and start scrolling the data to the right at a rate at 2 s/div.:<br><br><b>CMD\$="ASCR 2": CALL IBWRT(SCOPE%,CMD\$)</b>                                                                                                                                                                                                                                     |
| <b>RELATED COMMANDS</b> | MULTI_ZOOM, HOR_MAGNIFY, HOR_POSITION                                                                                                                                                                                                                                                                                                                                                                       |

**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, C3G,C4G**}

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.

**G AVAILABILITY**

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

**EXAMPLE**

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

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



## ACQUISITION

## BANDWIDTH\_LIMIT, BWLG Command/Query

### DESCRIPTION

BANDWIDTH\_LIMIT enables or disables the bandwidth-limiting low-pass filter. When Global\_BWL (see page 86) is on the BWL command applies to all channels; when off, 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>

Or, alternatively, to choose the bandwidth limit of an individual channel or channels when Global\_BWL is *off*:

**BandWidth\_Limit** <channel>, <mode>[, <channel>, <mode>[, <channel>, <mode>[, <channel>, <mode>]]]

<mode> : = {OFF, ON, 200MHZG}

<channel> : = {C1, C2, C3G, C4G}

### QUERY SYNTAX

**BandWidth\_Limit?**

### RESPONSE FORMAT

When Global\_BWL is *on*, or if Global\_BWL is *off* and all four channels have the same bandwidth limit, the response is:

**BandWidth\_Limit** <mode>

Or, alternatively, 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>]]]

## G AVAILABILITY

Not available on some models. And the 200 MHz setting is available only on certain models.

{C3, C4} : Available only on four-channel models.

## 9300 &amp; LC Series

**EXAMPLE**

The following turns on the bandwidth filter for all channels, when Global\_BWL is *on* (as it is by default):

**CMD\$="BWL ON": CALL IBWRT(SCOPE%,CMD\$)**

The following turns the bandwidth filter on for Channel 1 only (the first instruction turns *off* Global\_BWL):

**CMD\$="GBWL OFF": CALL IBWRT(SCOPE%,CMD\$)**

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

**RELATED COMMANDS**

GLOBAL\_BWL

**MISCELLANEOUS**

**BUZZER, BUZZ**  
Command

**DESCRIPTION**

The BUZZER command controls the built-in piezo-electric buzzer. This is useful for attracting the attention of a local operator in an interactive working application. The buzzer can either be activated for short beeps (about 400 ms long in BEEP mode) or continuously for a certain time interval selected by the user by turning the buzzer ON or OFF. This command is only usable in oscilloscopes fitted with the CLBZ hard option.

**COMMAND SYNTAX**

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

**EXAMPLE (GPIB)**

Sending the following code will cause the oscilloscope to sound two short tones.

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

**MISCELLANEOUS****\*CAL?**  
Query**DESCRIPTION**

The \*CAL? query cause the oscilloscope to perform an internal self-calibration and generates a response that indicates whether or not the instrument 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 instrument returns to the state it was in just prior to the calibration cycle.

**QUERY SYNTAX****\*CAL?****RESPONSE FORMAT**

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

**EXAMPLE (GPIB)**

The following instruction 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

|                         |                                                                                                                                                                                                                                                                                                                             |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The CAL_OUTPUT command is used to set the type of signal put out at the CAL BNC connector.                                                                                                                                                                                                                                  |
| <b>COMMAND SYNTAX</b>   | <b>Cal_OUTput</b> <mode>[, <level>[, <rate>]]<br><br><b>Cal_OUTput PULSE</b> [, <width>]<br><br><mode> : = {OFF, CALSQ, CALPU, PF, TRIG, LEVEL <sup>G</sup> , PULSE <sup>G</sup> , TRDY}<br><level> : = 0.0 to 1.00 V into 1 MΩ<br><rate> : = 500 to 2 000 000 Hz<br><br><width> : = 10 μs to 10 s (applies only to PULSE). |
| <b>QUERY SYNTAX</b>     | <b>Cal_OUTput?</b>                                                                                                                                                                                                                                                                                                          |
| <b>RESPONSE FORMAT</b>  | <b>Cal_OUTput</b> <mode>,<level>[,<rate>]                                                                                                                                                                                                                                                                                   |
| <b>G AVAILABILITY</b>   | <mode> : PULSE or LEVEL will only be accepted if the Cal_OUTput? mode was previously OFF.                                                                                                                                                                                                                                   |
| <b>EXAMPLE (GPIB)</b>   | The following command sets the calibration signal to give a 0–0.2 volt pulse of 25 ns width at a 10 kHz rate:<br><br><b>CMD\$="COUT CALPU,0.2 V,10 kHz":</b><br><b>CALL IBWRT(SCOPE%,CMD\$)</b>                                                                                                                             |
| <b>RELATED COMMANDS</b> | PASS_FAIL_DO                                                                                                                                                                                                                                                                                                                |

## 9300 &amp; LC Series

## ADDITIONAL INFORMATION

| Notation     |                                             |
|--------------|---------------------------------------------|
| <b>CALSQ</b> | Provides a square signal                    |
| <b>CALPU</b> | Provides a pulse signal                     |
| <b>PF</b>    | PASS/FAIL mode                              |
| <b>TRIG</b>  | Trigger Out mode                            |
| <b>LEVEL</b> | Provides a DC signal at the requested level |
| <b>OFF</b>   | Provides no signal (ground level)           |
| <b>PULSE</b> | Provides a single pulse                     |
| <b>TRDY</b>  | Trigger is ready for a new acquisition      |

**DISPLAY**

**CALL\_HOST, CHST**  
Command/Query

**DESCRIPTION**

The CALL\_HOST command allows the user to manually generate a service request (SRQ). Once the CALL\_HOST command has been received, the message "Call Host" will be displayed next to the lowest button on the menu-button column immediately next to the screen. Pressing this button while in the root menu sets the User Request status Register (URR) and the URQ bit of the Event Status Register. This can generate a SRQ in local mode, provided the service request mechanism has been enabled.

The response to the CALL\_HOST? query indicates whether CALL HOST is enabled (on) or disabled (off).

**COMMAND SYNTAX**

**Call\_HoST <state>**

<state> : = {ON, OFF}

**QUERY SYNTAX**

**Call\_HoST?**

**RESPONSE FORMAT**

**Call\_HoST <state>**

**EXAMPLE (GPIB)**

After executing the following code an SRQ request will be generated whenever the button is pressed (it is assumed that SRQ servicing has already been enabled):

**CMD\$="CHST ON": CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS**

URR

## 9300 &amp; LC Series

| <i>FUNCTION</i>         | <b>CLEAR_MEMORY, CLM</b><br>Command                                                                                                                          |
|-------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | 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. |
| <b>COMMAND SYNTAX</b>   | <b>CLear_Memory</b> < memory><br><memory> : = { <b>M1, M2, M3, M4</b> }                                                                                      |
| <b>EXAMPLE (GPIB)</b>   | The following command clears the memory M2.<br><b>CMD\$="CLM M2": CALL IBWRT(SCOPE%, CMD\$)</b>                                                              |
| <b>RELATED COMMANDS</b> | STORE                                                                                                                                                        |



| <i>FUNCTION</i>         | <b>CLEAR_SWEEPS, CLSW</b><br>Command                                                                                                                                                                         |
|-------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | 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. |
| <b>COMMAND SYNTAX</b>   | <b>CLear Sweeps</b>                                                                                                                                                                                          |
| <b>EXAMPLE (GPIB)</b>   | The following example will restart the cumulative processing:<br><b>CMD\$="CLSW": CALL IBWRT(SCOPE%,CMD\$)</b>                                                                                               |
| <b>RELATED COMMANDS</b> | DEFINE, INR                                                                                                                                                                                                  |

## 9300 &amp; LC Series

**STATUS****\*CLS**  
Command

|                         |                                                                                                                            |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The *CLS command clears all the status data registers.                                                                     |
| <b>COMMAND SYNTAX</b>   | <b>*CLS</b>                                                                                                                |
| <b>EXAMPLE (GPIB)</b>   | The following command causes all the status data registers to be cleared:<br><b>CMD\$="*CLS": CALL IBWRT(SCOPE%,CMD\$)</b> |
| <b>RELATED COMMANDS</b> | ALL_STATUS, CMR, DDR, *ESR, EXR, *STB, URR                                                                                 |

| STATUS           | CMR?<br>Query                                                                                                                                                                                                     |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DESCRIPTION      | The CMR? query reads and clears the contents of the CoMmand error Register (CMR) — <i>see table next page</i> — which specifies the last syntax error type detected by the instrument.                            |
| QUERY SYNTAX     | CMR?                                                                                                                                                                                                              |
| RESPONSE FORMAT  | CMR <value><br><value> : = 0 to 13                                                                                                                                                                                |
| EXAMPLE (GPIB)   | The following instruction reads the contents of the CMR register:<br><br><b>CMD\$="CMR?": CALL IBWRT(SCOPE%, CMD\$):</b><br><b>CALL IBRD(SCOPE%, RSP\$): PRINT RSP\$</b><br><br>Response message:<br><b>CMR 0</b> |
| RELATED COMMANDS | ALL_STATUS?, *CLS                                                                                                                                                                                                 |

## 9300 &amp; LC Series

## ADDITIONAL INFORMATION

| Command Error Status Register Structure (CMR) |                                                                 |
|-----------------------------------------------|-----------------------------------------------------------------|
| Value                                         | Description                                                     |
| 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 |

## DISPLAY

## COLOR, COLRG Command/Query

### DESCRIPTION

The COLOR command is used to select the color of an individual display object such as text, trace, grid or cursor.

The response to the COLOR? query indicates the color assigned to each display object, whether or not it is currently displayed.

*Note: This command is only effective if the color scheme (CSCH) is chosen from U1...U4.*

### COMMAND SYNTAX

**COLOR** <object, color>[,...<object>,<color>]

<object> := {BACKGND, C1, C2, C3, C4, TA, TB, TC, TD, GRID, TEXT, CURSOR, NEUTRAL, WARNING},

<color> := { WHITE, CYAN, YELLOW, GREEN, MAGENTA, BLUE, RED, LTGRAY, GRAY, SLGRAY, CHGRAY, DKCYAN, CREAM, SAND, AMBER, OLIVE, LTGREEN, JADE, LMGREEN, APGREEN, EMGREEN, GRGREEN, OCSPRAY, ICEBLUE, PASTBLUE, PALEBLUE, SKYBLUE, ROYBLUE, DEEPBLUE, NAVY, PLUM, PURPLE, AMETHYST, FUCHSIA, RASPBRY, NEONPINK, PALEPINK, PINK, VERMIL, ORANGE, CERISE, KHAKI, BROWN, BLACK}

### QUERY SYNTAX

**COLOR?**

### RESPONSE FORMAT

**COLOR** <object>, <color>[, ...<object>, <color>]

## G AVAILABILITY

Available on color instruments only.

### EXAMPLE (GPIB)

The following instruction selects color scheme U1, and then red as the color of Channel 1:

```
CMD$="CSCH U1": CALL IBWRT(SCOPE%,CMD$)
CMD$="COLR C1,RED": CALL IBWRT(SCOPE%,CMD$)
```

### RELATED COMMANDS

COLOR\_SCHEME, PERSIST\_COLOR

## 9300 &amp; LC Series

## ADDITIONAL INFORMATION

| Notation         |                 |                 |                                        |
|------------------|-----------------|-----------------|----------------------------------------|
| <color>          | Color           | <color>         | Color                                  |
| <b>WHITE</b>     | White           | <b>OCSPRAY</b>  | Ocean Spray                            |
| <b>CYAN</b>      | Cyan            | <b>ICEBLUE</b>  | Ice Blue                               |
| <b>YELLOW</b>    | Yellow          | <b>PASTBLUE</b> | Pastel Blue                            |
| <b>GREEN</b>     | Green           | <b>PALEBLUE</b> | Pale Blue                              |
| <b>MAGENTA</b>   | Magenta         | <b>SKYBLUE</b>  | Sky Blue                               |
| <b>BLUE</b>      | Blue            | <b>ROYLBLUE</b> | Royal Blue                             |
| <b>RED</b>       | Red             | <b>DEEPBLUE</b> | Deep Blue                              |
| <b>LTGRAY</b>    | Light Gray      | <b>NAVY</b>     | Navy                                   |
| <b>GRAY</b>      | Gray            | <b>PLUM</b>     | Plum                                   |
| <b>SLGRAY</b>    | Slate Gray      | <b>PURPLE</b>   | Purple                                 |
| <b>CHGRAY</b>    | Charcoal Gray   | <b>AMETHYST</b> | Amethyst                               |
| <b>DKCYAN</b>    | Dark Cyan       | <b>FUCHSIA</b>  | Fuchsia                                |
| <b>CREAM</b>     | Cream           | <b>RASPB</b>    | Raspberry                              |
| <b>SAND</b>      | Sand            | <b>NEONPINK</b> | Neon Pink                              |
| <b>AMBER</b>     | Amber           | <b>PALEPINK</b> | Pale Pink                              |
| <b>OLIVE</b>     | Olive           | <b>PINK</b>     | Pink                                   |
| <b>LTGREEN</b>   | Light Green     | <b>VERMIL</b>   | Vermilion                              |
| <b>JADE</b>      | Jade            | <b>ORANGE</b>   | Orange                                 |
| <b>LMGREEN</b>   | Lime Green      | <b>CERISE</b>   | Cerise                                 |
| <b>APGREEN</b>   | Apple Green     | <b>KHAKI</b>    | Khaki                                  |
| <b>EMGREEN</b>   | Emerald Green   | <b>BROWN</b>    | Brown                                  |
| <b>GRGREEN</b>   | Grass Green     | <b>BLACK</b>    | Black                                  |
| <object>         | Display Object  | <object>        | Display Object                         |
| <b>BACKGND</b>   | Background      | <b>CURSOR</b>   | cursors                                |
| <b>C1 . . C4</b> | Channel Traces  | <b>WARNING</b>  | Warning Messages                       |
| <b>TA . . TD</b> | Function Traces | <b>NEUTRAL</b>  | Neutral color                          |
| <b>GRID</b>      | Grid lines      | <b>OVERLAYS</b> | Menu background color<br>(Full Screen) |

**DISPLAY**

**COLOR\_SCHEME, CSCHG**  
Command/Query

**DESCRIPTION**

The COLOR\_SCHEME command is used to select the color scheme for the display.

The response to the COLOR\_SCHEME? query indicates the color scheme in use.

**COMMAND SYNTAX**

**Color\_SCHeMe** <scheme>

<scheme > : = {**1, 2, 3, 4, 5, 6, 7, U1, U2, U3, U4**}

**QUERY SYNTAX**

**Color\_SCHeMe?**

**RESPONSE FORMAT**

**Color\_SCHeMe** <scheme>

**G AVAILABILITY**

This command and query is available only on color instruments.

**EXAMPLE (GPIB)**

The following instruction selects the user color scheme U2:

**CMD\$="CSCH U2": CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS**

COLOR, PERSIST\_COLOR

**ACQUISITION****COMBINE\_CHANNELS, COMBG**  
Command/Query

**DESCRIPTION** On most models whose channels can be combined, the COMBINE\_CHANNELS command controls the channel interleaving function of the acquisition system (see *the Operator's Manual for an explanation of which models combine channels and how*). The COMBINE\_CHANNELS? query returns the channel interleaving function's current status.

**COMMAND SYNTAX** **COMBine\_channels** <state>  
 <state> : = {1G, 2, 4G, AUTOG}

**QUERY SYNTAX** **COMBine\_channels?**

**RESPONSE FORMAT** **COMB** <state>

**G AVAILABILITY** <state>1 and AUTO are only available on LC584 Series instruments.  
 <state>4 can be used on the four-channel oscilloscopes of the 9344C, LC374 and LC584 Series, as well as on models in the 9354C, 9374C, 9384C, LC334, LC534 and LC574 Series using the supplied adapter.

**EXAMPLE (GPIB)** The following instruction engages channel interleaving between Channels 1 and 2, and Channels 3 and 4 on four-channel instruments:

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

Only on LC584 Series models, 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.

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

### COMMAND SYNTAX

**Comm\_ForMaT** <block\_format>, <data\_type>, <encoding>

<block\_format> : = {**DEF9**, **IND0**, **OFF**}

<data\_type> : = {**BYTE**, **WORD**}

<encoding> : = {**BIN**, **HEX**}

(GPIB uses both encoding forms, RS-232-C always uses HEX)

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

For GPIB: **DEF9**, **WORD**, **BIN**

For RS-232-C: **DEF9**, **WORD**, **HEX**

### QUERY SYNTAX

**Comm\_ForMaT?**

### RESPONSE FORMAT

**Comm\_ForMaT** <block\_format>, <data\_type>, <encoding>

### EXAMPLE (GPIB)

The following code redefines the transmission format of waveform data. The data will be transmitted as a block of indefinite length. Data will be coded in binary and represented as 8-bit integers.

```
CMD$="CFMT IND0,BYTE,BIN": CALL IBWRT(SCOPE%,CMD$)
```

### 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>
```

## 9300 &amp; LC Series

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

**IND0:** Uses the IEEE 488.2 indefinite length arbitrary block response data format.

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>

**OFF:** Same as IND0. In addition, the data block type identifier and the leading #0 of the indefinite length block will be suppressed. The data presented above would be sent as:

**WF** <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 (GPIB only)

**HEX:** Hexadecimal encoding (bytes are converted to 2 hexadecimal ASCII digits (0, ...9, A, ...F))

#### RELATED COMMANDS

#### WAVEFORM

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

The COMM\_HEADER command controls the way the oscilloscope formats responses to queries. The instrument provides three response formats: LONG format, in which responses start with the long form of the header word; SHORT format, 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. Until the user requests otherwise, the SHORT response format is used.

This command does not affect the interpretation of messages sent to the oscilloscope. Headers may 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**}

*Note: The default mode, i.e. the mode just after power-on, is SHORT.*

**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 Assistant (*see oscilloscope Operator's Manual*), which assists in debugging remote control programs. Selected when using the instrument's front-panel via the "UTILITIES" and "SPECIAL MODES" menus, Remote Control Assistant can log all message transactions occurring between the external controller and the oscilloscope. The log may be viewed at any time in the provided menu on the screen and has four levels to choose from:

- 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.
- RS** Log the Full Dialog and send it to a recording device connected to the RS232 port.

**COMMAND SYNTAX**

**Comm\_HeLP** <level>

<level> : = { **OFF**, **EO**, **FD**, **RS**, }

The default level (i.e. the level just after power-on) is EO.

**QUERY SYNTAX**

**Comm\_HeLP?**

**RESPONSE FORMAT**

**Comm\_HeLP** <level>

**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?<br>Query                                                                                                                                                                                                                                     |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The COMM_HELP_LOG query returns the current contents of the log generated by the Remote Control Assistant (see <i>CHLP description</i> ). If the optional parameter CLR is specified, the log will be cleared after the transmission. Otherwise, it will be kept. |
| <b>QUERY SYNTAX</b>     | <b>Comm_HeLP_Log? [CLR]</b>                                                                                                                                                                                                                                       |
| <b>RESPONSE FORMAT</b>  | <b>Comm_HeLP_Log</b> <string containing the logged text>                                                                                                                                                                                                          |
| <b>EXAMPLE (GPIB)</b>   | The following code reads the remote control log and prints it:<br><b>CMD\$="CHL?": CALL IBWRT(SCOPE%,CMD\$)PRINT</b>                                                                                                                                              |
| <b>RELATED COMMANDS</b> | COMM_HELP                                                                                                                                                                                                                                                         |

**COMMUNICATION****COMM\_ORDER, CORD**  
Command/Query**DESCRIPTION**

The COMM\_ORDER command controls the byte order of waveform data transfers. Waveform data may be sent with the most significant byte (MSB) or the least significant byte (LSB) in the first position. The default mode is 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 selected, the most significant byte is sent first. When "COMM\_ORDER LO" is specified, the least significant byte is sent first.

The COMM\_ORDER? query returns the byte transmission order currently in 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                     |
|------------|-----------------------------|-----------------------------|
| Word       | <MSB><LSB>                  | <LSB><MSB>                  |
| Long/Float | <MSB><byte2><byte3><LSB>    | <LSB><byte3><byte2><MSB>    |
| Double     | <MSB><byte2>...<byte7><LSB> | <LSB><byte7>...<byte2><MSB> |

**RELATED COMMANDS**

WAVEFORM

**COMMUNICATION**

**COMM\_RS232, CORS**  
Command/Query

**DESCRIPTION**

The command COMM\_RS232 sets the parameters of the RS-232-C port for remote control.

The COMM\_RS232? query reports the settings of the parameters.

*Note: This command is ONLY valid if the oscilloscope is being remotely controlled via the RS-232-C port.*

The parameters are:

- (a) **End Input character:** When received by the oscilloscope, this character is interpreted as the END-of-a-command message marker. The commands received will be parsed and executed.
- (b) **End Output string:** The oscilloscope adds this string at the end of a response message. When the host computer receives this string, it knows that the oscilloscope has completed its response.
- (c) **Line Length:** This parameter defines the maximum number of characters sent to the host in a single line. Remaining characters of the response are output in separate additional lines. This parameter is only applicable if a line separator has been selected.
- (d) **Line Separator:** This parameter is used to select the line-splitting mechanism and to define the characters used to split the oscilloscope response messages into many lines. Possible line separators are: CR, LF, CRLF. <CR>, <LF> or <CR> followed by <LF>. These are sent to the host computer after <line\_length> characters.
- (e) **SRQ string:** This string is sent each time the oscilloscope signals an SRQ to the host computer.

Some parameters of this command require ASCII strings as actual arguments. In order to facilitate the embedding of non-printable characters into such strings, escape sequences may be used. The back-slash character ('\') is used as an escape character. The following escape sequences are recognized:

## 9300 &amp; LC Series

"\a": Bell character  
 "\b": Back space character  
 "\e": Escape character  
 "\n": Line feed character  
 "\r": Carriage return character  
 "\t": Horizontal tab character  
 "\\\": The back-slash character itself  
 "\ddd": ddd represents one to three decimal digit characters giving the code value of the corresponding ASCII character. This allows any ASCII code in the range 1 to 127 to be inserted.

Before using the string, the oscilloscope will replace the escape sequence by the corresponding ASCII character.

For example, the escape sequences "\r", "\13" and "\013" are all replaced by the single ASCII character <Carriage Return>.

| Notation   |                     |
|------------|---------------------|
| <b>EI</b>  | End input character |
| <b>E0</b>  | End output string   |
| <b>LL</b>  | Line length         |
| <b>LS</b>  | Line separator      |
| <b>SRQ</b> | SRQ service request |

## COMMAND SYNTAX

**C0mm\_RS232 EI**, <ei\_char>, **E0**, '<eo\_string>', **LL**, <line\_length>, **LS**, <Line\_sep>, **SRQ**, '<srq\_string>'

<ei\_char> : = 1 to 126 (default: 13 = Carriage Return)

<eo\_string> : = A non-empty ASCII string of up to 20 characters (default: "\n\r")

<line\_length> : = 40 to 1024 (default: 256)

<line\_sep> : = {**OFF**, **CR**, **LF**, **CRLF**} (default: **OFF**)

<srq\_string> : = An ASCII string of up to 20 characters which may be empty (default: empty string)



**QUERY SYNTAX**

**C0mm\_RS232?**

**RESPONSE FORMAT**

**C0mm\_RS232 EI, <ei\_char>, E0, "<eo\_string>", LL, <line\_length>, LS, <line\_sep>, SRQ, "<srq\_string>"**

**EXAMPLE**

After executing the command:

**COMM\_RS232 EI, 3, E0, "\r\nEND\r\n"**

the instrument will assume that it has received a complete message each time the <ETX> (decimal value 3) is detected. Response messages will be terminated by sending the character sequence "<CR><LF>END<CR><LF>".

**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**, **C3G**, **C4G**, **EX**, **EX10G**, **EX5G**}

                                         <coupling> : = {**A1MG**, **D1MG**, **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 instrument will disconnect the input.

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

                                         <channel> : = {**EX10**} available on all models except those in the LC564 and LC584 Series.

                                         <channel> : = {**EX5**} available on LC564 and LC584 Series oscilloscopes only.

                                         A1M and D1M are not available on model 9362C.

**EXAMPLE GPIB)**                    The following command 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.

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

| Notation     |                                      |
|--------------|--------------------------------------|
| <b>ABS</b>   | absolute reading of relative cursors |
| <b>CUST</b>  | custom parameters                    |
| <b>FAIL</b>  | pass/fail: fail                      |
| <b>HABS</b>  | horizontal absolute cursors          |
| <b>HPAR</b>  | standard time parameters             |
| <b>HREL</b>  | horizontal relative cursors          |
| <b>OFF</b>   | cursors and parameters off           |
| <b>PARAM</b> | synonym for VPAR                     |
| <b>PASS</b>  | pass/fail: pass                      |
| <b>SHOW</b>  | custom parameters (old form)         |
| <b>STAT</b>  | parameter statistics                 |
| <b>VABS</b>  | vertical absolute cursors            |
| <b>VPAR</b>  | standard voltage parameters          |
| <b>VREL</b>  | vertical relative cursors            |

*Note: The PARAM mode is turned OFF when the XY mode is ON.*

**COMMAND SYNTAX**

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

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

<submode> : = {**STAT**, **ABS**}

## 9300 &amp; LC Series

*Note 1: 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.*

*Note 2: The keyword ABS is optional with mode HREL. If it is present, ABS chooses absolute amplitude reading of relative cursors. Absence of ABS selects relative amplitude reading of relative cursors.*

**QUERY SYNTAX****CuRsor\_MeaSure?****RESPONSE FORMAT****CuRsor\_MeaSure** <mode>**EXAMPLE (GPIB)**

The following command switches on the vertical relative cursors:

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

The following command 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, PASS\_FAIL\_CLEAR,  
PASS\_FAIL\_CONDITION, PASS\_FAIL\_DELETE,  
PASS\_FAIL\_MASK,

**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 the following four forms:

**CURSOR\_MEASURE HABS**

**CURSOR\_MEASURE HREL**

**CURSOR\_MEASURE VABS**

**CURSOR\_MEASURE VREL**

To turn on parameter measurements without statistics, use one of the following three forms:

**CURSOR\_MEASURE CUST**  
**CURSOR\_MEASURE HPAR**  
**CURSOR\_MEASURE VPAR**

To turn on parameter statistics, add the keyword **STAT** to the above three forms.

To turn on Pass or Fail tests on parameter or mask tests, use:

**CURSOR\_MEASURE PASS**  
**CURSOR\_MEASURE FAIL**

Use the command:

**PASS\_FAIL\_CONDITION**

to select parameters in the Custom mode, and to modify the test conditions in the Pass/Fail mode.

## 9300 &amp; LC Series

**CURSOR****CURSOR\_READOUT, CRRDG**  
Command/Query

**DESCRIPTION** This command sets the readout of the time cursor amplitude in volts or dBm.

**COMMAND SYNTAX** **CuRsor\_ReaDout** <scale>  
<scale> : = {VOLTS,DBM}

**QUERY SYNTAX** **CRRD?**

**RESPONSE FORMAT** **CURSOR\_READOUT VOLTS**

**G AVAILABILITY** LC scopes only.

**EXAMPLE** The following command sets the amplitude of the time cursors to read out in dBm:

CMD\$="CRRD DBM"

CALL IBWRT (SCOPE%,CMD\$)

**RELATED COMMANDS** CURSOR\_MEASURE

# CURSOR

## CURSOR\_SET, CRST Command/Query

### DESCRIPTION

The CURSOR\_SET command allows the user to position any one of the eight independent cursors at a given screen location. The positions of the cursors can be modified or queried even if the required cursor is not currently displayed on the screen.

When setting a cursor position, a trace must be specified, relative to which the cursor will be positioned.

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

*Note: If the parameter display is turned on (or the pass/fail display or the extended parameters display), the parameters of the specified trace will be shown unless the newly chosen trace is not displayed or has been acquired in sequence mode; these conditions will produce an environment error, (see table on page 72). To change only the trace without repositioning the cursors, the CURSOR\_SET command may be given with no argument (for example, TB:CRST).*

| Notation    |                       |             |                     |
|-------------|-----------------------|-------------|---------------------|
| <b>HABS</b> | horizontal absolute   | <b>PREF</b> | parameter reference |
| <b>HDIF</b> | horizontal difference | <b>VABS</b> | vertical absolute   |
| <b>HREF</b> | horizontal reference  | <b>VDIF</b> | vertical difference |
| <b>PDIF</b> | parameter difference  | <b>VREF</b> | vertical reference  |

### COMMAND SYNTAX

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

<trace> : = {TA, TB, TC, TD, C1, C2, C3G,C4G}

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

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

<position> : = -29.5 to 29.5 DIV (vertical)

*Note 1: The suffix DIV is optional.*

## 9300 &amp; LC Series

*Note 2: Parameters are grouped in pairs. The first parameter specifies the cursor to be modified and the second one indicates its new value. Parameters may be grouped in any order and may be restricted to those items to be changed.*

**QUERY SYNTAX**

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

**<cursor> : = {HABS, VABS, HREF, HDIF, VREF, VDIF, PREF, PDIF, 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.

**G AVAILABILITY**

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

**EXAMPLE (GPIB)**

The following command positions the VREF and VDIF cursors at +3 DIV and -7 DIV respectively, using Trace A as a reference:

**CMD\$="TA:CRST VREF,3DIV,VDIF,-7DIV":  
CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS**

CURSOR\_MEASURE, CURSOR VALUE,  
PARAMETER\_VALUE, PER\_CURSOR\_SET,  
XY\_CURSOR\_SET



## 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.)

| Notation    |                     |             |                   |
|-------------|---------------------|-------------|-------------------|
| <b>HABS</b> | horizontal absolute | <b>VABS</b> | vertical absolute |
| <b>HREL</b> | horizontal relative | <b>VREL</b> | vertical relative |

### QUERY SYNTAX

<trace> : **CuRsor\_VAlue?** [<mode>, ...<mode>]

<trace> : = {TA, TB, TC, TD, C1, C2, C3<sup>G</sup>, C4<sup>G</sup>}

<mode> : = {HABS, HREL, VABS, VREL, ALL}

### RESPONSE FORMAT

<trace> : **CuRsor\_VAlue** HABS, <abs\_hori>, <abs\_vert>

<trace> : **CuRsor\_VAlue** HREL, <delta\_hori>, <delta\_vert>, <absvert\_ref>, <absvert\_dif>, <slope>

<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: If <mode> is not specified or equals ALL, all the measured cursor values for the specified trace are returned. If the value of a cursor cannot be determined in the current environment, the value UNDEF will be returned.*

## G 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**

### RELATED COMMANDS

CURSOR\_SET, PARAMETER\_VALUE,  
PER\_CURSOR\_VALUE, XY\_CURSOR\_VALUE

## 9300 &amp; LC Series

**DISPLAY****DATA\_POINTS, DPNTG**  
Command/Query**DESCRIPTION**

The DATA\_POINTS command is used to control whether the waveform sample points are shown as single display pixels or are made bold.

The response to the DATA\_POINTS? query indicates whether the waveform sample points are being displayed as single display pixels or in bold face.

**COMMAND SYNTAX**

**Data\_PoiNTs** <state>  
<state> : = {**NORMAL**, **BOLD**}

**QUERY SYNTAX**

**Data\_PoiNTs?**

**RESPONSE FORMAT**

**Data\_PoiNTs** <state>

**G AVAILABILITY**

Command/Query available on LC Series oscilloscopes only.

**EXAMPLE (GPIB)**

The following instruction highlights the waveform sample points:

**CMD\$="DPNT BOLD": CALL IBWRT(SCOPE%,CMD\$)**

| MISCELLANEOUS          | DATE<br>Command/Query                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>     | <p>The DATE command changes the date/time of the oscilloscope's internal real-time clock.</p> <p>The DATE? query returns the current date/time setting.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>COMMAND SYNTAX</b>  | <p><b>DATE</b> &lt;day&gt;, &lt;month&gt;, &lt;year&gt;, &lt;hour&gt;, &lt;minute&gt;, &lt;second&gt;<br/>           &lt;day&gt; : = 1 to 31<br/>           &lt;month&gt; : = {<b>JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC</b>}<br/>           &lt;year&gt; : = 1990 to 2089<br/>           &lt;hour&gt; : = 0 to 23<br/>           &lt;minute&gt; : = 0 to 59<br/>           &lt;second&gt; : = 0 to 59</p> <p><i>Note: It is not always necessary to specify all the DATE parameters. Only those parameters up to and including the parameter to be changed need be specified in order to change the "year" setting, specify day, month and year together with the required settings. The time settings will remain unchanged. To change the "second" setting, all the DATE parameters must be specified with the required settings.</i></p> |
| <b>QUERY SYNTAX</b>    | <b>DATE?</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>RESPONSE FORMAT</b> | <b>DATE</b> <day>,<month>,<year>,<hour>,<minute>,<second>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>EXAMPLE (GPIB)</b>  | <p>This instruction will change the date to January 1, 1997 and the time to 1:21:16 p.m. (13:21:16 in 24-hour notation):</p> <p><b>CMD\$="DATE 1, JAN, 1997, 13, 21, 16": CALL IBWRT (SCOPE%, CMD\$)</b></p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

## 9300 &amp; LC Series

**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****RESPONSE FORMAT****DDR?**

**DDR** <value>

<value> : = 0 to 65535

<value> : Bit 2, 3, 10, 11 — only on four-channel instruments.

**G AVAILABILITY****EXAMPLE (GPIB)**

The following instruction reads the contents of the DDR register:

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

Response message:

**DDR 0**

**RELATED COMMANDS**

ALL\_STATUS, \*CLS

## 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 functions in the standard oscilloscopes and WPOX processing packages.

### COMMAND SYNTAX

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

*Note 1: 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.*

*Note 2: 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>]

| <param_name>                                              | <value>       | Description                        |
|-----------------------------------------------------------|---------------|------------------------------------|
| EQN                                                       | '<equation>'  | Function equation as defined below |
| DELAY                                                     | <delay>       | Delay by time                      |
| MAXPTS                                                    | <max_points>  | Max. number of points to compute   |
| SWEEPS                                                    | <max_sweeps>  | Maximum number of sweeps           |
| <b>Parameters To Support Additional Functions in WP01</b> |               |                                    |
| BITS                                                      | <bits>        | Number of ERES bits                |
| UNITS                                                     | <units>       | Physical units                     |
| WEIGHT                                                    | <weight>      | Continuous Average weight          |
| <b>Parameters To Support Additional Functions in WP02</b> |               |                                    |
| WINDOW                                                    | <window_type> | FFT window function                |

## 9300 &amp; LC Series

| Parameters To Support Additional Functions in WP03 or DDM                 |                       |                                                   |
|---------------------------------------------------------------------------|-----------------------|---------------------------------------------------|
| MAXBINS                                                                   | <bins>                | Number of bins in histogram                       |
| MAX_EVENTS                                                                | <max_values>          | Max. no. of values in histogram                   |
| CENTER                                                                    | <center>              | Horizontal center position for histogram display. |
| WIDTH                                                                     | <width>               | Width of histogram display                        |
| VERT                                                                      | <vert_scale>          | Vertical scaling type                             |
| Parameters To Support Additional Functions in PRML                        |                       |                                                   |
| LENGTH                                                                    | <length>              | No. points to use from first waveform             |
| START                                                                     | <start>               | Starting point in second waveform                 |
| Function Equations And Names Available On All Models                      |                       |                                                   |
| <source>                                                                  | Identity              |                                                   |
| +<source>                                                                 | Identity              |                                                   |
| - <source>                                                                | Negation              |                                                   |
| <source1> + <source2>                                                     | Addition              |                                                   |
| <source1> - <source2>                                                     | Subtraction           |                                                   |
| <source1><source2>                                                        | Multiplication        |                                                   |
| <source1>/<source2>                                                       | Ratio                 |                                                   |
| AVGS(<source>)                                                            | Average Summed        |                                                   |
| RESAMP(<source>)                                                          | Resample (deskew)     |                                                   |
| SINX(<source>)                                                            | Sin(x)/x interpolator |                                                   |
| ZOOMONLY (<extended_source>)                                              | Zoom only (No Math)   |                                                   |
| Extended Functions Available On Instruments With WP01 Processing Firmware |                       |                                                   |
| ABS(<source>)                                                             | Absolute Value        |                                                   |
| AVGC(<source>)                                                            | Continuous Average    |                                                   |
| DERI(<source>)                                                            | Derivative            |                                                   |
| ERES(<source>)                                                            | Enhanced Resolution   |                                                   |

|                                                                                                                                                                                    |                                         |                       |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------|-----------------------|
| EXP(<source>)                                                                                                                                                                      | Exponential (power of e)                |                       |
| EXP10(<source>)                                                                                                                                                                    | Exponential (power of 10)               |                       |
| EXTR(<source>)                                                                                                                                                                     | Extrema (Roof and Floor)                |                       |
| FLOOR(EXTR(<source>))                                                                                                                                                              | Floor (Extrema source only)             |                       |
| INTG(<source>[ <b>{+, - }</b> <addend>])                                                                                                                                           | Integral                                |                       |
| LN(<source>)                                                                                                                                                                       | Logarithm base e                        |                       |
| LOG10(<source>)                                                                                                                                                                    | Logarithm base 10                       |                       |
| RESC([ <b>{+, - }</b> ][<multiplier>*<source>[ <b>{+, - }</b> <addend>])                                                                                                           | Rescale                                 |                       |
| ROOF(EXTR(<source>))                                                                                                                                                               | Roof (Extrema source only)              |                       |
| 1/<source>                                                                                                                                                                         | Reciprocal                              |                       |
| SQR(<source>)                                                                                                                                                                      | Square                                  |                       |
| SQRT(<source>)                                                                                                                                                                     | Square Root                             |                       |
| <b>FFT Functions Available on Instruments with WP02 Processing Firmware</b><br><i>Note: The source waveform must be a time-domain signal, single segment.</i>                      |                                         |                       |
| FFT(<source>)                                                                                                                                                                      | Fast Fourier Transform (complex result) |                       |
| REAL(FFT(<source>))                                                                                                                                                                | Real part of complex result             |                       |
| IMAG(FFT(<source>))                                                                                                                                                                | Imaginary part of complex result        |                       |
| MAG(FFT(<source>))                                                                                                                                                                 | Magnitude of complex result             |                       |
| PHASE(FFT(<source>))                                                                                                                                                               | Phase angle (degrees) of complex result |                       |
| PS(FFT(<source>))                                                                                                                                                                  | Power spectrum                          |                       |
| PSD(FFT(<source>))                                                                                                                                                                 | Power density                           |                       |
| RESC([ <b>{+, - }</b> ][<multiplier>*<source>[ <b>{+, - }</b> <addend>])                                                                                                           | Rescale                                 |                       |
| <b>Power Average Functions Available on Instruments with WP02 Processing Firmware</b><br><i>Note: The source waveform must be another function defined as a Fourier transform.</i> |                                         |                       |
| MAG(AVGP(<function>))                                                                                                                                                              | PS(AVGP(<function>))                    | PSD(AVGP(<function>)) |
| <b>Function Equations and Names Available on Instruments with WP03 or DDM Firmware</b>                                                                                             |                                         |                       |
| HIST(<custom_line>)                                                                                                                                                                | Histogram of parameter on custom line   |                       |
| <b>Function Equations and Names Available on Instruments with PRML Firmware</b>                                                                                                    |                                         |                       |
| CORR(<source1>,<source2>)                                                                                                                                                          | Cross Correlation                       |                       |

## 9300 &amp; LC Series

**Source values**

*Note: The numbers in CUST1, CUST2, CUST3, CUST4, and CUST5 refer to the line numbers of the selected custom parameters.*

<sourceN> : = {TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, C3G, C4G}

<function> : = {TA, TB, TC, TD}

<custom\_line> : = {CUST1, CUST2, CUST3, CUST4, CUST5}

<extended\_source> : = {C1, C2, C3G, C4G, 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 (For standard instruments)

<max\_sweeps> : = 1 to 1 000 000 (For WP01 only)

<max\_sweeps> : = 1 to 50 000 (WP02 Power Spectrum only)

**Values for Resample Function**

<delay> : =  $-2e-6$  to  $+2e-6$  seconds

**Values for Rescale Function**

<addend> : = 0.0 to  $1e15$

<multiplier> : = 0.0 to  $1e15$

**Values for Summation Average and ERES**

<weight> : = {1, 3, 7, 15, 31, 63, 127, 255, 511, 1023}

<bits> : = {0.5, 1.0, 1.5, 2.0, 2.5, 3.0}

**Values for FFT window function**

<window\_type> : = {BLHA, FLTP, HAMM, HANN, RECT}

| FFT Window Function Notation |                        |
|------------------------------|------------------------|
| LHA                          | Blackman–Harris window |
| FLTP                         | Flat Top window        |
| HABMM                        | Hamming window         |
| HANN                         | von Hann window        |
| RECT                         | Rectangular window     |



#### Values for WP03 histogramming

<max\_bins> : = {20, 50, 100, 200, 500, 1000, 2000}

<max\_events> : = 20 to 2e9 (in a 1–2–5 sequence)

<center> : = –1e15 to 1e15

<width> : = 1e–30 to 1e30 (in a 1–2–5 sequence)

<vert\_scale> : = {LIN, LOG, CONSTMAX}

| Histogram Notation |                                                           |
|--------------------|-----------------------------------------------------------|
| <b>LIN</b>         | Use linear vertical scaling for histogram display         |
| <b>LOG</b>         | Use log vertical scaling for histogram display            |
| <b>CONSTMAX</b>    | Use constant maximum linear scaling for histogram display |

#### Values for PRML correlation

<length> : = 0 to 10 divisions

<start> : = 0 to 10 divisions

## G AVAILABILITY

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

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

SWEEPS is the maximum number of sweeps (Average and Extrema only).

*Note: The pair SWEEPS,<max\_sweeps> applies only to the summed averaging (AVGS).*

### EXAMPLE (GPIB)

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

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

### WP01 EXAMPLE

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

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

## 9300 &amp; LC Series

**WP02 FFT EXAMPLE (GPIB)** The following instruction 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$="TA:DEF EQN, 'PS(FFT(C1))', MAXPTS, 1000, WINDOW, RECT": CALL IBWRT(SCOPE%, CMD$)
```

**WP02 PS EXAMPLE (GPIB)** The following instruction 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$="TB:DEF EQN, 'PS(AVGP(TA))', SWEEPS, 244": CALL IBWRT(SCOPE%, CMD$)
```

**WP03 EXAMPLE**

The following command 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

**MASS STORAGE**

**DELETE\_FILE, DELF  
Command**

**DESCRIPTION**                      The DELETE\_FILE command deletes files from the currently selected directory on mass storage.

**COMMAND SYNTAX**                **DELeTe\_File DISK, <device>, FILE, '<filename>'**  
                                          <device> : = {**CARD<sup>G</sup>, FLPY, HDD<sup>G</sup>**}  
                                          <filename> : = An alphanumeric string of up to eight characters, followed by a dot and an extension of up to three characters.

**G AVAILABILITY**                      <device> : CARD available only when MC01 option is fitted.  
                                          <device> : HDD available only when HD01 option is fitted.

**EXAMPLE (GPIB)**                    The following command deletes a front-panel setup from the memory card:  
                                          **CMD\$="DELF DISK, CARD, FILE, 'P001.PNL'":**  
                                          **CALL IBWRT(SCOPE%, CMD\$)**

**RELATED COMMANDS**                DIRECTORY, FORMAT\_CARD, FORMAT\_FLOPPY,  
                                          FORMAT\_HDD

**MASS STORAGE****DIRECTORY, DIR**  
Command/Query**DESCRIPTION**

The DIRECTORY command is used to manage the creation and deletion of file directories on mass storage devices. It also allows selection of the current working directory and listing of files in the directory.

The query response consists of a double-quoted string containing a DOS-like listing of the directory. If no mass storage device is present, or if it is not formatted, the string will be empty.

**COMMAND SYNTAX**

**DIRectory DISK**, <device>, **ACTION**, <action>, ' <directory> '

**QUERY SYNTAX**

**DIRectory?** **DISK**, <device> [, ' <directory> ']

<device> : = {**CARDG**, **FLPY**, **HDDG**}

<action> : = {**CREATE**, **DELETE**, **SWITCH**}

<directory> : = A legal DOS path or filename. (This can include the '\ ' character to define the root directory.)

*Note: the query DIRectory\_list? is also accepted for backward compatibility but may not be supported in the future.*

**RESPONSE FORMAT**

**DIRectory DISK**, <device> " <directory> "

<directory> : = A variable length string detailing the file content of the memory card, floppy disk or hard disk.

**G AVAILABILITY**

<device> : CARD available only when MC01 option is fitted.

<device> : HDD available only when HD01 option is fitted.

### EXAMPLE (GPIB)

The following asks for a listing of the directory of the memory card:

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

Response message:

```
DIR "  
Directory                   LECROY           1 DIR of 04-MAR-1994  
    10:46:20 on Memory Card  
SC1 000      2859      19-DEC-1994 16:33:06  
SC1 001      2859      19-DEC-1994 16:34:32  
TEST5        002      20359        12-MAR-1994  
    13:34:12  
3 File(s) 1948672 bytes free  
"
```

**DISPLAY****DISPLAY, DISP  
Command/Query****DESCRIPTION**

The DISPLAY command controls the display screen of the oscilloscope. When the user is remotely controlling the oscilloscope and does not need to use the display, it can be useful to switch off the display via the DISPLAY OFF command. This improves instrument 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 the display has been set to OFF, the real-time clock and the message field are updated. However, the waveforms and associated texts remain unchanged.*

**COMMAND SYNTAX**

**DISP**lay <state>  
<state> : = {**ON**, **OFF**}

**QUERY SYNTAX**

**DISP**lay?

**RESPONSE FORMAT**

**DISP**lay <state>

**EXAMPLE (GPIB)**

The following instruction turns off the display generation:

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

| <i>DISPLAY</i>         | <b>DOT_JOIN, DTJN</b><br>Command/Query                                                                            |
|------------------------|-------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>     | The DOT_JOIN command controls the interpolation lines between data points.                                        |
| <b>COMMAND SYNTAX</b>  | <b>DoT_JoiN</b> <state><br><state> : = { <b>ON</b> , <b>OFF</b> }                                                 |
| <b>QUERY SYNTAX</b>    | <b>DoT_JoiN?</b>                                                                                                  |
| <b>RESPONSE FORMAT</b> | <b>DoT_JoiN</b> <state>                                                                                           |
| <b>EXAMPLE (GPIB)</b>  | The following instruction turns off the interpolation lines:<br><b>CMD\$="DTJN OFF": CALL IBWRT(SCOPE%,CMD\$)</b> |

**DISPLAY****DUAL\_ZOOM, DZOM**  
Command/Query**DESCRIPTION**

By setting DUAL\_ZOOM ON, the horizontal magnification and positioning controls are applied to all expanded traces simultaneously. This command is useful if the contents of all expanded traces are to be examined at the same time.

The DUAL\_ZOOM? query indicates whether multiple zoom is enabled or not.

*Note: This command has the same effect as MULTI\_ZOOM.*

**COMMAND SYNTAX**

**Dual\_Zoom** <mode>

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

**QUERY SYNTAX**

**Dual\_Zoom?**

**RESPONSE FORMAT**

**Dual\_Zoom** <mode>

**EXAMPLE (GPIB)**

The following instruction turns dual zoom on:

**CMD\$="DZOM ON": CALL IBWRT(SCOPE%, CMD\$)**

**RELATED COMMANDS**

HOR\_MAGNIFY, HOR\_POSITION, MULTI\_ZOOM



| STATUS                  | <b>*ESE</b><br>Command/Query                                                                                                                                                                                                                                                                                                                                            |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>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. <i>For an overview of the ESB defined events refer to the ESR table on page 70.</i></p> <p>The *ESE? query reads the contents of the ESE register.</p> |
| <b>COMMAND SYNTAX</b>   | <p><b>*ESE</b> &lt;value&gt;<br/>           &lt;value&gt; : = 0 to 255</p>                                                                                                                                                                                                                                                                                              |
| <b>QUERY SYNTAX</b>     | <p><b>*ESE?</b></p>                                                                                                                                                                                                                                                                                                                                                     |
| <b>RESPONSE FORMAT</b>  | <p><b>*ESE</b> &lt;value&gt;</p>                                                                                                                                                                                                                                                                                                                                        |
| <b>EXAMPLE (GPIB)</b>   | <p>The following instruction 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.</p> <p><b>CMD\$="*ESE 72": CALL IBWRT(SCOPE%,CMD\$)</b></p>                                                                     |
| <b>RELATED COMMANDS</b> | <p>*ESR</p>                                                                                                                                                                                                                                                                                                                                                             |

**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     | <b>*ESR?</b>                                                                                                                                                                                                                                     |
| RESPONSE FORMAT  | <b>*ESR</b> <value><br><value> : = 0 to 255                                                                                                                                                                                                      |
| EXAMPLE (GPIB)   | <p>The following instruction reads and clears the contents of the ESR register:</p> <pre><b>CMD\$="*ESR?": CALL IBWRT(SCOPE%,CMD\$):</b><br/><b>CALL IBRD(SCOPE%,RSP\$): PRINT RSP\$</b></pre> <p>Response message:</p> <pre><b>*ESR 0</b></pre> |
| RELATED COMMANDS | ALL_STATUS, *CLS, *ESE                                                                                                                                                                                                                           |

## ADDITIONAL INFORMATION

| Standard Event Status Register (ES) |           |          |             |                                         |      |
|-------------------------------------|-----------|----------|-------------|-----------------------------------------|------|
| Bit                                 | Bit Value | Bit Name | Description |                                         | 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 specific Error occurred          | (5)  |
| 2                                   | 4         | QYE      | 1           | Query Error occurred                    | (6)  |
| 1                                   | 2         | RQC      | 0           | Instrument never requests bus control   | (7)  |
| 0                                   | 1         | OPC      | 0           | OPeration Complete bit not used         | (8)  |

**Notes**

- (1) *The Power On (PON) bit is always turned on (1) when the unit is powered up.*
- (2) *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.*
- (3) *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.*
- (4) *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 may be localized via the DDR? or the self test \*TST? 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.*

| STATUS                  | EXR?<br>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. <i>Refer to the table next page for further details.</i> |
| <b>QUERY SYNTAX</b>     | <b>EXR?</b>                                                                                                                                                                                                                   |
| <b>RESPONSE FORMAT</b>  | <b>EXR</b> <value><br><value> : = 21 to 64                                                                                                                                                                                    |
| <b>EXAMPLE (GPIB)</b>   | The following instruction reads the contents of the EXR register:<br><b>CMD\$="EXR?": CALL IBWRT(SCOPE%,CMD\$):</b><br><b>CALL IBRD(SCOPE%,RSP\$): PRINT RSP\$</b><br>Response message (if no fault):<br><b>EXR 0</b>         |
| <b>RELATED COMMANDS</b> | ALL_STATUS, *CLS                                                                                                                                                                                                              |

## 9300 &amp; LC Series

## 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 instrument 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, or 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. *  |

---

\* For floppy disk and on oscilloscopes fitted with the memory card (MCO1) or hard disk (HD01) options.

## 9300 &amp; LC Series

**DISPLAY****FAT\_CURSOR, FATCG**  
Command/Query

**DESCRIPTION** The FAT\_CURSOR command controls the width of the cursor.

**COMMAND SYNTAX** **FAT\_Cursor** <state>  
<state> : = {**ON**, **OFF**}

**QUERY SYNTAX** **FAT\_Cursor?**

**RESPONSE FORMAT** **FAT\_Cursor** <state>

**G AVAILABILITY** LC scopes only.

**EXAMPLE (GPIB)** The following sets the cursor's appearance to fat:  
**CMD\$="FAT\_Cursor ON": CALL IBWRT(SCOPE%,CMD\$)**



**MASS STORAGE**

**FILENAME, FLNM**  
Command/Query

|                         |                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The FILENAME command is used to change the default filename given to any traces, setups and hard copies when they are being stored to a mass storage device.                                                                                                                                                                                                                                                      |
| <b>COMMAND SYNTAX</b>   | <p><b>FiLeNaMe</b> TYPE, &lt;type&gt;, FILE, '&lt;filename&gt;'</p> <p>&lt;type&gt; := {C1, C2, C3G, C4G, TA, TB, TC, TD, SETUP, HCOPY }</p> <p>&lt;filename&gt; := For C1 to TD, an alphanumeric string of up to 8 characters forming a legal DOS filename. Up to 5 characters for SETUP and HCOPY.</p> <p><i>Note: No extension can be specified as this is automatically assigned by the oscilloscope.</i></p> |
| <b>QUERY SYNTAX</b>     | <p><b>FiLeNaMe?</b> TYPE, &lt;type&gt;</p> <p>&lt;type&gt; := {ALL, C1, C2, C3G, C4G, TA, TB, TC, TD, SETUP, HCOPY}</p>                                                                                                                                                                                                                                                                                           |
| <b>RESPONSE FORMAT</b>  | <p><b>FiLeNaMe</b></p> <p>TYPE, &lt;type&gt;, FILE, "&lt;filename&gt;"[, TYPE, &lt;type&gt;, FILE, "&lt;filename&gt;"...]</p>                                                                                                                                                                                                                                                                                     |
| <b>AVAILABILITY</b>     | <trace> := {C3, C4} available only on four-channel oscilloscopes.                                                                                                                                                                                                                                                                                                                                                 |
| <b>EXAMPLE (GPIB)</b>   | <p>The following command designates channel 1 waveform files to be "TESTPNT6.xxx" where xxx is a numeric extension assigned by the oscilloscope:</p> <p><b>CMD\$="FLNM TYPE,C1, FILE, 'TESTPNT6'":</b><br/><b>CALL IBWRT(SCOPE%,CMD\$)</b></p>                                                                                                                                                                    |
| <b>RELATED COMMANDS</b> | DIRECTORY, FORMAT_CARD, FORMAT_FLOPPY, FORMAT_HDD, DELETE_FILE                                                                                                                                                                                                                                                                                                                                                    |

**FUNCTION****FIND\_CTR\_RANGE, FCRG  
Command****DESCRIPTION**

The FIND\_CTR\_RANGE command automatically sets the center and width of a histogram to best display the accumulated events.

**COMMAND SYNTAX**

<function> : **Find\_Ctr\_Range**

<function> : = {**TA,TB,TC,TD**}

**G AVAILABILITY**

Only available on instruments fitted with WP03 or DDM option.

**EXAMPLE (GPIB)**

Assuming that Trace A (TA) has been defined as a histogram of one of the custom parameters, the following example will determine the best center and width and then rescale the histogram:

**CMD\$="TA:FCR": CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS**

DEFINE, PACU

**MASS STORAGE**

**FORMAT\_CARD, FCRD**  
Command/Query

**DESCRIPTION**                      The **FORMAT\_CARD** command formats the memory card according to the PCMCIA/JEIDA standard with a DOS partition.  
The **FORMAT\_CARD?** query returns the status of the card.

**COMMAND SYNTAX**                **Format\_CaRD**

**QUERY SYNTAX**                   **Format\_CaRD?**

**RESPONSE FORMAT**               **Format\_CaRD** <card\_status>[, <read/write>, <free\_space>, <card\_size>, <battery\_status>]  
  
<card\_status> : = {**NONE, BAD, BLANK, DIR\_MISSING, OK**}  
<read/write> : = {**WP, RW**}  
<free\_space> : = A decimal number giving the number of bytes still available on the card  
<card\_size> : = A decimal number giving the total number of bytes on the card.  
<battery\_status> : = {**BAT\_OK, BAT\_LOW, BAT\_BAD**}

**G AVAILABILITY**                      Available only on instruments fitted with the MC01 option.

**EXAMPLE (GPIB)**                   The following code will first format a memory card and then verify its status:

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

Response message:

**FCRD OK, RW, 130048, 131072, BAT\_OK**

**RELATED COMMANDS**               **DIRECTORY**

## 9300 &amp; LC Series

## ADDITIONAL INFORMATION

| Notation           |                                                                                                                  |
|--------------------|------------------------------------------------------------------------------------------------------------------|
| <b>BAD</b>         | Bad card after formatting                                                                                        |
| <b>BAT_BAD</b>     | Bad battery or no battery                                                                                        |
| <b>BAT_LOW</b>     | Battery should be replaced                                                                                       |
| <b>BAT_OK</b>      | Battery is in order                                                                                              |
| <b>BLANK</b>       | Current directory empty                                                                                          |
| <b>DIR_MISSING</b> | No subdirectory present. The directory "LECROY1_DIR" will be automatically created with the next "store" command |
| <b>NONE</b>        | No card                                                                                                          |
| <b>OK</b>          | Card is correctly formatted                                                                                      |
| <b>RW</b>          | Read/Write authorized                                                                                            |
| <b>WP</b>          | Write protected                                                                                                  |

**MASS STORAGE**

**FORMAT\_FLOPPY, FFLP**  
Command/Query

|                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>The <b>FORMAT_FLOPPY</b> command formats a floppy disk in the Double Density or High Density format.</p> <p>The <b>FORMAT_FLOPPY?</b> query returns the status of the floppy disk.</p>                                                                                                                                                                                                                                                           |
| <b>COMMAND SYNTAX</b>   | <p><b>Format_FLoPpy</b> [&lt;type&gt;]</p> <p>&lt;type&gt; : = {<b>DD, HD</b>}</p> <p>If no argument is supplied, HD is used by default.</p>                                                                                                                                                                                                                                                                                                        |
| <b>QUERY SYNTAX</b>     | <p><b>Format_FLoPpy?</b></p>                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>RESPONSE FORMAT</b>  | <p><b>Format_FLoPpy</b> &lt;floppy_status&gt;[, &lt;read/write&gt;, &lt;free_space&gt;, &lt;floppy_size&gt;]</p> <p>&lt;floppy_status&gt; : = {<b>NONE, BAD, BLANK, DIR_MISSING, OK</b>}</p> <p>&lt;read/write&gt; : = {<b>WP, RW</b>}</p> <p>&lt;free_space&gt; : = A decimal number giving the number of bytes still available on the floppy.</p> <p>&lt;floppy_size&gt; : = A decimal number giving the total number of bytes on the floppy.</p> |
| <b>EXAMPLE (GPIB)</b>   | <p>The following code will first format a floppy in the Double Density (720 kB) format and then verify its status:</p> <pre> <b>CMD\$="FFLP DD":IBWRT(SCOPE%,CMD\$)CMD\$="FFLP?":</b> <b>CALL IBWRT(SCOPE%,CMD\$):</b> <b>CALL IBRD(SCOPE%,RSP\$): PRINT RSP\$</b> </pre> <p>Response message:</p> <pre> <b>FFLP OK,RW,728064,737280,</b> </pre>                                                                                                    |
| <b>RELATED COMMANDS</b> | <p><b>DIRECTORY</b></p>                                                                                                                                                                                                                                                                                                                                                                                                                             |

## 9300 &amp; LC Series

## ADDITIONAL INFORMATION

| Notation           |                                                                                                                   |
|--------------------|-------------------------------------------------------------------------------------------------------------------|
| <b>BAD</b>         | Bad floppy after formatting                                                                                       |
| <b>BLANK</b>       | Current directory empty                                                                                           |
| <b>DD</b>          | Double Density 720 kB formatted                                                                                   |
| <b>DIR_MISSING</b> | No subdirectory present. The directory "LECROY1_DIR" will be automatically created with the next "store" command. |
| <b>HD</b>          | High Density 1.44 MB formatted                                                                                    |
| <b>NONE</b>        | No floppy                                                                                                         |
| <b>OK</b>          | Floppy is correctly formatted                                                                                     |
| <b>RW</b>          | Read/Write authorized                                                                                             |
| <b>WP</b>          | Write protected                                                                                                   |

**MASS STORAGE**

**FORMAT\_HDD, FHDD**  
Command/Query

**DESCRIPTION**                      The **FORMAT\_HDD** command formats the removable hard disk according to the PCMIA/JEIDA standard with a DOS partition.  
The **FORMAT\_HDD?** query returns the status of the hard disk.

**COMMAND SYNTAX**                **Format\_HDD** <type>  
<type> := {**QUICK, FULL**}  
If no argument is supplied, **QUICK** will be used.

**QUERY SYNTAX**                    **Format\_HDD?**

**RESPONSE FORMAT**                **Format\_HDD** <hdd\_status>[, <read/write>, <free\_space>, <hdd\_size>]  
<hdd\_status> := {**NONE, BAD, BLANK, DIR\_MISSING, OK**}  
<read/write> := {**WP, RW**}  
<free\_space> := A decimal number giving the number of byte still available on the hard disk  
<hdd\_size> := A decimal number giving the total number of bytes on the hard disk.

**G AVAILABILITY**                    Available only on instruments fitted with the HD01 option.

**EXAMPLE (GPIB)**                    The following code will first format a hard disk and then verify its status:

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

Response message:

**FHDD OK, RW, 3076096, 105744896**

**RELATED COMMANDS**                **DIRECTORY**

## 9300 &amp; LC Series

## ADDITIONAL INFORMATION

| Notation           |                                                                                                                  |
|--------------------|------------------------------------------------------------------------------------------------------------------|
| <b>BAD</b>         | Bad hard disk after formatting                                                                                   |
| <b>BLANK</b>       | Current directory empty                                                                                          |
| <b>DIR_MISSING</b> | No subdirectory present. The directory "LECROY1_DIR" will be automatically created with the next "store" command |
| <b>NONE</b>        | No hard disk                                                                                                     |
| <b>OK</b>          | Hard disk is correctly formatted                                                                                 |
| <b>RW</b>          | Read/Write authorized                                                                                            |
| <b>WP</b>          | Write protected                                                                                                  |



## ADDITIONAL INFORMATION

| Notation           |                                                                                                                  |
|--------------------|------------------------------------------------------------------------------------------------------------------|
| <b>BAD</b>         | Bad virtual disk after formatting                                                                                |
| <b>BLANK</b>       | Current directory empty                                                                                          |
| <b>DIR_MISSING</b> | No subdirectory present. The directory "LECROY1_DIR" will be automatically created with the next "store" command |
| <b>NONE</b>        | No virtual disk                                                                                                  |
| <b>OK</b>          | Virtual disk is correctly formatted                                                                              |
| <b>RW</b>          | Read/Write authorized                                                                                            |
| <b>WP</b>          | Write protected                                                                                                  |

**DISPLAY****FULL\_SCREEN, FSCR<sup>G</sup>**  
Command/Query**DESCRIPTION**

The FULL\_SCREEN command is used to control whether the currently selected grid style is displayed in normal presentation format or with a full-screen grid. In Full Screen format, the waveform display areas are enlarged to the maximum possible size.

The response to the FULL\_SCREEN? query indicates whether or not the display is operating in Full Screen presentation format.

**COMMAND SYNTAX**

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

**QUERY SYNTAX**

**FullScreen?**

**RESPONSE FORMAT**

**FullScreen** <state>

**G AVAILABILITY**

Available only on color instruments.

**EXAMPLE (GPIB)**

The following instruction enables the Full Screen presentation format:

**CMD\$="FSCR ON": CALL IBWRT(SCOPE%,CMD\$)**

| <i>FUNCTION</i>         | <b>FUNCTION_RESET, FRST</b><br>Command                                                                                                                                                                                                                                                                                                              |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The FUNCTION_RESET command resets a waveform processing function. The number of sweeps will be reset to zero and the process restarted.                                                                                                                                                                                                             |
| <b>COMMAND SYNTAX</b>   | <function> : <b>Function_ReSeT</b>                                                                                                                                                                                                                                                                                                                  |
| <b>EXAMPLE (GPIB)</b>   | <p data-bbox="618 552 984 583">&lt;function&gt; : = {<b>TA,TB,TC,TD</b>}</p> <p data-bbox="618 600 1385 695">Assuming that Trace A (TA) has been defined as the summed average of Channel 1, the following instruction will restart the averaging process:</p> <p data-bbox="618 705 1235 737"><b>CMD\$="TA:FRST": CALL IBWRT(SCOPE%,CMD\$)</b></p> |
| <b>RELATED COMMANDS</b> | DEFINE, INR                                                                                                                                                                                                                                                                                                                                         |

| ACQUISITION      |                                                                                                                                                                                                                                                                                                                         | GLOBAL_BWL, GBWL G<br>Command/Query |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------|
| DESCRIPTION      | This turns on or off the Global Bandwidth Limit. When activated, the Bandwidth Limit applies to all channels; when deactivated, a Bandwidth Limit can be set individually for each channel (see <i>BWL</i> , page 19). The response to the GLOBAL_BWL? query indicates whether the Global Bandwidth Limit is on or off. |                                     |
| COMMAND SYNTAX   | <b>Global_BWL</b> <mode><br><mode> : = { <b>OFF</b> , <b>ON</b> }                                                                                                                                                                                                                                                       |                                     |
| QUERY SYNTAX     | <b>Global_BWL?</b>                                                                                                                                                                                                                                                                                                      |                                     |
| RESPONSE FORMAT  | <b>Global_BWL</b> <mode>                                                                                                                                                                                                                                                                                                |                                     |
| EXAMPLE          | The following instruction deactivates the Global Bandwidth Limit, allowing a Bandwidth Limit to be set individually for each channel (using the BWL command syntax for individual channels):<br><b>CMD\$="GBWL OFF": CALL IBWRT(SCOPE%, CMD\$)</b>                                                                      |                                     |
| RELATED COMMANDS | BANDWIDTH_LIMIT                                                                                                                                                                                                                                                                                                         |                                     |

**DISPLAY**

**GRID**  
Command/Query

|                         |                                                                                                                                                                                       |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>The GRID command specifies whether the display is in single (1), dual (2), quad (4), XY or octal (8) grid mode.</p> <p>The GRID? query returns the grid mode currently in use.</p> |
| <b>COMMAND SYNTAX</b>   | <p><b>GRID</b> &lt;grid&gt;</p> <p>&lt;grid&gt; : = {<b>SINGLE, DUAL, QUAD, OCTAL, XYONLY</b>G}</p>                                                                                   |
| <b>QUERY SYNTAX</b>     | <b>GRID?</b>                                                                                                                                                                          |
| <b>AVAILABILITY</b>     | <p>&lt;grid&gt; : = XYONLY available only when XY Display used.</p>                                                                                                                   |
| <b>RESPONSE FORMAT</b>  | <b>GRID</b> <grid>                                                                                                                                                                    |
| <b>EXAMPLE (GPIB)</b>   | <p>The following instruction sets the screen display to dual grid mode:</p> <p><b>CMD\$="GRID DUAL": CALL IBWRT(SCOPE%,CMD\$)</b></p>                                                 |
| <b>RELATED COMMANDS</b> | COLOR, INTENSITY, FULL_SCREEN                                                                                                                                                         |

**HARD COPY****HARDCOPY\_SETUP, HCSU**  
Command/Query**DESCRIPTION**

The **HARDCOPY\_SETUP** command configures the instrument's hard-copy driver. It enables the user to specify the device type and transmission mode of the hard-copy unit connected to the oscilloscope. One or more individual settings can be changed by specifying the appropriate keyword(s), together with the new value(s). *See following pages for command notation and printer or plotter model availability.*

**COMMAND SYNTAX**

**HardCopy\_SetUp** **DEV**, <device>, **PORT**, <port>, **PFEED**, <page\_feed>, **PENS**, <plot\_pens>, **PSIZE**, <paper\_size> **CMDIV**, <cmdiv>, **AUTO**, <auto>, **FORMAT**, <format>, **BCKG**, <bckg>

<device> : = {**BMP**, **BMPCOMP**, **CANONCOL**, **EPSON**, **EPSONCOLG**,  
**HPDJ**, **HPDJBW**, **HPPJG**, **HPTJG**, **HPLJ**,  
**HP7470A**, **HP7550A**, **TIFF**, **TIFFCOLG**,  
**TIFFCOMP****G**}

<port> : = {**GPIB**, **RS**, **CENTG**, **FLPY**, **CARDG**, **HDDG**, **PRTG**, }

<page\_feed> : = {**OFF**, **ON**}

<plot\_pens> : = 1 to 8

<paper\_size> : = {**A5**, **A4**}

<cmdiv> : = {**1**, **2**, **5**, **10**, **20**, **50**, **100**, **200**}

<auto> : = {**OFF**, **ON**}

<format> : = {**PORTRAIT**, **LANDSCAPE**}

<bckg> : = {**BLACK**, **WHITE**}

**QUERY SYNTAX**

**HardCopy\_SetUp?**

**RESPONSE FORMAT**

**HardCopy\_SetUp** **DEV**, <device>, **PORT**, <port>, **PFEED**, <page\_feed>, **PENS**, <plot\_pens>, **PSIZE**, <paper\_size>, **CMDIV**, <cmdiv>, **AUTO**, <auto>, **FORMAT**, <format>, **BCKG**, <bckg>

**G AVAILABILITY**

<card> : CARD only available when MC01 Option is fitted.

<port> : HDD only when HD01 Option is fitted.

<port> : PRT only when GP01 option is fitted.

<cmdiv> only when GP01 option is fitted.

<auto> only when GP01 option is fitted.

<device> See table page 90 .

The following table lists the printer and graphic formats that can be used for producing hardcopies remotely, either with monochrome or color digital oscilloscopes, or both, using <device>.

| Notation        | Printer, Plotter or Protocol            | Monochrome Instruments | Color Instruments    |
|-----------------|-----------------------------------------|------------------------|----------------------|
| <b>BMP</b>      | BMP                                     | ✓                      | ✓                    |
| <b>BMPCOMP</b>  | BMP compressed                          | <i>not available</i>   | ✓                    |
| <b>CANONCOL</b> | Canon 200/600/800 Series color printers | <i>not available</i>   | ✓                    |
| <b>EPSON</b>    | Epson b&w                               | ✓                      | ✓                    |
| <b>EPSONCOL</b> | Epson color                             | <i>not available</i>   | ✓                    |
| <b>HPTJ</b>     | HP ThinkJet                             | ✓                      | <i>not available</i> |
| <b>HPPJ</b>     | HP Paint Jet                            | ✓                      | <i>not available</i> |
| <b>HPLJ</b>     | HP LaserJet                             | ✓                      | ✓                    |
| <b>HPDJ</b>     | HP Desk Jet color                       | ✓                      | ✓                    |
| <b>HPDJBW</b>   | HP Desk Jet b&w                         | ✓                      | ✓                    |
| <b>HP7470A</b>  | HP 7470A plotter                        | ✓                      | ✓                    |
| <b>HP7550A</b>  | HP 7550A plotter                        | ✓                      | ✓                    |
| <b>HPGL</b>     | Vector screen file                      | ✓                      | ✓                    |
| <b>TIFF</b>     | TIFF                                    | ✓                      | ✓                    |
| <b>TIFFCOMP</b> | TIFF compressed                         | ✓                      | <i>not available</i> |
| <b>TIFFCOL</b>  | TIFF color                              | <i>not available</i>   | ✓                    |

## 9300 &amp; LC Series

**EXAMPLE (GPIB)**

The following example selects an EPSON printer connected via the RS232 port:

```
CMD$="HCSU PORT,RS,DEV,EPSON"  
CALL IBWRT(SCOPE%,CMD$)
```

**RELATED COMMANDS**

HARDCOPY\_TRANSMIT, SCREEN\_DUMP

**ADDITIONAL INFORMATION**

Hardcopy 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 may be given in any order and may be restricted to those variables to be changed.

The following table gives the Hardcopy command notations and their meanings.

| Notation      |                                             |
|---------------|---------------------------------------------|
| <b>DEV</b>    | Device                                      |
| <b>PENS</b>   | Plotter: plot pens                          |
| <b>PFEED</b>  | Page feed                                   |
| <b>PORT</b>   | Transmission mode                           |
| <b>CARD</b>   | Memory card                                 |
| <b>HDD</b>    | Hard Disk                                   |
| <b>CENT</b>   | Centronics port                             |
| <b>FLPY</b>   | Floppy disk                                 |
| <b>GPIB</b>   | IEEE-488 port                               |
| <b>PRT</b>    | Internal printer                            |
| <b>RS</b>     | RS-232-C port                               |
| <b>CMDIV</b>  | Internal printer: cm/division               |
| <b>PSIZE</b>  | Plotter: paper size                         |
| <b>AUTO</b>   | Auto print                                  |
| <b>FORMAT</b> | Orientation of print: Portrait or Landscape |



**HARD COPY**

**HARDCOPY\_TRANSMIT, HCTR  
Command**

**DESCRIPTION**

The HARDCOPY\_TRANSMIT command sends a string of ASCII characters without modification to the hard-copy unit. This allows the user to control the hard-copy unit by sending device-specific control character sequences. It also allows placing of additional text on a screen dump for documentation purposes.

**COMMAND SYNTAX**

**HardCopy\_TRAnsmIt** '<string>'

<string> := Any sequence of ASCII characters or escape sequences.

*Note: This command accepts the escape sequences as described under the command COMM\_RS232. Before sending the string to the hard-copy unit the escape sequence is converted to the ASCII character code.*

**EXAMPLE (GPIB)**

The following instruction sends documentation data to a printer:

**CMD\$="HCTR 'Data from Oct.15\r\n'" CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS**

HARDCOPY\_SETUP, SCREEN\_DUMP

**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 167*) 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> : = {**TA, TB, TC, TD**}

<factor> : = 1 to 20000

**QUERY SYNTAX**

<exp\_source> : **Hor\_MAGnify?**

**RESPONSE FORMAT**

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

**EXAMPLE (GPIB)**

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

**CMD\$="TA: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 167) 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> : = {**TA, TB, TC, TD**}

<hor\_position> : = 0 to 10 DIV

<segment> : = 0 to max segments

*Note 1: The suffix DIV is optional.*

*Note 2: 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.*

## 9300 &amp; LC Series

|                  |                                                                                                                                                                                           |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| QUERY SYNTAX     | <exp_trace> : <b>Hor_POSition?</b>                                                                                                                                                        |
| RESPONSE FORMAT  | <exp_trace> : <b>Hor_POSition</b> <hor_position>[, <segment>]<br><i>Note 3: The segment number is only given for sequence waveforms.</i>                                                  |
| EXAMPLE (GPIB)   | The following instruction positions the center of the intensified zone on the trace currently viewed by Trace A (TA) at division 3:<br><b>CMD\$="TA:HPOS 3": CALL IBWRT(SCOPE%,CMD\$)</b> |
| RELATED COMMANDS | DUAL_ZOOM, 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 example issues an identification request to the scope:

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

Response message:

```
*IDN LECROY, 9314CM, 931401000, 7.7.0
```

| STATUS           | INE<br>Command/Query                                                                                                                                                                                                                                                                                                                                              |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DESCRIPTION      | <p>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. <i>For an overview of the INR defined events, refer to the table next page.</i></p> <p>The INE? query reads the contents of the INE register.</p> |
| COMMAND SYNTAX   | <p><b>INE</b> &lt;value&gt;</p> <p>&lt;value&gt; : = 0 to 65535</p>                                                                                                                                                                                                                                                                                               |
| QUERY SYNTAX     | <p><b>INE?</b></p>                                                                                                                                                                                                                                                                                                                                                |
| RESPONSE FORMAT  | <p><b>INE</b> &lt;value&gt;</p>                                                                                                                                                                                                                                                                                                                                   |
| EXAMPLE (GPIB)   | <p>The following instruction 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.</p> <p><b>CMD\$="INE 3": CALL IBWRT(SCOPE%,CMD\$)</b></p>                                                   |
| RELATED COMMANDS | <p>INR</p>                                                                                                                                                                                                                                                                                                                                                        |

**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...14                                 |           | 0           | Reserved for future use                                                   |
| 13                                      | 8192      | 1           | Trigger is ready                                                          |
| 12                                      | 4096      | 1           | Pass/Fail test detected desired outcome                                   |
| 11                                      | 2048      | 1           | Waveform processing has terminated in Trace D                             |
| 10                                      | 1024      | 1           | Waveform processing has terminated in Trace C                             |
| 9                                       | 512       | 1           | Waveform processing has terminated in Trace B                             |
| 8                                       | 256       | 1           | Waveform processing has terminated in Trace A                             |
| 7                                       | 128       | 1           | A memory card, floppy or hard disk exchange has been detected             |
| 6                                       | 64        | 1           | Memory card, 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                        |
| 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                                            |

**QUERY SYNTAX**
**INR?**
**RESPONSE FORMAT**

**INR** <state>  
<state> : = 0 to 65535

## 9300 & LC Series

### EXAMPLE (GPIB)

The following instruction 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 the user 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 query TEMPLATE? 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).**

The special logical block named WAVEDESC may 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     |                                                                                                            |
|--------------|------------------------------------------------------------------------------------------------------------|
| <b>BYTE</b>  | raw data as integers (truncated to 8 (m.s.b. <sup>†</sup> ))                                               |
| <b>FLOAT</b> | normalized data (gain, offset applied) as floating point numbers (gives measured values in volts or units) |
| <b>WORD</b>  | raw data as integers (truncated to 16 m.s.b.)                                                              |

**QUERY SYNTAX**

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

<trace> : = {**TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, c3G,c4G**}

<string> : = A valid name of a logical block or a valid name of a variable contained in block WAVEDESC (see the command TEMPLATE).

<data\_type> : = {**BYTE, WORD, FLOAT**}

*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.*

---

<sup>†</sup> most significant bits

## 9300 &amp; LC Series

## RESPONSE FORMAT

<trace> : **INSPect** "<string>"

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

**G** AVAILABILITY

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

## EXAMPLES (GPIB)

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

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

## RELATED COMMANDS

TEMPLATE, WAVEFORM\_SETUP

| <i>DISPLAY</i>         | <b>INTENSITY, INTS</b><br>Command/Query                                                                                                                                                                                                                                                                                                                                                                               |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>     | <p>The INTENSITY command sets the intensity level of the grid or the trace/text.</p> <p>The intensity level is expressed as a percentage (PCT). A level of 100 PCT corresponds to the maximum intensity whilst a level of 0 PCT sets the intensity to its minimum value.</p> <p>The response to the INTENSITY? query indicates the grid and trace intensity levels.</p>                                               |
| <b>COMMAND SYNTAX</b>  | <p><b>INTensity GRID, &lt;value&gt;, TRACE, &lt;value&gt;</b></p> <p>&lt;value&gt; : = 0 to 100 [PCT]</p> <p><i>Note 1: Parameters are grouped in pairs. The first of the pair names the variable to be modified, whilst the second gives the new value to be assigned. Pairs may be given in any order and be restricted to those variables to be changed.</i></p> <p><i>Note 2: The suffix PCT is optional.</i></p> |
| <b>QUERY SYNTAX</b>    | <b>INTensity?</b>                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>RESPONSE FORMAT</b> | <b>INTensity TRACE, &lt;value&gt;, GRID, &lt;value&gt;</b>                                                                                                                                                                                                                                                                                                                                                            |
| <b>EXAMPLE (GPIB)</b>  | <p>The following instruction enables remote control of the intensity, and changes the grid intensity level to 75%:</p> <p><b>CMD\$="INTS GRID, 75": CALL IBWRT(SCOPE%, CMD\$)</b></p>                                                                                                                                                                                                                                 |

**ACQUISITION****INTERLEAVED, ILVDG**  
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 *instrument Operator's Manual, Appendix A, for specifications*.

RIS is not available for sequence mode acquisitions.

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>

**G AVAILABILITY**

Query but not command available on model 9361C.

**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

| <b>STATUS</b>           | <b>*IST?</b><br>Query                                                                                                                                                                                                                  |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The *IST? (Individual SStatus) 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.                               |
| <b>QUERY SYNTAX</b>     | <b>*IST?</b>                                                                                                                                                                                                                           |
| <b>RESPONSE FORMAT</b>  | <b>*IST</b> <value><br><value> : = 0 or 1                                                                                                                                                                                              |
| <b>EXAMPLE (GPIB)</b>   | <p>The following instruction cause the contents of the IST bit to be read:</p> <pre><b>CMD\$="*IST?": CALL IBWRT(SCOPE%,CMD\$):</b> <b>CALL IBRD(SCOPE%,RSP\$): PRINT RSP\$</b></pre> <p>Response message</p> <pre><b>*IST 0</b></pre> |
| <b>RELATED COMMANDS</b> | *PRE                                                                                                                                                                                                                                   |

| DISPLAY          | KEY<br>Command                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DESCRIPTION      | <p>The KEY command allows control of a program from the front panel (<i>menus illustrated at right</i>). It displays strings of up to two lines of 13 characters as menus corresponding to and operated by the lower six menu buttons or soft keys (the top menu and button is automatically "GO TO LOCAL").</p> <p>String text assigned by the operator to these menus disappears on the next transition to local but reappears when the instrument is switched back into the remote state. Text is cleared at power-up, when the instrument is reset, or if an empty string is assigned to a location (for example: <b>KEY 2, ' '</b>).</p> <p>Pressing any one of the menu buttons while in remote mode causes the User Request status Register (URR) and the URQ bit of the Event Status Register to be set. This can generate an SRQ, provided that the service request mechanism has been enabled.</p> <div data-bbox="1214 306 1398 1031"> <p>REMOTE ENABLE</p> <p>GO TO LOCAL</p> <p>Pause</p> <p>Continue Measurement</p> <p>GO</p> </div> |
| COMMAND SYNTAX   | <p><b>KEY</b> &lt;button&gt;, '&lt;string&gt;', '&lt;string&gt;'</p> <p>&lt;button&gt; : = 1 to 5</p> <p>&lt;string&gt; : = Up to two 13-character strings (any ASCII code)</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| EXAMPLE (GPIB)   | <p>The menus illustrated this page were created by issuing the following instructions:</p> <pre> CMD\$="KEY 2, 'Pause'; KEY 3, 'Continue', 'Measurement'; KEY 4, ' ', 'GO': CALL IBWRT(SCOPE%,CMD\$) </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| RELATED COMMANDS | URR                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

**DISPLAY**

**LOGOG**  
Command/Query

**DESCRIPTION**                      The LOGO command controls the display of the LeCroy logo at the top left corner of the time grid or the XY grid.

**COMMAND SYNTAX**                **LOGO** <state>  
                                         <state> : = {ON, OFF}

**QUERY SYNTAX**                    **LOGO?**

**RESPONSE FORMAT**                **LOGO** <state>

**G AVAILABILITY**                    LC scopes only.

**EXAMPLE (GPIB)**                      The following instruction turns on the LeCroy logo:  
**CMD\$="LOGO ON": CALL IBWRT(SCOPE%,CMD\$)**

**CURSOR****MASK**  
Command/Query**DESCRIPTION**

For the PolyMask option

MASK COLOR allows you to select two colors: one for the mask and one for displaying circles around sample points outside the mask.

MASK DISP\_FILLED selects whether the mask is filled or not. During testing, the mask will always be filled, regardless of the state of this selection.

MASK DRAWTO draws a line from the current position to a new position. It is a command only.

MASK ERASE erases the current mask.

MASK FILL fills the enclosed polygon from starting position. It is a command only.

MASK MOVETO moves the cursor to a new position without drawing a line. It is a command only.

SHOW\_FAIL determines if errors inside or outside the mask should be circled.

**COMMAND SYNTAX**

<destination>: **MASK COLOR** <mask color>,<error color>

See the table of color choices under the COLOR command.

<destination>: **MASK DISP\_FILLED** <state>

<state> := {YES, NO}

<destination>: **MASK DRAWTO** <x\_value>,<y\_value>

<x\_value> := 0 to 10 divisions (–4 to +4 divisions for XY Plot)

<y\_value> := –4 to +4 divisions

<destination>: **MASK ERASE**

<destination>: **MASK FILL** <x\_value>,<y\_value>

<x\_value> := 0 to 10 divisions

<y\_value> := –4 to +4 divisions



<destination>: **MASK MOVETO**<x\_value>,<y\_value>  
<x\_value> := 0 to 10 divisions  
<y\_value> := -4 to +4 divisions  
<destination>: **SHOW\_FAIL** <state>,<count>  
<state> := {OFF, INSIDE, OUTSIDE}  
<count> := 1 to 1000  
<destination> := {C1, C2, C3, C4, TA, TB, TC, TD, TXY, PMXY}

## QUERY SYNTAX

**MASK? COLOR**  
**MASK? DISP\_FILLED**

## G AVAILABILITY

LC scopes only.

## EXAMPLE

The following instruction sets the mask color to blue and the color of the circles around outliers to red.

**TA:MASK COLOR,BLUE,RED**

## 9300 &amp; LC Series

**CURSOR****MATH\_LIMITS, MLIMG**  
Command/Query

|                         |                                                                                                                                                                                             |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | This command limits averaging to the area between horizontal relative cursors for increased throughput.                                                                                     |
| <b>COMMAND SYNTAX</b>   | <pre>&lt;source_header_pref&gt;: <b>Math_LIMITs</b> &lt;state&gt; &lt;state&gt; := {ON, OFF} &lt;source_header_pref&gt; := {TA, TB, TC, TD}</pre>                                           |
| <b>QUERY SYNTAX</b>     | <b>MLIM?</b>                                                                                                                                                                                |
| <b>RESPONSE FORMAT</b>  | <b>MATH_LIMIT</b> <state>                                                                                                                                                                   |
| <b>G AVAILABILITY</b>   | LC scopes only.                                                                                                                                                                             |
| <b>EXAMPLE (GPIB)</b>   | <p>The following instruction limits summation averaging to the data between the cursors:</p> <pre><b>CMD\$="MLIM ON": CALL IBWRT(SCOPE%,CMD\$) CALL IBRD (SCOPE%,RD\$):PRINT RD\$</b></pre> |
| <b>RESPONSE MESSAGE</b> | <b>MLIM OFF</b>                                                                                                                                                                             |
| <b>RELATED COMMANDS</b> | DEFINE, CURSOR_MEASURE                                                                                                                                                                      |

**DISPLAY**

**MEASURE\_GATE, MGATG**  
Command/Query

**DESCRIPTION**

The MEASURE\_GATE command is used to control whether or not the parameter measurement gate region (the region between the parameter cursors) is highlighted. Highlighting is performed by making the trace area outside the measurement gate region a neutral color.

The response to the MEASURE\_GATE? query indicates whether or not the parameter measurement gate region is highlighted.

**COMMAND SYNTAX**

**Measure\_GATE** <state>  
<state> : = {ON, OFF}

**QUERY SYNTAX**

**Measure\_GATE?**

**RESPONSE FORMAT**

**Measure\_GATE** <state>

**G AVAILABILITY**

Available only on color instruments.

**EXAMPLE (GPIB)**

The following instruction highlights the measurement gate region:

**CMD\$="MGAT ON": CALL IBWRT(SCOPE%,CMD\$)**

**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 *Appendix A of the oscilloscope Operator's Manual for memory size*.

Reducing the number of data points results in 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**

**Memory\_SIZE** <size>

<size> : = {**500, 1000, 2500, 5000, 10K, 25K, 50K, 100K, 250K, 500K, 1M, 2.5M, 5M, 10M**}

Or, alternatively, in standard numeric format

= {**500, 1e+3, ..., 2e+6, 4e+6, 8e+6**}, for example.

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

**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, COMB?

| <i>DISPLAY</i>         | <b>MESSAGE, MSG</b><br>Command/Query                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>     | <p>The MESSAGE command displays a string of characters in the Message Field above the grid. The string may be up to 49 characters in length. The string is displayed as long as the instrument is in remote mode and no internal status message is generated. Turning the oscilloscope back to local mode deletes the message. After the next transition from local to remote the message will be redisplayed. The message is cleared at power-up, when the instrument is reset, or if an empty string is sent (MSG "").</p> <p>The MESSAGE? query allows the user to read the last message sent.</p> |
| <b>COMMAND SYNTAX</b>  | <p><b>MeSsaGe</b> '&lt;string&gt;'</p> <p>&lt;string&gt; : = A string of a maximum of 49 characters</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>QUERY SYNTAX</b>    | <b>MeSsaGe?</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>RESPONSE FORMAT</b> | <b>MeSsaGe</b> "<string>"                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>EXAMPLE (GPIB)</b>  | <p>The following code causes the message "*Connect Probe 1*" to appear in the message field:</p> <pre><b>CMD\$="MSG '*Connect Probe 1*': CALL<br/>IBWRT(SCOPE%,CMD\$)</b></pre>                                                                                                                                                                                                                                                                                                                                                                                                                       |

**DISPLAY****MULTI\_ZOOM, MZOM**  
Command/Query**DESCRIPTION**

By setting MULTI\_ZOOM ON, the horizontal magnification and positioning controls apply to all expanded traces simultaneously. This command is useful if the contents of all expanded traces are to be examined at the same time.

The MULTI\_ZOOM? query indicates whether multiple zoom is enabled or not.

*Note: This command has the same effect as DUAL\_ZOOM.*

**COMMAND SYNTAX**

**Multi\_ZoOm** <mode>

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

**QUERY SYNTAX**

**Multi\_ZoOm?**

**RESPONSE FORMAT**

**Multi\_ZoOm** <mode>

**EXAMPLE (GPIB)**

The following example turns the multiple zoom on:

**CMD\$="MZOM ON": CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS**

HOR\_MAGNIFY, HOR\_POSITION, DUAL\_ZOOM

## 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 *oscilloscope Operator's Manual, Appendix A*, for specifications.

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

### COMMAND SYNTAX

<channel> : **OFFSet** <offset>

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

<offset> : = See *Appendix A of the instrument Operator's Manual* for specifications.

*Note: The suffix V is optional.*

### QUERY SYNTAX

<channel> : **OFFSet?**

### RESPONSE FORMAT

<channel> : **OFFSet** <offset>

## G AVAILABILITY

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

### EXAMPLE (GPIB)

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

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

## 9300 &amp; LC Series

**CURSOR****OFFSET\_CONSTANT, OFCTG**  
Command/Query

**DESCRIPTION** On gain changes, this command keeps the offset fixed, either volts or divisions.

**COMMAND SYNTAX** **Offset\_Constant** <mode>  
<mode> := {VOLTS, DIV}

**QUERY SYNTAX** **OFCT?**

**RESPONSE FORMAT** **OFCT VOLTS**

**G AVAILABILITY** LC scopes only.



| STATUS                  | *OPC<br>Command/Query                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <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 instrument 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> |
| <b>COMMAND SYNTAX</b>   | <b>*OPC</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>QUERY SYNTAX</b>     | <b>*OPC?</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>RESPONSE FORMAT</b>  | <b>*OPC 1</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>RELATED COMMANDS</b> | <b>*WAI</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

**MISCELLANEOUS****\*OPT?**  
Query

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>     | The *OPT? query identifies oscilloscope options, i.e. additional firmware or hardware options. The response consists of a series of response fields listing all the installed options.                                                                                                                                                                                                                                                                                |
| <b>QUERY SYNTAX</b>    | <b>*OPT?</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>RESPONSE FORMAT</b> | <p><b>*OPT</b> &lt;option_1&gt;, &lt;option_2&gt;, .., &lt;option_N&gt;<br/>         &lt;option_n&gt; := A three- or four-character ASCII string</p> <p><i>Note: If no option is present, the character 0 will be returned</i></p>                                                                                                                                                                                                                                    |
| <b>EXAMPLE (GPIB)</b>  | <p>The following queries the installed options:</p> <p><b>CMD\$="*OPT?": CALL IBWRT(SCOPE%, CMD\$):<br/>         CALL IBRD(SCOPE%, RSP\$): PRINT RSP\$</b></p> <p>If, for example, the waveform processing options WP01, WP02, WP03, DDM, CKIO, PRML, ORM, MC01 and JTA are installed, the response will be returned as:</p> <p><b>*OPT WP01, WP02, DDM, CKIO, PRML, ORM, MC01, JTA</b></p> <p>Response message if no options are installed:</p> <p><b>*OPT 0</b></p> |

## ADDITIONAL INFORMATION

| Notation    |                                       |
|-------------|---------------------------------------|
| <b>CKTR</b> | CKTRIG Clock-Trigger-Ext. ref. Option |
| <b>DDFA</b> | Disk Drive Failure Analysis Option    |
| <b>DDM</b>  | Disk Drive Measurements Option        |
| <b>FD01</b> | Floppy Disk Option                    |
| <b>GP01</b> | Internal Printer/Centronics Option    |
| <b>HD01</b> | Hard Disk Option                      |
| <b>JTA</b>  | Jitter and Timing Analysis Option     |
| <b>ORM</b>  | Optical Recording Measurements Option |
| <b>PMT</b>  | Power Measurement Tools               |
| <b>PRML</b> | PRML Measurements Option              |
| <b>MC01</b> | Memory Card Option                    |
| <b>WP01</b> | Waveform Processing Option WP01       |
| <b>WP02</b> | Waveform Processing Option WP02       |
| <b>WP03</b> | Waveform Processing Option WP03       |

**SAVE/RECALL SETUP****PANEL\_SETUP, PNSU**  
Command/Query**DESCRIPTION**

The PANEL\_SETUP command complements the \*SAV/\*RST commands. The PANEL\_SETUP command allows panel setups to be archived in encoded form on external storage media.

Only setup data read by the PNSU? query may be recalled into the oscilloscope. A panel setup error (see table on page 72) 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 instruction saves the instrument'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 instruction recalls the front-panel setup, stored previously in the file PANEL.SET, into the oscilloscope:

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

**RELATED COMMANDS**

\*RCL, \*SAV

**CURSOR**

**PARAMETER\_CLR, PACL  
Command**

**DESCRIPTION**

The `PARAMETER_CLR` command clears all the current parameters from the five-line list used in the Custom and Pass/Fail modes.

*Note: This command has the same effect as the command `PASS_FAIL_CONDITION`, given without any arguments.*

**COMMAND SYNTAX**

**`PParameter_Clear`**

**RELATED COMMANDS**

`PARAMETER_DELETE`, `PARAMETER_VALUE`,  
`PASS_FAIL_CONDITION`

## 9300 &amp; LC Series

**CURSOR****PARAMETER\_CUSTOM, PACU**  
Command/Query**DESCRIPTION**

The **PARAMETER\_CUSTOM** command controls the parameters that have customizable qualifiers, (for example, *Dt@lev* or *r@level*) and may also be used to assign any parameter for histogramming.

*Note: The measured value of a parameter setup with PACU may be read using PAVA?*

**COMMAND SYNTAX**

**Parameter\_Custom** <line>, <parameter>, <qualifier>[, <qualifier>, ...]

<line> : = 1 to 5

<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.

| <param>                                                                                  | definition                   | <qualifier> list                                                                  |
|------------------------------------------------------------------------------------------|------------------------------|-----------------------------------------------------------------------------------|
| <b>Parameters available on all models</b>                                                |                              |                                                                                   |
| <b>DC2DPOS</b>                                                                           | delta clock to data positive | <source1>,<clockedge>,<level1>,<source2>,<slope2>,<level2>,<hysteresis>           |
| <b>DC2DNEG</b>                                                                           | delta clock to data negative | <source1>,<clockedge>,<level1>,<source2>,<slope2>,<level2>,<hysteresis>           |
| <b>DDL</b>                                                                               | delta delay                  | <source1>,<source2>                                                               |
| <b>DTLEV</b>                                                                             | delta time at level          | <source1>,<slope1>,<level1>,<source2>,<slope2>,<level2>,<hysteresis>              |
| <b>FLEV</b>                                                                              | fall at level                | <source>,<high>,<low>                                                             |
| <b>PHASE</b>                                                                             | phase difference             | <source1>,<edge1>,<level1>,<source2>,<edge2>,<level2>,<hysteresis>,<angular unit> |
| <b>RLEV</b>                                                                              | rise at level                | <source>,<low>,<high>                                                             |
| <b>TLEV</b>                                                                              | time at level                | <source>,<slope>,<level>,<hysteresis>                                             |
| <b>Parameters available on instruments equipped with WP03 or DDM processing firmware</b> |                              |                                                                                   |
| <b>FWXX</b>                                                                              | full width at xx% of max     | <source>,<threshold>                                                              |
| <b>PCTL</b>                                                                              | percentile                   | <source>,<threshold>                                                              |
| <b>XAPK</b>                                                                              | x position at peak           | <source>,<rank>                                                                   |

| <param>                                                                          | definition                          | <qualifier> list                   |
|----------------------------------------------------------------------------------|-------------------------------------|------------------------------------|
| <b>Parameters available on instruments equipped with DDM processing firmware</b> |                                     |                                    |
| <b>LBASE</b>                                                                     | local base                          | <source>,<hysteresis>              |
| <b>LBSEP</b>                                                                     | local baseline separation           | <source>,<hysteresis>              |
| <b>LMAX</b>                                                                      | local maximum                       | <source>,<hysteresis>              |
| <b>LMIN</b>                                                                      | local minimum                       | <source>,<hysteresis>              |
| <b>LNUM</b>                                                                      | number of local events              | <source>,<hysteresis>              |
| <b>LPP</b>                                                                       | local peak to peak                  | <source>,<hysteresis>              |
| <b>LTBE</b>                                                                      | local time between events           | <source>,<hysteresis>              |
| <b>LTBP</b>                                                                      | local time between peaks            | <source>,<hysteresis>              |
| <b>LTBT</b>                                                                      | local time between troughs          | <source>,<hysteresis>              |
| <b>LTMN</b>                                                                      | local time at minima                | <source>,<hysteresis>              |
| <b>LTMX</b>                                                                      | local time at maxima                | <source>,<hysteresis>              |
| <b>LTOT</b>                                                                      | local time over threshold           | <source>,<hysteresis>,<threshold>  |
| <b>LTPT</b>                                                                      | local time peak to trough           | <source>,<hysteresis>              |
| <b>LTTP</b>                                                                      | local time trough to peak           | <source>,<hysteresis>              |
| <b>LTUT</b>                                                                      | local time under threshold          | <source>,<hysteresis>,<threshold>  |
| <b>NBPH</b>                                                                      | narrow band phase                   | <source>,<freq>                    |
| <b>NBPW</b>                                                                      | narrow band power                   | <source>,<freq>                    |
| <b>OWRITE</b>                                                                    | overwrite                           | <source 1>,<source 2>,<freq>       |
| <b>PW50</b>                                                                      | pulse width 50                      | <source>,<hysteresis>              |
| <b>PW50NEG</b>                                                                   | pulse width 50 for troughs          | <source>,<hysteresis>              |
| <b>PW50POS</b>                                                                   | pulse width 50 for peaks            | <source>,<hysteresis>              |
| <b>RES</b>                                                                       | resolution                          | <source 1>,<source 2>,<hysteresis> |
| <b>TAA</b>                                                                       | track average amplitude             | <source>,<hysteresis>              |
| <b>TAANEG</b>                                                                    | track average amplitude for troughs | <source>,<hysteresis>              |
| <b>TAAPOS</b>                                                                    | track average amplitude for peaks   | <source>,<hysteresis>              |

## 9300 &amp; LC Series

| <param>                                                                    | definition                       | <qualifier> list          |
|----------------------------------------------------------------------------|----------------------------------|---------------------------|
| Parameters available on instruments equipped with PRML processing firmware |                                  |                           |
| ACSN                                                                       | auto correlation signal to noise | <source>,<length>         |
| NLTS                                                                       | non-linear transition shift      | <source>,<length>,<delay> |

Where:

<sourceN> : = {**C1, C2, C3, C4, TA, TB, TC, TD**}

<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

<rank> : = 1 to 100

<threshold> : = 0 to 100 percent

<angular unit> = {**PCT, DEG, RAD**}

**QUERY SYNTAX**                      **PArameter\_CUstom?** <line>

**RESPONSE FORMAT**                **PArameter\_Custom** <line>,<parameter>,<qualifier>[,<qualifier>,...]

**G AVAILABILITY**                      <sourceN> : = {**C3, C4**} only on four-channel instruments.

**EXAMPLE 1**                              **DTLEV**

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

**Query/Response Examples**        **PACU? 2** returns:  
**PACU 2,DTLEV,C1,POS,345E-3,C2,NEG,-789E-3**  
**PAVA? CUST2** returns:  
**C2:PAVA CUST2,789 NS**



**EXAMPLE 2**

**DDL**

**Command Example**

**PACU 2, DDL, C1, C2**

**Query/Response Examples**

**PACU? 2** returns:  
**PACU 2, DDL, C1, C2**  
**PAVA? CUST2** returns:  
**C2: PAVA CUST2, 123 NS**

**EXAMPLE 3**

**RLEV**

**Command Example**

**PACU 3, RLEV, C1, 2PCT, 67PCT**

**Query/Response Examples**

**PACU? 3** returns:  
**PACU 3, RLEV, C1, 2PCT, 67PCT**  
**PAVA? CUST3** returns:  
**C1: PAVA CUST3, 23 MS**

**EXAMPLE 4**

**FLEV**

**Command Example**

**PACU 3, FLEV, C1, 345E-3, 122E-3**

**Query/Response Examples**

**PACU? 3** returns:  
**PACU 3, FLEV, C1, 345E-3, 122E-3**  
**PAVA? CUST3** returns:  
**C1: PAVA CUST3, 23 MS**

**RELATED COMMANDS**

**PARAMETER\_DELETE, PARAMETER\_VALUE,  
PASS\_FAIL\_CONDITION**

**CURSOR****PARAMETER\_DELETE, PADL  
Command****DESCRIPTION**

The **PARAMETER\_DELETE** command deletes a parameter at a specified line from the list of parameters used in the Custom and Pass/Fail modes.

| Notation |        |                                |
|----------|--------|--------------------------------|
| <b>1</b> | line 1 | of Custom or Pass/Fail display |
| <b>2</b> | line 2 | of Custom or Pass/Fail display |
| <b>3</b> | line 3 | of Custom or Pass/Fail display |
| <b>4</b> | line 4 | of Custom or Pass/Fail display |
| <b>5</b> | line 5 | of Custom or Pass/Fail display |

**COMMAND SYNTAX**

**PARameter\_DeLete** <line>

<line> : = {**1, 2, 3, 4, 5**}

*Note: This command has the same effect as the command **PASS\_FAIL\_CONDITION** <line>, given without any further arguments.*

**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**,  
**PASS\_FAIL\_CONDITION**

**CURSOR**
**PARAMETER\_STATISTICS?, PAST?**

Query

**DESCRIPTION**

The **PARAMETER\_STATISTICS?** query returns the current values of statistics for the specified pulse parameter mode and the result type, for all five lines of the pulse parameters display.

| Notation          |                                            |
|-------------------|--------------------------------------------|
| <b>AVG</b>        | average                                    |
| <b>CUST</b>       | custom parameters                          |
| <b>HIGH</b>       | highest value                              |
| <b>HPAR</b>       | horizontal standard parameters             |
| <b>LOW</b>        | lowest value                               |
| <b>PARAM</b>      | parameter definition for each line         |
| <b>NUM_ACQ</b>    | number of contributing acquisitions        |
| <b>NUM_VALUES</b> | number of measurements taken for parameter |
| <b>SIGMA</b>      | sigma (standard deviation)                 |
| <b>SWEEPS</b>     | number of sweeps accumulated for each line |
| <b>VPAR</b>       | vertical standard parameters               |

**QUERY SYNTAX**

**Parameter\_Statistics?** <mode>, <result>

<mode> : = {**CUST**, **HPAR**, **VPAR**}

<result> : = {**AVG**, **LOW**, **HIGH**, **NUM\_ACQ**, **NUM\_VALUES**, **SIGMA**, **SWEEPS**, **PARAM**}

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

**EXAMPLE (GPIB)**

The following query reads the average values of the five standard vertical parameters:

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

**RESPONSE FORMAT**

**PAST VPAR, AVG, 13V, 26V, 47V, 1V, 0V**

**RELATED COMMANDS**

**PARAMETER\_VALUE**

## 9300 &amp; LC Series

**CURSOR****PARAMETER\_VALUE?, PAVA?**

Query

**DESCRIPTION**

The **PARAMETER\_VALUE** query returns the current value(s) of the pulse waveform parameter(s) 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. For LC scopes, the command **PFDO PASS**, **TESTING\_OFF** turns off Pass/Fail testing. **Testing Off** is then displayed below the grid.

| Parameters Available on All Models                                           |                                                |               |                       |               |                       |
|------------------------------------------------------------------------------|------------------------------------------------|---------------|-----------------------|---------------|-----------------------|
| <b>ALL</b>                                                                   | all parameters                                 | <b>DUTY</b>   | duty cycle            | <b>OVSP</b>   | positive overshoot    |
| <b>AMPL</b>                                                                  | amplitude                                      | <b>FALL</b>   | falltime              | <b>PER</b>    | period                |
| <b>AREA</b>                                                                  | area                                           | <b>FALL82</b> | fall 80 to 20%        | <b>PKPK</b>   | peak-to-peak          |
| <b>BASE</b>                                                                  | base                                           | <b>FREQ</b>   | frequency             | <b>PNTS</b>   | points                |
| <b>CMEAN</b>                                                                 | mean for cyclic waveform                       | <b>FRST</b>   | first point           | <b>RISE</b>   | risetime              |
| <b>CMEDI</b>                                                                 | median for cyclic waveform                     | <b>LAST</b>   | last point            | <b>RISE28</b> | rise 20 to 80%        |
| <b>CRMS</b>                                                                  | root mean square for cyclic part of waveform   | <b>MAX</b>    | maximum               | <b>RMS</b>    | root mean square      |
| <b>CSDEV</b>                                                                 | standard deviation for cyclic part of waveform | <b>MEAN</b>   | mean                  | <b>SDEV</b>   | standard deviation    |
| <b>CYCL</b>                                                                  | cycles                                         | <b>MEDI</b>   | median value          | <b>TOP</b>    | top                   |
| <b>DLY</b>                                                                   | delay                                          | <b>MIN</b>    | minimum               | <b>WID</b>    | width                 |
| <b>DUR</b>                                                                   | duration of acquisition                        | <b>OVSN</b>   | negative overshoot    |               |                       |
| Custom Parameters Defined using <b>PARAMETER_CUSTOM</b> Command <sup>‡</sup> |                                                |               |                       |               |                       |
| <b>CUST1</b>                                                                 | <b>CUST2</b>                                   | <b>CUST3</b>  | <b>CUST4</b>          | <b>CUST5</b>  |                       |
| Parameters Available on Instruments with WP03 or DDM Processing Firmware     |                                                |               |                       |               |                       |
| <b>AVG</b>                                                                   | average of distribution                        | <b>HMEDI</b>  | median of a histogram | <b>PKS</b>    | number of peaks       |
| <b>DATA</b>                                                                  | data values                                    | <b>HRMS</b>   | histogram rms value   | <b>RANGE</b>  | range of distribution |
| <b>FWHM</b>                                                                  | full width at half max                         | <b>HTOP</b>   | histogram top value   | <b>SIGMA</b>  | sigma of distribution |
| <b>HAMPL</b>                                                                 | histogram amplitude                            | <b>LOW</b>    | low of distribution   | <b>TOTP</b>   | total population      |

<sup>‡</sup> The numbers in the terms **CUST1**, **CUST2**, **CUST3**, **CUST4** and **CUST5** refer to the line numbers of the selected custom parameters.

|                                     |                                                           |                 |                                                            |  |  |
|-------------------------------------|-----------------------------------------------------------|-----------------|------------------------------------------------------------|--|--|
| <b>HBASE</b>                        | histogram base                                            | <b>MAXP</b>     | maximum population                                         |  |  |
| <b>HIGH</b>                         | high of histogram                                         | <b>MODE</b>     | mode of distribution                                       |  |  |
| <b>Parameter Computation States</b> |                                                           |                 |                                                            |  |  |
| <b>AV</b>                           | averaged over several (up to 100) periods                 | <b>OF</b>       | signal partially in overflow                               |  |  |
| <b>GT</b>                           | greater than given value                                  | <b>OK</b>       | deemed to be determined without problem                    |  |  |
| <b>IV</b>                           | invalid value (insufficient data provided)                | <b>OU</b>       | signal partially in overflow and underflow                 |  |  |
| <b>LT</b>                           | less than given value                                     | <b>PT</b>       | window has been period truncated                           |  |  |
| <b>NP</b>                           | no pulse waveform                                         | <b>UF</b>       | signal partially in underflow                              |  |  |
| <b>Mask Test Names</b>              |                                                           |                 |                                                            |  |  |
| <b>ALL_IN</b>                       | all points of waveform inside mask (TRUE = 1, FALSE = 0)  | <b>SOME_IN</b>  | some points of waveform inside mask (TRUE = 1, FALSE = 0)  |  |  |
| <b>ALL_OUT</b>                      | all points of waveform outside mask (TRUE = 1, FALSE = 0) | <b>SOME_OUT</b> | some points of waveform outside mask (TRUE = 1, FALSE = 0) |  |  |

## QUERY SYNTAX

<trace> : **PARameter\_VALue?** [<parameter>,...,<parameter>]

<trace> : = {**TA, TB, TC, TD, C1, C2, C3, C4**}

<parameter> : = See table of parameter names on previous page.

### Alternative forms of query for mask tests:

<trace> : **PARameter\_VALue?** <old\_mask\_test>

<trace> : **PARameter\_VALue?** <mask\_test>, <mask>

<mask\_test> : = {**ALL\_IN, SOME\_IN, ALL\_OUT, SOME\_OUT**}

<old\_mask\_test> : = {**ALLI, ANYI, ALLO, ANYO**}

<mask> : = {**TA, TB, TC, TD**}

*Note: Old mask test keywords ALLI, ANYI, ALLO, ANYO imply testing of <trace> against the mask waveform TD. Old mask test keywords INSIDE and OUTSIDE are equivalent to ALL\_IN and SOME\_OUT; they are only supported for compatibility with older-model instruments.*

## 9300 &amp; LC Series

## RESPONSE FORMAT

<trace> : **P**ArAmeter\_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 the standard voltage and standard time parameters followed by their values and states are returned.*

**G** AVAILABILITY

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

## EXAMPLE (GPIB)

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

```
CMD$="TB:PAVA? RISE": CALL IBWRT(SCOPE%, CMD$):  
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

**CURSOR**

**PASS\_FAIL\_CONDITION, PFCO**  
Command/Query

**DESCRIPTION**

The PASS\_FAIL\_CONDITION command adds a Pass/Fail test condition or a custom parameter at the specified line on the Pass/Fail or Custom Parameter display.

The PASS\_FAIL\_CONDITION? query indicates the current Pass/Fail test setup or the current selection of custom parameters at the specified line.

*Note 1: Up to five test conditions (or custom parameters) can be specified at five different display lines on the screen. The command PASS\_FAIL\_CONDITION deals with one line at a time.*

| Notation  |              |           |            |
|-----------|--------------|-----------|------------|
| <b>GT</b> | greater than | <b>LT</b> | lower than |

**COMMAND SYNTAX**

**Pass\_Fail\_Condition** [<line>, <trace>, <parameter>[, <rel\_op> [, <ref\_value>]]]

<line> : = {1,2,3,4,5}

<trace> : = {TA, TB, TC, TD, C1, C2, C3G, C4G}

<parameter> : = See tables of parameter names on pages 120 and 126.

<rel\_op> : = {GT, LT}

<ref\_value> : = -1e15 to +1e15

*Note 2: The PFCO command with no arguments (i.e. "PFCO") deletes all conditions. The PFCO command with a single argument (i.e. "PFCO <line>") deletes the condition at <line>.*

*Note 3: Old mask test keywords ALLI and ANYO imply testing of <trace> against the mask waveform TD. Old mask test keywords INSIDE and OUTSIDE are equivalent to ALL\_IN and SOME\_OUT; they are only supported for compatibility with former versions.*

Alternative form of command for mask tests:

**Pass\_Fail\_Condition** [<line>, <trace>, <mask\_test>, <mask>]

<mask\_test> : = {ALL\_IN, SOME\_IN, ALL\_OUT, SOME\_OUT}

## 9300 &amp; LC Series

|                       |                                                                                                                                                                                                                                          |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| QUERY SYNTAX          | <p>&lt;mask&gt; : = {<b>TA</b>, <b>TB</b>, <b>TC</b>, <b>TD</b>}</p> <p><b>PFC0?</b> &lt;line&gt;</p>                                                                                                                                    |
| RESPONSE FORMAT       | <p><b>PFC0</b> &lt;line&gt;, &lt;trace&gt;, &lt;parameter&gt;, &lt;rel_op&gt;, &lt;ref_value&gt;</p> <p>Alternative form of response for mask tests:</p> <p><b>PFC0</b> &lt;line&gt;, &lt;trace&gt;, &lt;mask_test&gt;, &lt;mask&gt;</p> |
| <b>G</b> AVAILABILITY | <p>&lt;trace&gt; : = {C3, C4} only on four-channel instruments.</p>                                                                                                                                                                      |
| EXAMPLE (GPIB)        | <p>The following instruction sets the first test condition in the list to be "frequency on Channel 1 lower than 10 kHz":</p> <p><b>CMD\$="PFC0 1, C1, FREQ, LT, 10000":</b><br/> <b>CALL IBWRT(SCOPE%, CMD\$)</b></p>                    |
| RELATED COMMANDS      | <p>CURSOR_MEASURE, CURSOR_SET, PASS_FAIL_COUNTER,<br/> PASS_FAIL_DO, PASS_FAIL_MASK, PARAMETER_VALUE</p>                                                                                                                                 |



| <i>CURSOR</i>           | <b>PASS_FAIL_COUNTER, PFCT</b><br>Command/Query                                                                                               |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The PASS_FAIL_COUNTER command resets the Passed/Failed acquisitions counters. The PASS_FAIL_COUNTER? query returns the current counts.        |
| <b>COMMAND SYNTAX</b>   | <b>Pass_Fail_CounTer</b>                                                                                                                      |
| <b>QUERY SYNTAX</b>     | <b>Pass_Fail_CounTer?</b>                                                                                                                     |
| <b>RESPONSE FORMAT</b>  | <b>Pass_Fail_CounTer</b> <pass/fail>, <value>, <b>OF</b> , <value><br><value> : = 0 to 999999<br><pass/fail> : = { <b>PASS, FAIL</b> }        |
| <b>EXAMPLE (GPIB)</b>   | The following query reads the counters:<br><b>CMD\$="PFCT?": CALL IBWRT(SCOPE%, CMD\$)</b><br>Response message:<br><b>PFCT PASS, 8, OF, 9</b> |
| <b>RELATED COMMANDS</b> | CURSOR_MEASURE, CURSOR_SET, PASS_FAIL_DO,<br>PASS_FAIL_MASK, PARAMETER_VALUE                                                                  |

**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 by the oscilloscope after a Pass/Fail test. The PASS\_FAIL\_DO? query indicates which actions are currently selected.

| Notation    |                                     |
|-------------|-------------------------------------|
| <b>BEEP</b> | emit a beep                         |
| <b>PULS</b> | emit a pulse on the CAL connector   |
| <b>SCDP</b> | make a hard copy                    |
| <b>STO</b>  | store in memory or on storage media |
| <b>STOP</b> | stop acquisition                    |

**COMMAND SYNTAX**

**Pass\_Fail\_DO** [<outcome>[, <act>[, <act>...]]]

<outcome> : = {**PASS,FAIL**}

<act> : = {**STOP, SCDP, STO**}

*Note 1: The BEEP command is accepted only on models equipped with the CLBZ hardware option.*

*Note 2: The PULS command is accepted only on models equipped with the CKIO software option.*

*Note 3: The PFDO command with no arguments (i.e. "PFDO") deletes all actions.*

*Note 4: The STO command performs the store operation as described in the Waveform Store chapter in the Operator's Manual.*

*Note 5: After every pass or fail detected, the instrument sets the INR bit 12.*

**QUERY SYNTAX**

**Pass\_Fail\_DO?**

|                  |                                                                                                                                                                  |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RESPONSE FORMAT  | <b>Pass_Fail_DO</b> [<pass_fail>[, <act>[, <act>...]]]                                                                                                           |
| EXAMPLE (GPIB)   | <p>This following instruction forces the oscilloscope to stop acquiring when the test passes:</p> <p><b>CMD\$="PFDO PASS,STOP": CALL IBWRT(SCOPE%,CMD\$)</b></p> |
| RELATED COMMANDS | BUZZER, CURSOR_MEASURE, CURSOR_SET, INR, PARAMETER_VALUE, PASS_FAIL_COUNTER, PASS_FAIL_MASK                                                                      |

**CURSOR****PASS\_FAIL\_MASK, PFMS  
Command****DESCRIPTION**

The PASS\_FAIL\_MASK command generates a tolerance mask around a chosen trace and stores the mask in the selected memory.

**COMMAND SYNTAX**

**Pass\_Fail\_MaSk** [<trace>[, <htol>[, <vtol>[, <mask>]]]]  
 <trace> : = {TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, C3, C4}  
 <htol> : = 0.0 to 5.0  
 <vtol> : = 0.0 to 4.0  
 <mask> : = {M1, M2, M3, M4}

*Note: if any arguments are missing, the previous settings will be used.*

The alternative form of command:

**Pass\_Fail\_MaSk INVT** [, <mask>]

inverts the mask in the selected mask memory. If <mask> is missing, M4 is implied.

**G AVAILABILITY**

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

**EXAMPLE (GPIB)**

The following instruction generates a tolerance mask around the Channel 1 trace and stores it in M2:

```
CMD$="PASS_FAIL_MASK C1, 0.2, 0.3, M2":  
CALL IBWRT(SCOPE%, CMD$)
```

**RELATED COMMANDS**

PASS\_FAIL\_DO, PARAMETER\_VALUE

| <i>CURSOR</i>           | <b>PASS_FAIL_STATUS?, PFST?</b><br>Query                                                                                                  |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The PASS_FAIL_STATUS query returns the status of the pass/fail test for a given line number.                                              |
| <b>QUERY SYNTAX</b>     | <b>Pass_Fail_Status?</b> <line><br><line> := { <b>1, 2, 3, 4, 5</b> }                                                                     |
| <b>RESPONSE FORMAT</b>  | <b>Pass_Fail_Status</b> <line>, <state><br><state> := { <b>TRUE, FALSE</b> }                                                              |
| <b>EXAMPLE (GPIB)</b>   | The following queries the state of the pass/fail test condition specified for line 3.<br><b>CMD\$="PFST? 3": CALL IBWRT(SCOPE%,CMD\$)</b> |
| <b>RELATED COMMANDS</b> | PASS_FAIL_DO, PASS_FAIL_CONDITION,<br>PARAMETER_VALUE                                                                                     |

## 9300 &amp; LC Series

**ACQUISITION****PEAK\_DETECT, PDET**  
Command/Query**DESCRIPTION**

The PEAK\_DETECT command switches ON or OFF the peak detector built into the acquisition system.

The PEAK\_DETECT? query returns the current status of the peak detector.

**COMMAND SYNTAX**

**Peak\_DETect** <state>  
<state> : = {**ON**, **OFF**}

**QUERY SYNTAX**

**Peak\_DETect?**

**RESPONSE FORMAT**

**PDET** <state>

**G AVAILABILITY**

Available on 9350C, 9354C, 9370C, 9374C, 9384C, LC534A and LC574A models only.

**EXAMPLE (GPIB)**

The following instruction turns on the peak detector:

**CMD\$="PDET ON": CALL IBWRT(SCOPE%,CMD\$)**

## CURSOR

## PER\_CURSOR\_SET, PECS Command/Query

### DESCRIPTION

The PER\_CURSOR\_SET command allows you to position any one of the six independent cursors at a given screen location. The position of the cursor can be modified or queried even if the cursor is not currently displayed on the screen.

The PER\_CURSOR\_SET? query indicates the current position of the cursor(s).

The vertical cursor positions are the same as those controlled by the CURSOR\_SET command.

| Notation    |                       |             |                     |
|-------------|-----------------------|-------------|---------------------|
| <b>HABS</b> | horizontal absolute   | <b>VABS</b> | vertical absolute   |
| <b>HDIF</b> | horizontal difference | <b>VDIF</b> | vertical difference |
| <b>HREF</b> | horizontal reference  | <b>VREF</b> | vertical reference  |

### COMMAND SYNTAX

<trace> : **PER\_Cursor\_Set** <cursor>,  
<position>[, <cursor>, <position>, ..., <cursor>, <position>]

trace> := {**TA, TB, TC, TD, C1, C2, C3, C4**}

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

<position> := 0 to 10 DIV (horizontal), -29.5 to 29.5 DIV (vertical)

*Note 1: The suffix DIV is optional.*

*Note 2: Parameters are grouped in pairs. The first of the pair names the variable to be modified, whilst the second gives the new value to be assigned. Pairs may be in any order and be restricted to those variables to be changed.*

### QUERY SYNTAX

<trace> : **PER\_Cursor\_Set?** <cursor>[, <cursor>, ..., <cursor>]

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

*Note 3: 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.*

## 9300 &amp; LC Series

## RESPONSE FORMAT

**PER\_Cursor\_Set** <cursor>, <position>[, <cursor>, <position>, ..., <cursor>, <position>]

**G** AVAILABILITY

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

## EXAMPLE (GPIB)

The following code 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, 7.4DIV"**

## RELATED COMMANDS

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



## CURSOR

## PER\_CURSOR\_VALUE?, PECV?

Query

### DESCRIPTION

The PER\_CURSOR\_VALUE? query returns the values measured by the cursors specified below while in Persistence Mode.

| Notation    |                     |             |                   |
|-------------|---------------------|-------------|-------------------|
| <b>HABS</b> | horizontal absolute | <b>VABS</b> | vertical absolute |
| <b>HREL</b> | horizontal relative | <b>VREL</b> | vertical relative |

### QUERY SYNTAX

<trace> : **PER\_Cursor\_Value?** <cursor>[, <cursor>, ..., <cursor>]

<trace> : = {**TA, TB, TC, TD, C1, C2, C3, C4**}

<cursor> : = {**HABS, HREL, VABS, VREL, ALL**}

*Note: If <cursor> is not specified, ALL will be assumed.*

### RESPONSE FORMAT

<trace> : **PER\_Cursor\_Value** <cursor>,  
<value>[, <cursor>, <value>, ..., <cursor>, <value>]

### G AVAILABILITY

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

### EXAMPLE (GPIB)

The following code returns the value measured with the vertical relative cursor on Channel 1:

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

Response message:

```
C1:PECV VREL,56 MV
```

### RELATED COMMANDS

CURSOR\_MEASURE, PERSIST, PER\_CURSOR\_SET

## 9300 &amp; LC Series

**DISPLAY****PERSIST, PERS**  
Command/Query

|                         |                                                                                                          |
|-------------------------|----------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The PERSIST command enables or disables the persistence display mode.                                    |
| <b>COMMAND SYNTAX</b>   | <b>PERSist</b> <mode><br><mode> : = { <b>ON</b> , <b>OFF</b> }                                           |
| <b>QUERY SYNTAX</b>     | <b>PERSist?</b>                                                                                          |
| <b>RESPONSE FORMAT</b>  | <b>PERSist</b> <mode>                                                                                    |
| <b>EXAMPLE (GPIB)</b>   | The following code turns the persistence display ON:<br><b>CMD\$="PERS ON": CALL IBWRT(SCOPE%,CMD\$)</b> |
| <b>RELATED COMMANDS</b> | PERSIST_COLOR, PERSIST_LAST, PERSIST_SAT,<br>PERSIST_SETUP                                               |

**DISPLAY**

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

|                         |                                                                                                                                                                                                                                   |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>The PERSIST_COLOR command controls the color rendering method of persistence traces.</p> <p>The response to the PERSIST_COLOR? query indicates the color rendering method, Analog Persistence or Color Graded Persistence.</p> |
| <b>COMMAND SYNTAX</b>   | <p><b>Persist_Color</b> &lt;state&gt;<br/>           &lt;state&gt; : = {ANALOG, COLOR_GRADED}</p>                                                                                                                                 |
| <b>QUERY SYNTAX</b>     | <p><b>Persist_Color?</b></p>                                                                                                                                                                                                      |
| <b>RESPONSE FORMAT</b>  | <p><b>Persist_Color</b> &lt;state&gt;</p>                                                                                                                                                                                         |
| <b>G AVAILABILITY</b>   | <p>Available only on color instruments.</p>                                                                                                                                                                                       |
| <b>EXAMPLE (GPIB)</b>   | <p>The following instruction sets the persistence trace color to an intensity-graded range of the selected trace color:</p> <p><b>CMD\$="PECL ANALOG": CALL IBWRT(SCOPE%,CMD\$)</b></p>                                           |
| <b>RELATED COMMANDS</b> | <p>COLOR, COLOR_SCHEME, PERSIST, PERSIST_LAST, PERSIST_SAT, PERSIST_SETUP</p>                                                                                                                                                     |

## 9300 &amp; LC Series

**DISPLAY****PERSIST\_LAST, PELTG**  
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\_LastT** <state>  
<state> : = {**ON**, **OFF**}

**QUERY SYNTAX**

**Persist\_LastT?**

**RESPONSE FORMAT**

**Persist\_LastT** <state>

**G AVAILABILITY**

Available on color instruments only.

**EXAMPLE (GPIB)**

The following instruction 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, PESAG**  
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, TA, TB, TC, TD, ALL**}

<value> : = 0 to 100 PCT

*Note: The suffix PCT is optional.*

**QUERY SYNTAX**

**Persist\_Sat?**

**RESPONSE FORMAT**

**Persist\_Sat** <trace>, <value>

**G AVAILABILITY**

Available on color instruments only.

**EXAMPLE (GPIB)**

The following instruction sets the saturation level of the persistence data map for channel 3 to be 60%, i.e. 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

## 9300 &amp; LC Series

**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**

**P**ersist\_**S**etUp <time>,<mode>

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

<mode> := {**TOP2**, **ALL**}

**QUERY SYNTAX**

**P**ersist\_**S**etUp?

**RESPONSE FORMAT**

**P**ersist\_**S**etUp <time>,<mode>

**EXAMPLE (GPIB)**

The following instruction sets the variable persistence at 10 seconds on the top two traces:

**CMD\$="PESU 20,TOP2": CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS**

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

| <b>STATUS</b>           | <b>*PRE</b><br>Command/Query                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>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. The *PRE command allows the user to specify which bit(s) of the parallel poll register will affect the 'ist' individual status bit.</p> <p>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.</p> |
| <b>COMMAND SYNTAX</b>   | <p><b>PRE</b> &lt;value&gt;<br/>           &lt;value&gt; : = 0 to 65 535</p>                                                                                                                                                                                                                                                                                                                                                                                |
| <b>QUERY SYNTAX</b>     | <b>*PRE?</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>RESPONSE FORMAT</b>  | <b>*PRE</b> <value>                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>EXAMPLE (GPIB)</b>   | <p>The following instruction 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. This yields the PRE value 16.</p> <p><b>CMD\$="*PRE 16": CALL IBWRT(SCOPE%,CMD\$)</b></p>                                                                                                                                                                                                                           |
| <b>RELATED COMMANDS</b> | *IST                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**PROBES****PROBE\_CAL?, PRCA?**

Query

|                         |                                                                                                                                                                                                                                                                                 |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The PROBE_CAL? query performs a complete auto-calibration of a current probe connected to the instrument. At the end of this calibration, the response indicates how the calibration has terminated, and the instrument then returns to the state it was in prior to the query. |
| <b>QUERY SYNTAX</b>     | <channel> : <b>PROBE_CAL?</b>                                                                                                                                                                                                                                                   |
| <b>RESPONSE FORMAT</b>  | <b>PROBE_CAL</b> <diagnostics><br><diagnostics> : = 0 or 1<br>0 = Calibration successful                                                                                                                                                                                        |
| <b>EXAMPLE (GPIB)</b>   | The following instruction forces a self-calibration:<br><b>CMD\$="PROBE_CAL?": CALL IBWRT(SCOPE%, CMD\$):</b><br><b>CALL IBRD(SCOPE%, RD\$): PRINT RD\$</b><br>Response message (if no failure): <b>PROBE_CAL 0</b>                                                             |
| <b>RELATED COMMANDS</b> | AUTO_CALIBRATE, *CAL?, PROBE_DEGAUSS?                                                                                                                                                                                                                                           |



| <i><b>PROBES</b></i>    | <b>PROBE_DEGAUSS?, PRDG?</b><br>Query                                                                                                                                                                                                                                                                                                |
|-------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The PROBE_DEGAUSS? query performs the automatic degaussing of the current probe connected to the instrument. This eliminates core saturation by use of a backing current and application of an alternating field, reduced in amplitude over time from an initial high value. After the degaussing, a probe calibration is performed. |
| <b>QUERY SYNTAX</b>     | <channel> : <b>PROBE_DEGAUSS?</b>                                                                                                                                                                                                                                                                                                    |
| <b>RESPONSE FORMAT</b>  | <b>PROBE_DEGAUSS</b> <diagnostics><br><diagnostics> : = 0 or 1<br>0 = Degaussing and calibration successful                                                                                                                                                                                                                          |
| <b>EXAMPLE (GPIB)</b>   | The following instruction degausses and calibrates the connected probe:<br><br><b>CMD\$="PROBE_DEGAUSS?": CALL IBWRT(SCOPE%, CMD\$):<br/>CALL IBRD(SCOPE%, RD\$): PRINT RD\$</b><br><br>Response message (if no failure):<br><b>PROBE_DEGAUSS 0</b>                                                                                  |
| <b>RELATED COMMANDS</b> | PROBE_CAL?, PROBE_NAME?                                                                                                                                                                                                                                                                                                              |

## 9300 &amp; LC Series

| <i>PROBES</i>           | <b>PROBE_INFOTEXT?, PRIT?</b><br>Query                                                                                                                                                  |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The PROBE_INFOTEXT? Query returns informative text about a probe connected to your oscilloscope.                                                                                        |
| <b>QUERY SYNTAX</b>     | <channel> : <b>Probe_InfoText?</b><br><channel> := {C1, C2, C3, C4, EX, EX5}                                                                                                            |
| <b>RESPONSE FORMAT</b>  | <channel> : <b>PRIT</b> "<info>"                                                                                                                                                        |
| <b>G AVAILABILITY</b>   | LC scopes only.                                                                                                                                                                         |
| <b>EXAMPLE (GPIB)</b>   | The following instruction obtains the text for the probe in channel 1:<br><br><b>CMD\$="C1:PROBE_INFOTEXT?": CALL<br/>IBWRT(SCOPE%, CMD\$):<br/>CALL IBRD(SCOPE%, RD\$): PRINT RD\$</b> |
| <b>RELATED COMMANDS</b> | PROBE_CAL?, PROBE_NAME?                                                                                                                                                                 |

| <i>PROBES</i>           | PROBE_NAME?, PRNA?<br>Query                                                                                                                                                     |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The PROBE_NAME? query returns the name of a probe connected to the instrument. Passive probes are identified by their attenuation factor.                                       |
| <b>QUERY SYNTAX</b>     | <channel> : <b>PROBE_NAME?</b>                                                                                                                                                  |
| <b>RESPONSE FORMAT</b>  | <channel> : <b>PROBE_NAME</b> <probe name>                                                                                                                                      |
| <b>EXAMPLE (GPIB)</b>   | The following instruction obtains an identification of the connected probe:<br><br><b>CMD\$="PROBE_NAME?": CALL IBWRT(SCOPE%,CMD\$):<br/>CALL IBRD(SCOPE%,RD\$): PRINT RD\$</b> |
| <b>RELATED COMMANDS</b> | PROBE_CAL? PROBE_DEGAUSS?                                                                                                                                                       |

## 9300 &amp; LC Series

**MISCELLANEOUS****REAR\_OUTPUT, ROUTG**  
Command/Query

**DESCRIPTION** The REAR\_OUTPUT command is used to set the type of signal put out at the BNC connector. The query REAR\_OUTPUT? returns the current mode of the connector.

**COMMAND SYNTAX** **Rear\_OUTput** <mode>[,<level>[,<rate>]]  
 <mode> := { **OFF**, **PF**, **TRIG**, **TRDY**, **PULSE**}

**QUERY SYNTAX** **Rear\_OUTput?**

**RESPONSE FORMAT** **Rear\_OUTput** <mode>,<level>[,<rate>]

**G AVAILABILITY** LC scopes only.

**EXAMPLE (GPIB)** The following instruction turns off the BNC output:  
**CMD\$="ROUT OFF": CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS** PASS\_FAIL\_DO, CAL\_OUTPUT

**ADDITIONAL INFORMATION**

| NOTATION |                                        |
|----------|----------------------------------------|
| OFF      | Turns off rear output                  |
| PF       | Pass/Fail mode                         |
| PULSE    | Provides a single pulse                |
| TRIG     | Trigger Out mode                       |
| TRDY     | Trigger is ready for a new acquisition |

## SAVE/RECALL SETUP

**\*RCL**  
Command

### DESCRIPTION

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

The \*RCL command produces the opposite effect 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 the instrument setup previously stored in panel setup 3:

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

### RELATED COMMANDS

PANEL\_SETUP, \*SAV, EXR

**WAVEFORM TRANSFER****RECALL, REC  
Command****DESCRIPTION**

The RECALL command recalls a waveform file from the current directory on mass storage into any or all of the internal memories M1 to M4. *Note that only waveforms stored in BINARY format can be recalled.*

**COMMAND SYNTAX**

<memory> : **RECALL DISK**, <device>, FILE, ' <filename> '

<memory> : = {**M1**, **M2**, **M3**, **M4**, **ALL**}

<device> : = {**CARD**, **FLPY**, **HDD**}

<filename> : = An alphanumeric string of up to eight characters, followed by a dot and an extension of up to three digits.

**G AVAILABILITY**

<device> : CARD only available when MC01 Option is fitted.

<device> : HDD only available when HD01 Option is fitted.

**EXAMPLE (GPIB)**

The following recalls a waveform file called "SC1.001" from the memory card into Memory M1:

**CMD\$="M1:REC DISK,CARD,FILE,'SC1.001'": CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS**

STORE, INR?

## 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\_PaNe1 DISK, <device>, FILE, '<filename>'**  
                                          <device> := {**CARDG, FLPY, HDDG**}  
                                          <filename> := A string of up to eight characters, with the extension **".PNL"**.

**G AVAILABILITY**                      <device> : CARD only available when MC01 Option is fitted.  
                                          <device> : HDD only available when HD01 Option is fitted.

**EXAMPLE (GPIB)**                    The following recalls the front-panel setup from file P012.PNL on the floppy disk:  
                                          **CMD\$="RCPN DISK, FLPY, FILE, 'P012.PNL'":**  
                                          **CALL IBWRT(SCOPE%, CMD\$)**

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

**ACQUISITION****REFERENCE\_CLOCK, RCLK**  
**Command/Query**

|                        |                                                                                                                                               |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>     | The REFERENCE_CLOCK command selects the system clock source, allowing the instrument to be phase synchronized to an external reference clock. |
| <b>COMMAND SYNTAX</b>  | <b>Reference_Clock</b> <mode><br><mode> : = {INT, EXT}                                                                                        |
| <b>QUERY SYNTAX</b>    | <b>Reference_Clock?</b>                                                                                                                       |
| <b>RESPONSE FORMAT</b> | <b>RCLK</b> <mode>                                                                                                                            |



## SAVE/RECALL SETUP

**\*RST**  
Command

|                         |                                                                                                                          |
|-------------------------|--------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The *RST command initiates a device reset. The *RST sets all eight traces to the GND line and recalls the default setup. |
| <b>COMMAND SYNTAX</b>   | <b>*RST</b>                                                                                                              |
| <b>EXAMPLE (GPIB)</b>   | This example resets the oscilloscope:<br><b>CMD\$="*RST": CALL IBWRT(SCOPE%,CMD\$)</b>                                   |
| <b>RELATED COMMANDS</b> | *CAL, *RCL                                                                                                               |

**ACQUISITION****SAMPLE\_CLOCK, SCLKG**  
Command/Query**DESCRIPTION**

The SAMPLE\_CLOCK command allows the user to control an external timebase. The user sets the number of data points that will be acquired when the instrument is using the external clock. When the optional suffix NUM is used with the query, the response will be returned in standard numeric format.

**COMMAND SYNTAX**

**Sample\_Clock** <state>[, <recordlength>][, <coupling>]

<state> : = {INT, ECL, LV0, TTL, RP<sub>G</sub>}

<recordlength> : = {50, 100, 200, 500, 1K, 2K, 5K, 10K, 20K, 50K, 100K, 200K, 500K, 1M, 2M}

Or, alternatively, in standard numeric format:

= {10e+3, 10.0e+3, 11e+3...}, for example.

<coupling> : = {D1M or D50<sub>G</sub>}

*Note: The record length cannot be larger than the maximum available memory of the model being used. The instrument will adapt to the closest valid <recordlength>. See Appendix A of the instrument Operator's Manual for maximums.*

**QUERY SYNTAX**

**Sample\_Clock?** [NUM]

**G AVAILABILITY**

Not available on 9361C or 9362C Series.

<state> : {RP} only available on oscilloscopes fitted with the CKTRIG option.

<coupling> : {D50} not available when <state> {RP} is selected.

**RESPONSE FORMAT**

**Sample\_Clock** <state>, <recordlength>

**EXAMPLE**

The following sets the instrument to use the external clock with 1000 data point records.

**CMD\$="SCLK ECL,1000": CALL IBWRT(SCOPE%,CMD\$)**

## SAVE/RECALL SETUP

**\*SAV**  
Command

### DESCRIPTION

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

*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 (\*SRE, \*PRE, \*ESE, INE) are not saved by this command.*

### 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

## 9300 &amp; LC Series

**DISPLAY****SCREEN**  
Command**DESCRIPTION**

The SCREEN command turns the screen on or off independently of the SCREEN\_SAVE command.

**NOTE:** When the screen is off, the oscilloscope is still full functional

**COMMAND SYNTAX**

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

**G AVAILABILITY**

LC scopes only.

**HARD COPY**

**SCREEN\_DUMP, SCDP**  
Command/Query

|                         |                                                                                                                                                                                                                                                                                        |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>The SCREEN_DUMP command causes the oscilloscope to dump the screen contents onto the hard-copy device. This command will halt the instrument's activities.</p> <p>The time/date stamp which appears on the print-out corresponds to the time at which the command was executed.</p> |
| <b>COMMAND SYNTAX</b>   | <b>Screen_Dump</b>                                                                                                                                                                                                                                                                     |
| <b>QUERY SYNTAX</b>     | <b>Screen_Dump?</b>                                                                                                                                                                                                                                                                    |
| <b>RESPONSE FORMAT</b>  | <p><b>Screen_Dump</b> &lt;status&gt;</p> <p>&lt;status&gt; : = {<b>OFF</b>}</p>                                                                                                                                                                                                        |
| <b>EXAMPLE (GPIB)</b>   | <p>The following initiates a screen dump:</p> <p><b>CMD\$="SCDP": CALL IBWRT(SCOPE%, CMD\$)</b></p>                                                                                                                                                                                    |
| <b>RELATED COMMANDS</b> | INR, HARDCOPY_SETUP, HARDCOPY_TRANSMIT                                                                                                                                                                                                                                                 |

## 9300 &amp; LC Series

**DISPLAY****SCREEN\_SAVE, SCSVG**  
Command/Query**DESCRIPTION**

The SCREEN\_SAVE command controls the automatic Screen Saver, which automatically shuts down the internal color monitor after a preset time.

The response to the SCREEN\_SAVE? query indicates whether the automatic screen saver feature is on or off.

*Note: When the screen save is in effect, the oscilloscope is still fully functional.*

**COMMAND SYNTAX**

**Screen\_Save** <enabled>  
<enabled> : = {**YES**, **NO**}

**QUERY SYNTAX**

**Screen\_Save?**

**RESPONSE FORMAT**

**Screen\_Save** <state>

**G AVAILABILITY**

Available only on color models.

**EXAMPLE (GPIB)**

The following enables the automatic screen saver:

**CMD\$="SCSV YES": CALL IBWRT(SCOPE%,CMD\$)**

| <i>DISPLAY</i>          | <b>SELECT, SEL</b><br>Command/Query                                                                                                                                                                                                                                                    |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>The <b>SELECT</b> command selects the specified trace for manual display control. An environment error (see <i>table on page 72</i>) is generated if the specified trace is not displayed.</p> <p>The <b>SELECT?</b> query returns the selection status of the specified trace.</p> |
| <b>COMMAND SYNTAX</b>   | <p>&lt;trace&gt; : <b>SElect</b></p> <p>&lt;trace&gt; : = {<b>TA</b>, <b>TB</b>, <b>TC</b>, <b>TD</b>}</p>                                                                                                                                                                             |
| <b>QUERY SYNTAX</b>     | <p>&lt;trace&gt; : <b>SElect?</b></p>                                                                                                                                                                                                                                                  |
| <b>RESPONSE FORMAT</b>  | <p>&lt;trace&gt; : <b>SElect</b> &lt;mode&gt;</p> <p>&lt;mode&gt; : = {<b>ON</b>, <b>OFF</b>}</p>                                                                                                                                                                                      |
| <b>EXAMPLE (GPIB)</b>   | <p>The following selects Trace B (TB):</p> <p><b>CMD\$="TB:SEL": CALL IBWRT(SCOPE%,CMD\$)</b></p>                                                                                                                                                                                      |
| <b>RELATED COMMANDS</b> | <p>TRACE</p>                                                                                                                                                                                                                                                                           |

**ACQUISITION****SEQUENCE, SEQG**  
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.

**COMMAND SYNTAX**

**SEQUENCE** <mode>[, <segments>[, <max\_size>]]

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

<segments> := the right-hand column in the table below:

| Max. memory length per channel | Max. number of segments |
|--------------------------------|-------------------------|
| 10 k                           | 50                      |
| 25 k                           | 50                      |
| 50 k                           | 200                     |
| 100 k                          | 500                     |
| 200 k                          | 500                     |
| 250 k                          | 500                     |
| 500 k                          | 2000                    |
| 1 M                            | 2000                    |
| 2 M                            | 2000                    |

<max\_size> := {50, 100, 250, 500, 1000, 2500, 5K, 10K, 25K, 50K, 100K, 250K, 500K, 1M}

Or, alternatively, in standard numeric format:

= {...10e+3, 10.0e+3, ...11e+3, ...}, for example.

*Note: The instrument 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**}

**G AVAILABILITY**

Not available on 9361C or 9362C Series.

**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

**MISCELLANEOUS****SLEEP**  
**Command****DESCRIPTION**

The SLEEP command makes the scope's remote command interpreter wait the time specified in the argument before interpreting any remote commands that follow it. Typically it is used to let an external signal settle before the scope performs new acquisitions and processing defined by the remote commands that follow.

**COMMAND SYNTAX**

**SLEEP** <n>  
<n> : = {0 to 1000.0 s}

**G AVAILABILITY**

LC scopes only.

**EXAMPLE (GPIB)**

The following instruction causes the scope to sleep for 3.2 seconds:

**CMD\$="SLEEP 3.2": CALL IBWRT(SCOPE%,CMD\$)**

**RELATED COMMANDS**

\*TRG, WAIT

**STATUS****\*SRE**  
Command/Query**DESCRIPTION**

The \*SRE command sets the Service Request Enable register (SRE). This command allows the user to specify which summary message bit(s) in the STB register will generate a service request. *Refer to the table on page 167 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 instruction 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 status register (STB), and the Master Summary Status (MSS). The response represents the values of bits 0 to 5 and 7 of the Status Byte 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 167 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

## 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<br>MSS=1<br>RQS=1 | at least 1 bit in STB masked by SRE is 1<br>service is requested | (1)<br>(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)        |

### Notes

- (1) The Master Summary Status (MSS) indicates that the instrument requests service, whilst 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.

| <i>ACQUISITION</i>      |                                                                                                                                                                             | <b>STOP</b><br>Command |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------|
| <b>DESCRIPTION</b>      | The STOP command immediately stops the acquisition of a signal. If the trigger mode is AUTO or NORM, it will change to trigger mode STOPPED to prevent further acquisition. |                        |
| <b>COMMAND SYNTAX</b>   | <b>STOP</b>                                                                                                                                                                 |                        |
| <b>EXAMPLE</b>          | The following stops the acquisition process:<br><b>CMD\$ ="STOP": CALL IBWRT(SCOPE%,CMD\$)</b>                                                                              |                        |
| <b>RELATED COMMANDS</b> | ARM_ACQUISITION, TRIG_MODE, WAIT                                                                                                                                            |                        |

## 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> := {TA, TB, TC, TD, C1, C2, C3<sup>G</sup>, C4<sup>G</sup>, ALL\_DISPLAYED}  
                                          <dest> := {M1, M2, M3, M4, CARD<sup>G</sup>, FLPY<sup>G</sup>, HDD<sup>G</sup>}  
*Note: If the STORE command is sent without any argument, all traces currently enabled in the Store Setup will be stored. This setup can be modified using the STORE\_SETUP command.*

**G AVAILABILITY**                    <trace> := {C3, C4} only available on four-channel oscilloscopes.  
                                          <dest> : CARD only available when MC01 option is fitted.  
                                          <dest> : HDD only available when HD01 option is fitted.

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

**CMD\$="STO TA,M1": CALL IBWRT(SCOPE%,CMD\$)**

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

**CMD\$="STO ALL\_DISPLAYED, CARD":  
CALL IBWRT(SCOPE%,CMD\$)**

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

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

**RELATED COMMANDS**               STORE\_SETUP, RECALL

**SAVE/RECALL SETUP****STORE\_PANEL, STPN**  
Command**DESCRIPTION**

The STORE\_PANEL command stores the complete front-panel setup of the instrument, 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 (\*SRE, \*PRE, \*ESE, INE) are not saved by this command.*

**COMMAND SYNTAX**

**Store\_PaNe1 DISK, <device>, FILE, '<filename>'**

<device> : = {CARDG, FLPY, HDDG}

<filename> : = A string of up to 8 characters, with the extension ".PNL".

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

**G AVAILABILITY**

<device> : CARD only available when MC01 option is fitted.

<device> : HDD only available when HD01 option is fitted.

**EXAMPLE (GPIB)**

The following code saves the current instrument setup to the memory card in a file called "DIODE.PNL":

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

**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. A single trace or all displayed traces may be enabled for storage. This applies to auto-storing or to the STORE, STO command. Traces may be auto-stored to mass storage after each acquisition until the mass storage device becomes full (FILL), or continuously (WRAP), replacing the oldest traces by new ones.

The STORE\_SETUP? query returns the current mode of operation of Autostore, the current trace selection, and the current destination.

*Note that only waveforms stored in BINARY format can be recalled.*

### COMMAND SYNTAX

**STore\_SetUp** [<trace>, <dest>] [, **AUTO**, <mode>] [, **FORMAT**, <type>]

<trace> := {**TA**, **TB**, **TC**, **TD**, **C1**, **C2**, **C3****G**, **C4****G**,  
**ALL\_DISPLAYED**}

<dest> := {**M1**, **M2**, **M3**, **M4**, **CARD****G**, **FLPY**, **HDD****G**}

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

<type> : {**BINARY**, **SPREADSHEET**, **MATHCAD**, **MATLAB**}

### QUERY SYNTAX

**STore\_SetUp?**

### RESPONSE FORMAT

**STore\_SetUp** <trace>, <dest>, **AUTO**, <mode>

### **G** AVAILABILITY

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

<dest> : **CARD** only available when MC01 option is fitted.

<dest> : **HDD** only available when HD01 option is fitted.

### EXAMPLE (GPIB)

The following command selects Channel 1 to be stored. It enables an "autostore" to the card until no more space is left on the memory card (AUTO, FILL).

**CMD\$="STST C1, CARD, AUTO, FILL":**  
**CALL IBWRT(SCOPE%, CMD\$)**

### RELATED COMMANDS

STORE, INR

**WAVEFORM TRANSFER****STORE\_TEMPLATE, STTM  
Command****DESCRIPTION**

The STORE\_TEMPLATE command stores the instrument's waveform template on a mass-storage device. A filename is automatically generated in the form of "LECROYvv.TPL" where "vv" is the two-digit revision number.

*Note: For revision 2.1, for example, the file name generated will be LECROY21.TPL.*

*Refer to Chapter 4 for more on the waveform template, and Appendix B for a copy of the template itself.*

**COMMAND SYNTAX**

**Store\_TeMplate DISK, <device>**

<device> : = {CARD<sup>G</sup>, FLPY, HDD<sup>G</sup>}

**G AVAILABILITY**

<device> : CARD only available when MC01 option is fitted.

<device> : HDD only available when HD01 option is fitted.

**EXAMPLE (GPIB)**

The following code stores the current waveform template on the memory card for future reference:

**CMD\$="STTM DISK, CARD": CALL IBWRT(SCOPE%, CMD\$)**

**RELATED COMMANDS**

TEMPLATE

| MISCELLANEOUS          | TDISP<br>Command/Query                                                                                                                                                                                                                                                                                     |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>     | The TSDP command controls the presence and format of the time and date display on the oscilloscope screen. Keyword CURRENT selects the current time and date for display. Keyword NONE removes both time and date. Keyword TRIGGER selects the trigger time of the uppermost waveform currently displayed. |
| <b>COMMAND SYNTAX</b>  | <b>TDISP</b> <mode><br><mode> := {CURRENT, NONE, TRIGGER}                                                                                                                                                                                                                                                  |
| <b>QUERY SYNTAX</b>    | <b>TDISP?</b>                                                                                                                                                                                                                                                                                              |
| <b>RESPONSE FORMAT</b> | <b>TDISP</b> <mode>                                                                                                                                                                                                                                                                                        |
| <b>EXAMPLE (GPIB)</b>  | The following instruction turns off the time and date display on the oscilloscope screen:<br><b>CMD\$="TDISP NONE": CALL IBWRT(SCOPE%,CMD\$)</b>                                                                                                                                                           |

**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. *Refer to Chapter 4 for more on the waveform template, and Appendix B 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<br>Command/Query                                                                                                                                                                                                                                                                                                                                                                         |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>The TIME_DIV command modifies the timebase setting. The new timebase setting may 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 <i>table on page 167</i>) to be set.</p> <p>The TIME_DIV? query returns the current timebase setting.</p> |
| <b>COMMAND SYNTAX</b>   | <p><b>Time_DIV</b> &lt;value&gt;</p> <p>&lt;value&gt; := See <i>Appendix A of the instrument Operator's Manual for specifications.</i></p> <p><i>Note: The suffix S (seconds) is optional.</i></p>                                                                                                                                                                                                      |
| <b>QUERY SYNTAX</b>     | <b>Time_DIV?</b>                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>RESPONSE FORMAT</b>  | <b>Time_DIV</b> <value>                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>EXAMPLE (GPIB)</b>   | <p>The following sets the time base to 500 <math>\mu</math>s/div:</p> <p><b>CMD\$="TDIV 500US": CALL IBWRT(SCOPE%,CMD\$)</b></p> <p>The following sets the time base to 2 ms/div:</p> <p><b>CMD\$="TDIV 0.002": CALL IBWRT(SCOPE%,CMD\$)</b></p>                                                                                                                                                        |
| <b>RELATED COMMANDS</b> | 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 72*) 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> : **TRAc**e <mode>

<trace> : = {**C1**, **C2**, **C3G**, **C4G**, **TA**, **TB**, **TC**, **TD**}

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

**QUERY SYNTAX**

<trace> : **TRAc**e?

**RESPONSE FORMAT**

<trace> : **TRAc**e <mode>

**G AVAILABILITY**

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

**EXAMPLE (GPIB)**

The following command displays Trace A (TA):

**CMD\$="TA:TRA ON": CALL IBWRT(SCOPE%,CMD\$)**

|                |                                           |
|----------------|-------------------------------------------|
| <b>DISPLAY</b> | <b>TRACE_LABEL, TRLB</b><br>Command/Query |
|----------------|-------------------------------------------|

|                        |                                                                                                                                                                    |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>     | The TRACE_LABEL command allows you to enter a label for a trace. The label is displayed in the channel descriptor field on the left side of the display.           |
| <b>COMMAND SYNTAX</b>  | <code>&lt;channel&gt;: Trace_LaBe1 "&lt;text&gt;"</code><br><code>&lt;channel&gt; : = {C1, C2, C3, C4}</code>                                                      |
| <b>QUERY SYNTAX</b>    | <code>&lt;channel&gt;: TRLB?</code>                                                                                                                                |
| <b>RESPONSE FORMAT</b> | <code>&lt;channel&gt;: TRACE_LABEL "&lt;text&gt;"</code>                                                                                                           |
| <b>EXAMPLE (GPIB)</b>  | <p>The following instruction sets the trace for channel 2 to "headsig".</p> <p><b>CMD\$="C2:TRACE_LABEL 'HEADSIG'":</b></p> <p><b>CALL IBWRT(SCOPE%,CMD\$)</b></p> |

**DISPLAY****TRACE\_OPACITY, TOPA**  
**G**  
**Command/Query****DESCRIPTION**

The TRACE\_OPACITY command controls the opacity and the transparency of the trace color. The trace can be made either opaque (traces will overwrite and obscure each other) or transparent (overlapping traces can be distinguished from one another).

The response to the TRACE\_OPACITY? query indicates whether the traces are opaque or transparent.

**COMMAND SYNTAX**

**Trace\_OPAcity** <type>  
 <type> := {**OPAQUE**, **TRANSPARENT**}

**QUERY SYNTAX**

**Trace\_OPAcity?**

**RESPONSE FORMAT**

**Trace\_OPAcity** <type>

**G** **AVAILABILITY**

Available on color instruments only.

**EXAMPLE (GPIB)**

The following allows traces to be distinguished even when they overlap:

**CMD\$="TOPA TRANSPARENT": CALL BWRT(SCOPE%,CMD\$)**



| <i>DISPLAY</i>          | <b>TRACE_ORDER, TORD</b><br>Command/Query                                                                              |
|-------------------------|------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The TRACE_ORDER command allows you to specify the display order of traces.                                             |
| <b>COMMAND SYNTAX</b>   | <b>Trace_ORDER</b> <slot><trace><br><slot> : = {1, 2, 3, 4, 5, 6, 7, 8}                                                |
| <b>QUERY SYNTAX</b>     | <b>TORD?</b>                                                                                                           |
| <b>RESPONSE FORMAT</b>  | <b>TORD 1,C1,2,OFF,3,OFF,4,OFF,<br/>5,OFF,6,OFF,7,OFF,8,OFF</b>                                                        |
| <b>EXAMPLE (GPIB)</b>   | The following instruction displays Trace A above channel 1:<br><b>CMD\$="TORD 1,TA,2,C1": CALL IBWRT(SCOPE%,CMD\$)</b> |
| <b>RELATED COMMANDS</b> | TRACE                                                                                                                  |

**WAVEFORM STORAGE****TRANSFER\_FILE, TRFL**  
Command/Query**DESCRIPTION**

The TRANSFER\_FILE 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> := {CARD, FLPY, HDD}

Note: HDD is available only if the HD01 option is present

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

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

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

**QUERY SYNTAX**

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

**RESPONSE FORMAT**

**TRFL** #9nnnnnnnnnnnn<file content><crc>

**G AVAILABILITY**

LC scopes only.

**EXAMPLE (GPIB)**

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

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

**CALL IBWRT(SCOPE%, CMD\$)**

**RELATED COMMANDS**

DIRECTORY

| ACQUISITION      |                                                                                                                                                           | *TRG<br>Command |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|
| DESCRIPTION      | <p>The *TRG command executes an ARM command.</p> <p><i>Note: The *TRG command is the equivalent of the 488.1 GET (Group Execute Trigger) message.</i></p> |                 |
| COMMAND SYNTAX   | <b>*TRG</b>                                                                                                                                               |                 |
| EXAMPLE (GPIB)   | <p>The following command enables signal acquisition:</p> <p><b>CMD\$="*TRG": CALL IBWRT(SCOPE%, CMD\$)</b></p>                                            |                 |
| RELATED COMMANDS | ARM_ACQUISITION, STOP, WAIT                                                                                                                               |                 |

**ACQUISITION****TRIG\_COUPLING, TRCPG**  
Command/Query**DESCRIPTION**

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

*Note 1: The trigger slope is automatically determined by the instrument when in HFDIV coupling. The TRIG\_SLOPE command is not used in HFDIV coupling mode.*

*Note 2: HFDIV is indicated as HF in the trigger setup menus.*

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

**COMMAND SYNTAX**

<trig\_source> : **TRig\_CouPling** <trig\_coupling>

<trig\_source> : = {**C1**, **C2**, **c3G**, **c4G**, **EX**, **EX10G**, **EX5G**}

<trig\_coupling> : = {**ACG**, **DC**, **HFREJG**, **LFREJG**, **AUTOG**}

**QUERY SYNTAX**

<trig\_source> : **TRig\_CouPling?**

**RESPONSE FORMAT**

<trig\_source> : **TRig\_CouPling** <trig\_coupling>

**G AVAILABILITY**

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

<trig\_source> : = EXT10 not on LC564 or LC584 Series models.

<trig\_source> : = EXT5 only on LC564 and LC584 Series models.

<trig\_coupling> : AUTO only available on model 9362C.

<trig\_coupling> : AC, HFREJ, LFREJ not available on 9362C.

**EXAMPLE (GPIB)**

The following command 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\_LEVEL\_2, TRIG\_MODE, TRIG\_SELECT, TRIG\_SLOPE,  
TRIG\_WINDOW

| ACQUISITION             | TRIG_DELAY, TRDL<br>Command/Query                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>The TRIG_DELAY command sets the time at which the trigger is to occur with respect to the first acquired data point (displayed at the left-hand edge of the screen).</p> <p>The command expects positive trigger delays 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.</p> <p>If a value outside the range <math>-10\ 000\ \text{div} \cdot \text{time/div}</math> 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.</p> <p>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.</p> |
| <b>COMMAND SYNTAX</b>   | <p><b>TRig_DeLay</b> &lt;value&gt;<br/>           &lt;value&gt; : = 0.00 PCT to 100.00 PCT (pretrigger)<br/>                     -20 PS to -10 MAS (post-trigger)</p> <p><i>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.</i></p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>QUERY SYNTAX</b>     | <b>TRig_DeLay?</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>RESPONSE FORMAT</b>  | <b>TRig_DeLay</b> <value>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>EXAMPLE (GPIB)</b>   | <p>The following command sets the trigger delay to -20 s (post-trigger):</p> <p><b>CMD\$="TRDL -20S": CALL IBWRT(SCOPE%,CMD\$)</b></p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>RELATED COMMANDS</b> | TIME_DIV, TRIG_COUPLING, TRIG_LEVEL, TRIG_LEVEL_2, TRIG_MODE, TRIG_SELECT, TRIG_SLOPE, TRIG_WINDOW                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

**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, C3G, C4G, EX, EX10G, EX5G**}

<trig\_level> : = See *instrument Operator's Manual, Appendix A*, for specifications.

*Note: The suffix V is optional.*

**QUERY SYNTAX**

<trig\_source> : **TRig\_LeVel?**

**RESPONSE FORMAT**

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

**G AVAILABILITY**

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

<trig\_source> : = EXT10 not on LC564 or LC584 Series models.

<trig\_source> : = EXT5 only on LC564 and LC584 Series models.

**EXAMPLE (GPIB)**

The following code 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\_LEVEL\_2,  
TRIG\_MODE, TRIG\_SELECT, TRIG\_SLOPE, TRIG\_WINDOW

## ACQUISITION

## TRIG\_LEVEL\_2, TRLV2G Command/Query

### DESCRIPTION

The TRIG\_LEVEL\_2 command adjusts the level of the second trigger threshold reference, used in advanced SMART Trigger modes such as Runt and Slew Rate.

The response to the TRIG\_LEVEL\_2? query returns the current level of the second trigger.

### COMMAND SYNTAX

<trig\_source> : **Trigger\_LeVel\_2** <trig\_level>

<trig\_source> : = {**C1, C2, C3, C4, 2, EX, EX5**}

<trig\_level> := See *Operator's Manual, Appendix A, for specifications.*

### QUERY SYNTAX

**Trigger\_LeVel\_2?**

### RESPONSE FORMAT

<trig\_source> : **Trigger\_LeVel\_2** <trig\_level>

## G AVAILABILITY

Available only on LC564 and LC584A Series models.

### EXAMPLE (GPIB)

The following instruction sets the trigger level of channel 2 to 1.5 V:

**CMD\$="C2:TRLV2 1.5V": CALL IBWRT(SCOPE%,CMD\$)**

### RELATED COMMANDS

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

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

|                         |                                                                                                                                    |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The TRIG_MODE command specifies the trigger mode.<br>The TRIG_MODE? query returns the current trigger mode.                        |
| <b>COMMAND SYNTAX</b>   | <b>TRig_MoDe</b> <mode><br><mode> : = { <b>AUTO, NORM, SINGLE, STOP</b> }                                                          |
| <b>QUERY SYNTAX</b>     | <b>TRig_MoDe?</b>                                                                                                                  |
| <b>RESPONSE FORMAT</b>  | <b>TRig_MoDe</b> <mode>                                                                                                            |
| <b>EXAMPLE (GPIB)</b>   | The following selects the normal mode:<br><b>CMD\$="TRMD NORM": CALL IBWRT(SCOPE%,CMD\$)</b>                                       |
| <b>RELATED COMMANDS</b> | ARM_ACQUISITION, STOP, TRIG_SELECT, SEQUENCE,<br>TRIG_COUPLING, TRIG_LEVEL, TRIG_LEVEL_2,<br>TRIG_PATTERN, TRIG_SLOPE, TRIG_WINDOW |



## ACQUISITION

## TRIG\_PATTERN, TRPAG 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, and Channel 3 and Channel 4 on four-channel models), 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.

| Notation  |                 |           |                |
|-----------|-----------------|-----------|----------------|
| <b>L</b>  | low             | <b>H</b>  | high           |
| <b>PR</b> | pattern present | <b>AB</b> | pattern absent |
| <b>EN</b> | pattern entered | <b>EX</b> | pattern exited |

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**, **C3G**, **C4G**, **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**, **C3G**, **C4G**, **EX**}

<state> : = {**L**, **H**}

## 9300 &amp; LC Series

## RESPONSE FORMAT

**Trig\_Pattern**

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

**G** AVAILABILITY

Only available on 9350C, 9354C, 9362C, 9370C, 9374C, 9384C and LC Series models.

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

## 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\_LEVEL\_2, TRIG\_MODE, TRIG\_SELECT, TRIG\_SLOPE,  
TRIG\_WINDOW

## 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 may be given in any order and restricted to those variables to be changed.

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

| Trigger Notation |                       |              |                                      |
|------------------|-----------------------|--------------|--------------------------------------|
| <b>DROP</b>      | Dropout               | <b>RUNT</b>  | Runt                                 |
| <b>EDGE</b>      | Edge                  | <b>SLEW</b>  | Slew Rate                            |
| <b>EV</b>        | Event                 | <b>SNG</b>   | Single source                        |
| <b>GLIT</b>      | Glitch                | <b>SQ</b>    | State-Qualified                      |
| <b>HT</b>        | Hold type             | <b>SR</b>    | Source                               |
| <b>HV</b>        | Hold value            | <b>STD</b>   | <sup>§</sup> Standard (Edge Trigger) |
| <b>HV2</b>       | Second hold value     | <b>TEQ</b>   | Edge-Qualified                       |
| <b>IL</b>        | Interval larger       | <b>TEQ1</b>  | Qualified First                      |
| <b>INTV</b>      | Interval              | <b>TI</b>    | Time                                 |
| <b>IS</b>        | Interval smaller      | <b>TL</b>    | Time within                          |
| <b>I2</b>        | Interval-width window | <b>TV**</b>  | TV                                   |
| <b>OFF</b>       | No hold-off on wait   | <b>CHAR</b>  | Characteristics                      |
| <b>PA</b>        | Pattern               | <b>FLD</b>   | Field                                |
| <b>PL</b>        | Pulse larger          | <b>FLDC</b>  | Field count                          |
| <b>PS</b>        | Pulse smaller         | <b>ILACE</b> | Interlace                            |
| <b>P2</b>        | Pulse-width window    | <b>LINE</b>  | Line                                 |
| <b>QL</b>        | Qualifier             | <b>LPIC</b>  | Lines per picture                    |

<sup>§</sup> Note: HT and HV do not apply to the Standard Trigger.

\*\* TV Trigger has its own particular command syntax, using the notation listed below it on this table.

## 9300 &amp; LC Series

**COMMAND SYNTAX**  
For all but TV Trigger

**TRig\_SELECT** <trig\_type>, **SR**, <source>, **QL**, <source>,  
**HT**, <hold\_type>, **HV**, <hold\_value>, **HV2**, <hold value>  
 <trig\_type> := {**DROP**, **EDGE**, **GLIT**, **INTV**, **PA**, **STD**, **SNG**,  
**SQ**, **TEQ**, **TEQ1**, **RUNT**, **SLEW**}  
 <source> := {**C1**, **C2**, **C3G**, **C4G**, **LINE**, **EX**, **EX10G**, **EX5G**, **PA**}  
 <hold\_type> := {**TI**, **TL**, **EV**, **PS**, **PL**, **IS**, **IL**, **P2**, **I2**, **OFF**}  
 <hold\_value> := See *instrument 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>

*Note: HV2 only returned if <hold\_type> is P2 or I2*

**G AVAILABILITY**

<source> : {C3, C4} only available on four-channel instruments.  
 <source> := EXT10 not on LC564 or LC584 Series models.  
 <source> := EXT5 only on LC564 and LC584 Series models.

**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\$)**

## TV COMMAND SYNTAX

**TRig\_SELECT TV, SR, <source>, FLDC, <field\_count>, FLD, <field>, CHAR, <characteristics>, LPIC, <lpic>, ILAC, <ilace>, LINE, <line>**  
 <trig\_type> : = {TV**G**}  
 <source> : = {C**1**, C**2**, C**3G**, C**4G**, NE, EX, EX**10G**, EX**5G**}  
 <field\_count> : = {**1**, **2**, **4**, **8**}  
 <field> : = 1 to field\_count  
 <characteristics> : = {NTSC, PALSEC, CUST**50**, CUST**60**}  
 <lpic> : = 1 to 1500  
 <ilace> : = {**1**, **2**, **4**, **8**}  
 <line> : = 1 to 1500 or 0 for any line

*Note: The FLD value is interpreted with the current FLDC value. The LINE value is interpreted with the current FLD and CHAR values.*

## QUERY SYNTAX

**TRig\_SELECT?**

## RESPONSE FORMAT

**TRig\_SELECT TV, SR, <source>, FLDC, <field\_count>, FLD, <field>, CHAR, <characteristic>, LINE, <line>**

## **G** AVAILABILITY

<source> : = {C3, C4} only on four-channel instruments.  
 <source> : = EXT10 not on LC564 or LC584 Series models.  
 <source> : = EXT5 only on LC564 and LC584 Series models.  
 <trig\_type> : = TV not available on model 9362C.

## EXAMPLE (GPIB)

The following sets up the trigger system to trigger on the 3rd field, line 17, of the 8-field PAL/SECAM TV signal applied to the external input.

**CMD\$="TRSE TV, SR, EX, FLDC, 8, FLD, 3, CHAR, PALSEC, LINE, 17": CALL IBWRT(SCOPE%, CMD\$)**

## RELATED COMMANDS

TRIG\_COUPLING, TRIG\_DELAY, TRIG\_LEVEL, TRIG\_LEVEL\_2, TRIG\_PATTERN, TRIG\_MODE, TRIG\_SLOPE, TRIG\_WINDOW

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

The TRIG\_SLOPE command sets the trigger slope of the specified trigger source. An environment error (*see table on page 72*) will be generated when an incompatible TRSL order is received while the trigger coupling is set to HFDIV (see TRIG\_COUPLING).

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<sup>G</sup>, C4<sup>G</sup>, LINE, EX, EX10<sup>G</sup>, EX5<sup>G</sup>**}

<trig\_slope> := {**NEG, POS, WINDOW<sup>G</sup>**}

**QUERY SYNTAX**

<trig\_source> : **TRig\_SLope?**

**RESPONSE FORMAT**

<trig\_source> : **TRig\_SLope** <trig\_slope>

**G AVAILABILITY**

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

<trig\_source> := EXT10 not on LC564 or LC584 Series models.

<trig\_source> := EXT5 only on LC564 and LC584 Series models.

<trig\_slope> := WINDOW only available on certain models. And not available when the trigger source is set to LINE.

**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\_LEVEL\_2, TRIG\_PATTERN, TRIG\_MODE,  
TRIG\_SELECT, TRIG\_SLOPE, TRIG\_WINDOW

## ACQUISITION

## TRIG\_WINDOW, TRWIG Command/Query

### DESCRIPTION

The TRIG\_WINDOW command sets the window amplitude in volts on the current Edge trigger source. The window is centered around the Edge trigger level.

The TRIG\_WINDOW? query returns the current window amplitude.

### COMMAND SYNTAX

**TRig\_WIndow** <value>

<value> : = 0 to 25 V (maximum range)

*Note: The suffix V is optional.*

### QUERY SYNTAX

**TRig\_WIndow?**

### RESPONSE FORMAT

**TRig\_WIndow** <trig\_level>

## G AVAILABILITY

Not available on 9350C, 9354C, 9370C, 9374C, 9384C or 9362C Series oscilloscopes, nor on color models other than those in the LC564 and LC584 Series.

### EXAMPLE

The following command adjusts the window size to +0.5 V:

**CMD\$="TRWI 0.5V": CALL IBWRT(SCOPE%,CMD\$)**

### RELATED COMMANDS

TRIG\_COUPLING, TRIG\_DELAY, TRIG\_LEVEL,  
TRIG\_LEVEL\_2, TRIG\_PATTERN, TRIG\_MODE,  
TRIG\_SELECT, TRIG\_SLOPE

**MISCELLANEOUS****\*TST?**  
Query**DESCRIPTION**

The \*TST? query performs an internal self-test, the response indicating whether the self-test has detected any errors. The self-test includes testing the hardware of all channels, the timebase and the trigger circuits.

Hardware failures are identified by a unique binary code in the returned <status> number. A "0" response indicates that no failures occurred.

**QUERY SYNTAX****\*TST?****RESPONSE FORMAT****\*TST** <status>

&lt;status&gt; : = 0 self-test successful

**EXAMPLE (GPIB)**

The following causes a self-test to be performed:

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

Response message (if no failure):

**\*TST 0****RELATED COMMANDS****\*CAL**



## STATUS

**URR?**  
Query

### DESCRIPTION

The URR? query reads and clears the contents of the User Request status Register (URR). The URR register specifies which button in the menu field was pressed.

In remote mode, the URR register indicates the last button was pressed, provided it was activated with a KEY command (see *page 104*). In local mode, the URR register indicates whether the CALL HOST button has been pressed. If no menu button has been pressed since the last URR? query, the value 0 is returned.

| User Request Status Register Structure (URR) |                                                                                                          |
|----------------------------------------------|----------------------------------------------------------------------------------------------------------|
| Value                                        | Description                                                                                              |
| 0                                            | No button has been pressed                                                                               |
| 1                                            | The top menu button has been pressed                                                                     |
| 2                                            | The second-from-top menu button has been pressed                                                         |
| 3                                            | The third-from-top menu button has been pressed                                                          |
| 4                                            | The fourth-from-top menu button has been pressed                                                         |
| 5                                            | The fifth-from-top menu button has been pressed                                                          |
| 100                                          | When the "Call Host" command is "On" (the bottom button for the root, or primary, menu has been pressed) |

### QUERY SYNTAX

**URR?**

### RESPONSE FORMAT

**URR** <value>  
<value> : = 0 to 5, 100

### EXAMPLE (GPIB)

The following instruction reads the contents of the URR register:

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

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

Response message:

**URR 0**

### RELATED COMMANDS

CALL\_HOST, KEY, ALL\_STATUS, \*CLS

**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> : = {**TA, TB, TC, TD**}  
<factor> : = 4.0E-3 to 50 (maximum)

**QUERY SYNTAX**

<trace> : **Vert\_MAGnify?**

**RESPONSE FORMAT**

<trace> : **Vert\_MAGnify** <factor>

**EXAMPLE**

The following command 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

| <i>DISPLAY</i>          | <b>VERT_POSITION, VPOS</b><br>Command/Query                                                                                                                                                                                                                                                                                                                |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>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.</p> <p>The VERT_POSITION? query returns the current vertical position of the specified trace.</p>                                                                              |
| <b>COMMAND SYNTAX</b>   | <p>&lt;trace&gt; : <b>Vert_POSITION</b> &lt;display_offset&gt;</p> <p>&lt;trace&gt; : = {<b>TA, TB, TC, TD</b>}</p> <p>&lt;display_offset&gt; : = -5900 to +5900 DIV</p> <p><i>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.</i></p> |
| <b>QUERY SYNTAX</b>     | <trace> : <b>Vert_POSition?</b>                                                                                                                                                                                                                                                                                                                            |
| <b>RESPONSE FORMAT</b>  | <trace> : <b>Vert_POSITION</b> <display_offset>                                                                                                                                                                                                                                                                                                            |
| <b>EXAMPLE</b>          | <p>The following shifts Trace A (TA) upwards by +3 divisions relative to the position at the time of acquisition:</p> <p><b>CMD\$="TA:VPOS 3DIV": CALL IBWRT(SCOPE%,CMD\$)</b></p>                                                                                                                                                                         |
| <b>RELATED COMMANDS</b> | 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 167*) is set if an out-of-range value is entered.

*Note: 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, C3G, C4G**}

<v\_gain> : = See *Operator's Manual, Appendix A, for specifications.*

*Note: The suffix V is optional.*

**QUERY SYNTAX**

<channel> : **Volt\_DIV?**

**RESPONSE FORMAT**

<channel> : **Volt\_DIV** <v\_gain>

**G AVAILABILITY**

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

**EXAMPLE**

The following command sets the vertical sensitivity of channel 1 to 50 mV/div:

**CMD\$="C1:VDIV 50MV": CALL IBWRT(SCOPE%, CMD\$)**

|               |                         |
|---------------|-------------------------|
| <b>STATUS</b> | <b>*WAI<br/>Command</b> |
|---------------|-------------------------|

|                         |                                                                                                                                                                                                                      |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The *WAI (WAI to continue) command, required by the IEEE 488.2 standard, has no effect on the instrument, as the oscilloscope only starts processing a command when the previous command has been entirely executed. |
| <b>COMMAND SYNTAX</b>   | <b>*WAI</b>                                                                                                                                                                                                          |
| <b>RELATED COMMANDS</b> | *OPC                                                                                                                                                                                                                 |

| <b>ACQUISITION</b>      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | <b>WAIT</b><br>Command |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------|
| <b>DESCRIPTION</b>      | The WAIT command prevents the instrument from analyzing new commands until the oscilloscope has completed the current acquisition.                                                                                                                                                                                                                                                                                                                              |                        |
| <b>COMMAND SYNTAX</b>   | <b>WAIT</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                        |
| <b>EXAMPLE (GPIB)</b>   | <pre>send: "TRMD SINGLE" loop {send: "ARM; WAIT; C1: PAVA?MAX"       read response       process response       }</pre> <p>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.</p> <p>"C1: PAVA?MAX" instructs the instrument to evaluate the maximum data value in the Channel 1 waveform.</p> |                        |
| <b>RELATED COMMANDS</b> | *TRG                                                                                                                                                                                                                                                                                                                                                                                                                                                            |                        |

## 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.

The WAVEFORM command stores an external waveform back into the oscilloscope's internal memory. A waveform consists of several distinct entities:

1. the descriptor (DESC)
2. the user text (TEXT)
3. the time (TIME) descriptor
4. the data (DAT1) block, and, optionally
5. a second block of data (DAT2).

*For further information on the structure of the waveform refer to Chapter 4.*

*Note: Only complete waveforms queried with "WAVEFORM? ALL" can be restored into the oscilloscope.*

The WAVEFORM? query instructs the oscilloscope to transmit a waveform to the controller. The entities may 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 command, the last COMM\_ORDER command, and the last COMM\_FORMAT command.*

### COMMAND SYNTAX

<memory> : **WaveForm ALL** <waveform\_data\_block>

<memory> : = {**M1, M2, M3, M4**}

<waveform\_data\_block> : = Arbitrary data block (see Chapter 5).

### QUERY SYNTAX

<trace> : **WaveForm?** <block>

<trace> : = {**TA, TB, TC,TD, M1, M2, M3, M4, C1, C2, C3G,c4G**}

<block> : = {**DESC, TEXT, TIME, DAT1, DAT2, ALL**}

*Note: If no parameter is given ALL will be assumed.*

## 9300 &amp; LC Series

## RESPONSE FORMAT

<trace> : **WaveForm** <block>, <waveform\_data\_block>

*Note: It may be convenient to disable the response header if the waveform is to be restored. Refer to command COMM\_HEADER for further details.*

**G** 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 instrument, 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  |                |           |                  |
|-----------|----------------|-----------|------------------|
| <b>FP</b> | first point    | <b>NP</b> | number of points |
| <b>SN</b> | segment number | <b>SP</b> | 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    |

## 9300 &amp; LC Series

The WAVEFORM\_SETUP? query returns the transfer parameters currently in use.

**COMMAND SYNTAX****WaveForm\_SetUp**

**SP**, <sparsing>, **NP**, <number>, **FP**, <point>, **SN**,  
<segment>

*Note 1: After power-on, all values are set to 0 (i.e. entire waveforms will be transmitted without sparsing).*

*Note 2: Parameters are grouped in pairs. The first of the pair names the variable to be modified, whilst the second gives the new value to be assigned. Pairs may be given in any order and may be 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 command specifies that every 3rd data point (SP=3) starting at address 200 should be transferred:

**CMD\$="WFSU SP, 3, FP, 200": CALL IBWRT(SCOPE%, CMD\$)**

**RELATED COMMANDS**

INSPECT, WAVEFORM, TEMPLATE

## WAVEFORM TRANSFER

## WAVEFORM\_TEXT, WFTX Command/Query

|                       |                                                                                                                                                                                                                                              |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DESCRIPTION           | <p>The WAVEFORM_TEXT command is used to document the conditions under which a waveform has been acquired. The text buffer is limited to 160 characters.</p> <p>The WAVEFORM_TEXT? query returns the text section of the specified trace.</p> |
| COMMAND SYNTAX        | <p>&lt;trace&gt; : <b>WaveForm_Text</b> '&lt;text&gt;'</p> <p>&lt;trace&gt; : = {<b>TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, C3, C4</b>}</p> <p>&lt;text&gt; : = An ASCII message (max. 160 characters long)</p>                              |
| QUERY SYNTAX          | <trace> : <b>WaveForm_Text?</b>                                                                                                                                                                                                              |
| RESPONSE FORMAT       | <trace> : <b>WaveForm_Text</b> "<text>"                                                                                                                                                                                                      |
| <b>G</b> AVAILABILITY | <trace> : = {C3, C4} only on four-channel instruments.                                                                                                                                                                                       |
| EXAMPLE (GPIB)        | <p>The following documents Trace A (TA):</p> <p><b>MSG\$= "'Averaged pressure signal. Experiment carried out Jan.15, 98'"</b></p> <p><b>CMD\$= "TA:WFTX"+ MSG\$: CALL IBWRT(SCOPE%,CMD\$)</b></p>                                            |
| RELATED COMMAND       | INSPECT, WAVEFORM, TEMPLATE                                                                                                                                                                                                                  |

**DISPLAY****XY\_ASSIGN?, XYAS?**

Query

**DESCRIPTION** The XY\_ASSIGN? query returns the traces currently assigned to the XY display. If there is no trace assigned to the X-axis and/or the Y-axis the value UNDEF will be returned instead of the trace name.

**QUERY SYNTAX** XY\_ASsign?

**RESPONSE FORMAT** XY\_ASsign <X\_source>, <Y\_source>

<X\_source> := {UNDEF, TA, TB, TC, TD, C1, C2, C3, C4}

<Y\_source> := {UNDEF, TA, TB, TC, TD, C1, C2, C3, C4}

**G AVAILABILITY**

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

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

**EXAMPLE (GPIB)**

The following query finds the traces assigned to the X-axis and the Y-axis respectively:

**CMD\$\$="XYAS?": CALL IBWRT(SCOPE%, CMD\$)**

Example of response message:

**XYAS C1, C2**

**RELATED COMMANDS** TRACE

| <b>CURSOR</b>           | <b>XY_CURSOR_ORIGIN, XYCO</b><br>Command/Query                                                                                                                                                                                                                                                                                                                                               |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | <p>The XY_CURSOR_ORIGIN command sets the position of the origin for absolute cursor measurements on the XY display.</p> <p>Absolute cursor values may be measured either with respect to the point (0,0) volts (OFF) or with respect to the center of the XY grid (ON).</p> <p>The XY_CURSOR_ORIGIN query returns the current assignment of the origin for absolute cursor measurements.</p> |
| <b>COMMAND SYNTAX</b>   | <p><b>XY_Cursor_Origin</b> &lt;mode&gt;</p> <p>&lt;mode&gt; : = {<b>ON, OFF</b>}</p>                                                                                                                                                                                                                                                                                                         |
| <b>QUERY SYNTAX</b>     | <b>XY_Cursor_Origin?</b>                                                                                                                                                                                                                                                                                                                                                                     |
| <b>RESPONSE FORMAT</b>  | <b>XY_Cursor_Origin</b> <mode>                                                                                                                                                                                                                                                                                                                                                               |
| <b>EXAMPLE (GPIB)</b>   | <p>The following command sets the origin for absolute cursor measurements to the center of the XY grid.</p> <p><b>CMD\$\$="XYCO ON": CALL IBWRT(SCOPE%,CMD\$\$)</b></p>                                                                                                                                                                                                                      |
| <b>RELATED COMMANDS</b> | XY_CURSOR_VALUE                                                                                                                                                                                                                                                                                                                                                                              |

**CURSOR****XY\_CURSOR\_SET, XYCS**  
Command/Query**DESCRIPTION**

The XY\_CURSOR\_SET command allows the user to position any one of the six independent XY voltage cursors at a given screen location. The positions of the cursors can be modified or queried even if the required cursor is not currently displayed or if the XY display mode is OFF.

The XY\_CURSOR\_SET? query indicates the current position of the cursor(s).

The CURSOR\_SET command is used to position the time cursors.

| Notation    |                               |
|-------------|-------------------------------|
| <b>XABS</b> | vertical absolute on X axis   |
| <b>XREF</b> | vertical reference on X axis  |
| <b>XDIF</b> | vertical difference on X axis |
| <b>YABS</b> | vertical absolute on Y axis   |
| <b>YREF</b> | vertical reference on Y axis  |
| <b>YDIF</b> | vertical difference on Y axis |

**COMMAND SYNTAX****XY\_Cursor\_Set**

<cursor>, <position>[, <cursor>, <position>, ...

<cursor>, <position>]

<cursor> : = {XABS, XREF, XDIF, YABS, YREF, YDIF}

<position> : = -4 to 4 DIV

*Note 1: The suffix DIV is optional.*

*Note 2: Parameters are grouped in pairs. The first of the pair names the cursor to be modified, whilst the second indicates its new value. Pairs may be given in any order and may be restricted to those items to be changed.*

**QUERY SYNTAX**

**XY\_Cursor\_Set?** [<cursor>, ...<cursor>]

<cursor> : = {XABS, XREF, XDIF, YABS, YREF, YDIF, ALL}

*Note: If <cursor> is not specified, ALL will be assumed.*

**RESPONSE FORMAT**

**XY\_Cursor\_Set**

<cursor>, <position>[, <cursor>, <position>...,  
<cursor>, <position>]

**EXAMPLE (GPIB)**

The following command positions the XREF and YDIF at +3 DIV and -2 DIV respectively.

**CMDS\$="XYCS XREF, 3DIV, YDIF, -2DIV": CALL  
IBWRT(SCOPE%, CMDS\$)**

**RELATED COMMANDS**

XY\_CURSOR\_VALUE, CURSOR\_MEASURE, CURSOR\_SET

**CURSOR****XY\_CURSOR\_VALUE?, XYCV?**

Query

**DESCRIPTION**

The XY\_CURSOR\_VALUE? query returns the current values of the X versus Y cursors. The X versus Y trace does not need to be displayed to obtain these parameters, but valid sources must be assigned to the X and Y axes.

| Notation                                   |                       |
|--------------------------------------------|-----------------------|
| <cursor type> : = [HABS, HREL, VABS, VREL] |                       |
| <cursor type>_X                            | X                     |
| <cursor type>_Y                            | Y                     |
| <cursor type>_RATIO                        | DY/DX                 |
| <cursor type>_PROD                         | DY* DX                |
| <cursor type>_ANGLE                        | arc tan(DY/DX)        |
| <cursor type>_RADIUS                       | sqrt(DX* DX + DY* DY) |

**QUERY SYNTAX**

**XY\_Cursor\_Value?** [<parameter>,...<parameter>]

<parameter> : = {HABS\_X, HABS\_Y, HABS\_RATIO, HABS\_PROD, HABS\_ANGLE, HABS\_RADIUS, HREL\_X, HREL\_Y, HREL\_RATIO, HREL\_PROD, HREL\_ANGLE, HREL\_RADIUS, VABS\_X, VABS\_Y, VABS\_RATIO, VABS\_PROD, VABS\_ANGLE, VABS\_RADIUS, VREL\_X, VREL\_Y, VREL\_RATIO, VREL\_PROD, VREL\_ANGLE, VREL\_RADIUS, ALL}

*Note: If <parameter> is not specified or equals ALL, all the measured cursor values are returned. If the value of a cursor could not be determined in the current environment, the value UNDEF will be returned. If no trace has been assigned to either the X axis or the Y axis, an environment error will be generated.*

**RESPONSE FORMAT**

**XY\_Cursor\_Value** <parameter>, <value>[, ...<parameter>, <value>]

<value> : = A decimal value or UNDEF



**EXAMPLE (GPIB)**

The following query reads the ratio of the absolute horizontal cursor, the angle of the relative horizontal cursor, and the product of the absolute vertical cursors:

```
CMD$="XYCV? HABS_RATIO,HREL_ANGLE,VABS_PROD:  
CALL IBWRT(SCOPE%,CMD$)
```

**RELATED COMMANDS**

CURSOR\_MEASURE, CURSOR\_VALUE,  
XY\_CURSOR\_ORIGIN

**DISPLAY****XY\_DISPLAY, XYDS**  
Command/Query

|                         |                                                                                                                                                                                           |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>      | The XY_DISPLAY command enables or disables the XY display mode. When off, the scope is in standard display mode.<br><br>The XY_DISPLAY? query returns the current mode of the XY display. |
| <b>COMMAND SYNTAX</b>   | <b>XY_DiSplay</b> <mode><br><br><mode> : = { <b>ON</b> , <b>OFF</b> }                                                                                                                     |
| <b>QUERY SYNTAX</b>     | <b>XY_DiSplay?</b>                                                                                                                                                                        |
| <b>RESPONSE FORMAT</b>  | <b>XY_DiSplay</b> <mode>                                                                                                                                                                  |
| <b>EXAMPLE (GPIB)</b>   | The following turns the XY display ON:<br><br><b>CMD\$\$="XYDS ON": CALL IBWRT(SCOPE%,CMD\$\$)</b>                                                                                        |
| <b>RELATED COMMANDS</b> | GRID                                                                                                                                                                                      |

| <i>DISPLAY</i>         | <b>XY_RENDER, XYRDG</b><br>Command/Query                                                                                                                                  |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DESCRIPTION</b>     | The XY_RENDER command controls the rendering of the XY plot on screen. In Smooth mode, the dots representing XY pairs are connected. In Sharp mode, they are unconnected. |
| <b>COMMAND SYNTAX</b>  | <b>XY_RenDer</b> <state><br><state> : = {SHARP,SMOOTH}                                                                                                                    |
| <b>QUERY SYNTAX</b>    | <b>XY_RENDER?</b>                                                                                                                                                         |
| <b>RESPONSE FORMAT</b> | <b>XY_RENDDER</b> <state>                                                                                                                                                 |
| <b>G AVAILABILITY</b>  | LC scopes only.                                                                                                                                                           |
| <b>EXAMPLE (GPIB)</b>  | The following instruction sets the rendering to SHARP:<br><b>CMD\$="XY_RENDER SHARP": CALL IBWRT(SCOPE%,CMD\$)</b>                                                        |

**DISPLAY****XY\_SATURATION, XYSA**  
Command/Query**DESCRIPTION**

The XY\_SATURATION command sets the level at which the color spectrum of the persistence display is saturated in XY display mode. 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 XY\_SAT? query indicates the saturation level of the persistence data maps.

**COMMAND SYNTAX**

**XY\_Saturation** <trace>, <value> [<trace>, <value>]

<trace> := { C1, C2, C3<sup>G</sup>, C4<sup>G</sup>, TA, TB, TC, TD, ALL }

<value> := 0 to 100 PCT

*Note: The suffix PCT is optional.*

**QUERY SYNTAX**

**XY\_Saturation?**

**RESPONSE FORMAT**

**XY\_Saturation** <trace>, <value>

**AVAILABILITY**

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

**EXAMPLE (GPIB)**

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

CMD\$="XYSA C3,60": CALL IBWRT(SCOPE%,CMD\$)

**RELATED COMMANDS**

PERSIST\_SAT

§ § §

## Example 1

### Use of the Interactive GPIB Program "IBIC"

This example assumes the use of an IBM PC or compatible equipped with a National Instruments GPIB interface card. The GPIB driver is left in default state so that the device name "dev4" corresponds to the GPIB address 4, the oscilloscope address. All text is user-entered:

```

IBIC<cr>
    program announces itself
: ibfind<CR>
    enter board/device name: dev4<CR>
dev4: ibwrt<CR>
    enter string: "tdiv?"<CR>
[0100]          ( cmpl )
count: 5
dev4: ibrd<CR>
    enter byte count: 10<CR>
[0100]          ( cmpl )
count: 10
54 44 49 56 20 35 30 45          T D I V   5 0 E
2D 39                          -   9
dev4: ibwrt<CR>
    enter string: "c1:cpl?"<CR>
[0100]          ( cmpl )
count: 7
dev4: ibrd<CR>
    enter byte count: 20<CR>
[2100]          ( end cmpl )
count: 11
43 31 3A 43 50 4C 20 44          C 1 : C P L D
35 30 0A                        5 0 z
dev4: q<CR>                          to quit the program.
  
```

## Example 2

### GPIB Program for IBM PC (High-Level Function Calls)

The following BASICA program allows full interactive control of the oscilloscope using an IBM PC as GPIB controller. As in Example 1, it is assumed the controller is equipped with a National Instruments GPIB interface card. All commands listed in the *System Commands* section can be used following this example simply by entering the text string of the command. For example, "C1:VDIV 50 MV", without the quotation marks. The program automatically displays the information sent back by the oscilloscope in response to queries.

In addition, a few utilities have been provided for convenience. The commands ST and RC enable waveform data to be stored on or retrieved from disk if correct drive and file names are provided. The command LC returns the oscilloscope to local mode. Responses sent back by the oscilloscope are interpreted as character strings and are thus limited to a maximum of 255 characters.

```

1-99  <DECL.BAS>
100   CLS
110   PRINT "Control of the 9300 via GPIB and IBM PC"
115   PRINT ""
120   PRINT "Options :      EX to exit      LC local mode"
125   PRINT "          ST store data RC recall data"
130   PRINT ""
140   LINE INPUT "GPIB-address of oscilloscope (1...16)? :",ADDR$
145   DEV$ = "DEV" + ADDR$
150   CALL IBFIND(DEV$,SCOPE%)
155   IF SCOPE% < 0 THEN GOTO 830
160   TMO% = 10 'timeout = 300 msec (rather than default 10 sec)
165   CALL IBTMO(SCOPE%,TMO%)
170   '
200   LOOP% = 1
205   WHILE LOOP%
210       LINE INPUT "Enter command (EX --> Exit) : ",CMD$
220       IF CMD$ = "ex" OR CMD$ = "EX" THEN LOOP% = 0 : GOTO 310
230       IF CMD$ = "st" OR CMD$ = "ST" THEN GOSUB 600 : GOTO 300
240       IF CMD$ = "rc" OR CMD$ = "RC" THEN GOSUB 700 : GOTO 300
250       IF CMD$ = "lc" OR CMD$ = "LC" THEN GOSUB 400 : GOTO 300
260       IF CMD$ = "" THEN GOTO 300

```

```
270          CALL IBWRT(SCOPE%,CMD$)
275          IF IBSTA% < 0 THEN GOTO 840
280          GOSUB 500
300      WEND
310      GOSUB 400
320      END
400      '
405      'SUBROUTINE LOCAL_MODE
410      '
420      CALL IBLOC(SCOPE%)
425      PRINT ""
430      RETURN
500      '
505      'SUBROUTINE GET_DATA
510      'If there are no data to read, simply wait until timeout occurs
515      '
520      CALL IBRD(SCOPE%,RD$)
525      I = IBCNT% 'IBCNT% is the number of characters read
530      FOR J = 1 TO I
535          PRINT MID$(RD$,J,1);
540      NEXT J
545      PRINT ""
550      RETURN
600      '
605      'SUBROUTINE STORE_DATA
610      '
615      RD1$=SPACE$(3)
620      LINE INPUT "Specify trace (TA...TD,M1...M4,C1...C4): ",TRACE$
625      LINE INPUT "Enter filename : ",FILE$
630      CMD$="WFSU NP,0,SP,0,FP,0,SN,0; CHDR SHORT"
640      CALL IBWRT(SCOPE%,CMD$)
645      CMD$=TRACE$+"WF?"
650      CALL IBWRT(SCOPE%,CMD$)
660      CALL IBRD(SCOPE%,RD1$)          'Discard first 3 chars of response
665      CALL IBRDF(SCOPE%,FILE$)
670      IF IBSTA% < 0 THEN GOTO 840
675      PRINT ""
680      RETURN
700      '
705      'SUBROUTINE RECALL_DATA
710      '
715      LINE INPUT "Specify target memory (M1...M4):",MEM$
720      LINE INPUT "Enter filename : ",FILE$
730      CMD$=MEM$+" ":
735      CALL IBWRT(SCOPE%,CMD$)
740      CALL IBWRTF(SCOPE%,FILE$)
745      IF IBSTA% < 0 THEN GOTO 840
```

## GPIB Program Examples



```
750 PRINT ""
755 RETURN
800 '
810 'ERROR HANDLER
820 '
830 PRINT "IBFIND ERROR"
835 END
840 PRINT "GPIB ERROR -- IBERR: ";IBERR%;"IBSTA: ";HEX$(IBSTA%)
845 END
```

**Note:**

*It is assumed that the National Instruments GPIB driver GPIB.COM is in its default state. This means that the interface board can be referred to by its symbolic name 'GPIB0' and that devices on the GPIB with addresses 1 to 16 can be called by the symbolic name 'DEV1' to 'DEV16'.*

*Lines 1–99 are a copy of the file DECL.BAS supplied by National Instruments. The first six lines are required for the initialization of the GPIB handler. DECL.BAS requires access to the file BIB.M during the GPIB initialization. BIB.M is one of the files supplied by National Instruments, and must exist in the directory currently in use.*

*The first two lines of DECL.BAS each contain a string "XXXXX" which must be replaced by the number of bytes which determine the maximum workspace for BASICA (computed by subtracting the size of BIB.M from the currently available space in BASICA). For example, if the size of BIB.M is 1200 bytes and when BASICA is loaded it reports "60200 bytes free", "XXXXX" would be replaced by the value 59000 or less.*

*The default timeout of 10 seconds is modified to 300 ms during the execution of this program. However, the default value of the GPIB handler remains unchanged. Whenever a remote command is entered by the user, the program sends it to the instrument with the function call IBWRT. Afterwards, it always executes an IBRD call, regardless of whether or not a response is expected. If a response is received it is immediately displayed. If there is no response, the program waits until time-out and then asks for the next command.*



## Example 3

### GPIB Program for IBM PC (Low-Level Function Calls)

This example has the same function as Example 2, but is written with low-level function calls. The program assumes that the controller (board) and oscilloscope (device) are at addresses 0 and 4, respectively, and the decimal addresses are:

|            | Listener Address   | Talker Address    |
|------------|--------------------|-------------------|
| Controller | 32(ASCII <space>)  | 64 (ASCII @)      |
| Device     | 32+4=36 (ASCII \$) | 64+4=68 (ASCII D) |

```

1-99  <DECL.BAS>
100  CLS
110  PRINT "Control of the 9300 (address 4) via GPIB and IBM PC"
115  PRINT "": PRINT "Options :  EX to exit          LC local mode"
120  PRINT "          ST store data      RC recall data": PRINT""
125  LOOP=1
130  CMD1$ = "?_@$" 'Unlisten, Untalk, Board talker, Device listener
135  CMD2$ = "?_ D" 'Unlisten, Untalk, Board listener, Device talker
140  BDNAME$= "GPIB0": CALL IBFIND(BDNAME$,BRD0%)
145  IF BRD0% < 0 THEN GOTO 420
150  CALL IBSIC(BRD0%): IF IBSTA% < 0 THEN GOTO 425
155  WHILE LOOP
160      LINE INPUT "Enter command (EX --> Exit) : ",CMD$
165      V% = 1: CALL IBSRE(BRD0%,V%)
170      IF CMD$ = "ex" OR CMD$ = "EX" THEN LOOP = FALSE: GOTO 205
175      IF CMD$ = "st" OR CMD$ = "ST" THEN GOSUB 285: GOTO 200
180      IF CMD$ = "rc" OR CMD$ = "RC" THEN GOSUB 365: GOTO 200
185      IF CMD$ = "lc" OR CMD$ = "LC" THEN GOSUB 240: GOTO 200
190      IF CMD$ = "" THEN GOTO 200
195      CALL IBCMD(BRD0%,CMD1$): CALL IBWRT(BRD0%,CMD$): GOSUB 270
200  WEND
205  CALL IBSIC(BRD0%): V%=0: CALL IBSRE(BRD0%,V%)
210  CALL IBSIC(BRD0%)
215  END
220  '
230  'LOCAL MODE
235  '
240  V% = 0: CALL IBSRE(BRD0%,V%): PRINT ""
245  RETURN
250  '

```



## --- GPIB Program Examples

```

260 'SUBROUTINE GET_DATA
265 '
270 CALL IBCMD(BRD0%,CMD2$): CALL IBRD(BRD0%,RD$): I=IBCNT%
275 FOR J=1 TO I: PRINT MID$(RD$,J,1);: NEXT J: PRINT ""
280 RETURN
285 '
290 'SUBROUTINE STORE_DATA
295 '
300 RD1$=SPACE$(3)
305 LINE INPUT "Specify trace (TA...TD,M1...M4,C1...C4): ",TRACES$
310 LINE INPUT "Enter filename : ",FILES$
315 CALL IBCMD(BRD0%,CMD1$)
320 CMD$="WFSU NP,0,SP,0,FP,0,SN,0;CHDR SHORT"
321 CALL IBWRT(BRD0%,CMD$)
325 CMD$=TRACES$+":WF?": CALL IBWRT(BRD0%,CMD$)
330 CALL IBCMD(BRD0%,CMD2$): CALL IBRD(BRD0%,RD1$)
335 CALL IBRDF(BRD0%,FILES$)
340 IF IBSTA% < 0 THEN GOTO 430
345 PRINT ""
350 RETURN
355 '
360 'SUBROUTINE RECALL_DATA
365 '
370 LINE INPUT "Specify target memory (M1...M4): ",MEM$
375 LINE INPUT "Enter filename : ",FILES$
380 CALL IBCMD(BRD0%,CMD1$)
385 CMD$=MEM$+":": CALL IBWRT(BRD0%,CMD$)
390 CALL IBWRTF(BRD0%,FILES$)
395 IF IBSTA% < 0 THEN GOTO 430
400 PRINT ""
405 RETURN
410 '
415 'ERROR HANDLER
420 '
425 PRINT "IBFIND ERROR": STOP
430 PRINT "GPIB ERROR -- IBERR : ";IBERR%;"IBSTA : ";HEX$(IBSTA%)
435 STOP
440 END

```

*Note: The Template also describes an array named DUAL. This is simply a way to allow the INSPECT? command to examine the two data arrays together.*

## Waveform Template

This template is the instrument's response to a command of the form "TMPL?":

```
/00
0000000      LECROY_2_2:  TEMPLATE
              8 66 111
;
; Explanation of the formats of waveforms and their descriptors on the
; LeCroy Digital Oscilloscopes,
;   Software Release 44.1.1.1, 94/04/18.
;
; A descriptor and/or a waveform consists of one or several logical data blocks
; whose formats are explained below.
; Usually, complete waveforms are read: at the minimum they consist of
;   the basic descriptor block WAVEDESC
;   a data array block.
; Some more complex waveforms, e.g. Extrema data or the results of a Fourier
; transform, may contain several data array blocks.
; When there are more blocks, they are in the following sequence:
;   the basic descriptor block WAVEDESC
;   the history text descriptor block USERTEXT (may or may not be present)
;   the time array block (for RIS and sequence acquisitions only)
;   data array block
;   auxiliary or second data array block
;
; In the following explanation, every element of a block is described by a
; single line in the form
;
; <byte position>  <variable name>: <variable type> ; <comment>
;
; where
;
; <byte position> = position in bytes (decimal offset) of the variable,
;                  relative to the beginning of the block.
;
; <variable name> = name of the variable.
;
; <variable type> = string          up to 16-character name
;                  terminated with a null byte
;                  byte            8-bit signed data value
;                  word            16-bit signed data value
;                  long            32-bit signed data value
;                  float           32-bit IEEE floating point value
```

**WAVEDESC: BLOCK**

```

;
; Explanation of the wave descriptor block WAVEDESC;
;
;
< 0>          DESCRIPTOR_NAME: string ; the first 8 chars are always WAVEDESC
;
< 16>          TEMPLATE_NAME: string
;
< 32>          COMM_TYPE: enum          ; chosen by remote command COMM_FORMAT
                _0          byte
                _1          word
                endenum
;
< 34>          COMM_ORDER: enum
                _0          HIFIRST
                _1          LOFIRST
                endenum
;
;
; The following variables of this basic wave descriptor block specify
; the block lengths of all blocks of which the entire waveform (as it is
; currently being read) is composed. If a block length is zero, this
; block is (currently) not present.
;
;
;BLOCKS :
;
< 36>          WAVE_DESCRIPTOR: long      ; length in bytes of block WAVEDESC
< 40>          USER_TEXT: long           ; length in bytes of block USERTEXT
< 44>          RES_DESC1: long           ;
;
;ARRAYS :
;
< 48>          TRIGTIME_ARRAY: long      ; length in bytes of TRIGTIME array
;
< 52>          RIS_TIME_ARRAY: long      ; length in bytes of RIS_TIME array
;
< 56>          RES_ARRAY1: long           ; an expansion entry is reserved
;
< 60>          WAVE_ARRAY_1: long         ; length in bytes of 1st simple
                                           ; data array. In transmitted waveform,
                                           ; represent the number of transmitted
                                           ; bytes in accordance with the NP
                                           ; parameter of the WFSU remote command
                                           ; and the used format (see COMM_TYPE).
;
< 64>          WAVE_ARRAY_2: long         ; length in bytes of 2nd simple
                                           ; data array
;

```



```

< 68>      RES_ARRAY2: long
< 72>      RES_ARRAY3: long      ; 2 expansion entries are reserved
;
; The following variables identify the instrument
;
< 76>      INSTRUMENT_NAME: string
;
< 92>      INSTRUMENT_NUMBER: long
;
< 96>      TRACE_LABEL: string    ; identifies the waveform.
;
<112>      RESERVED1: word
<114>      RESERVED2: word      ; 2 expansion entries
;
; The following variables describe the waveform and the time at
; which the waveform was generated.
;
<116>      WAVE_ARRAY_COUNT: long ; number of data points in the data
;                                ; array. If there are two data
;                                ; arrays (FFT or Extrema), this number
;                                ; applies to each array separately.
;
<120>      PNTS_PER_SCREEN: long  ; nominal number of data points
;                                ; on the screen
;
<124>      FIRST_VALID_PNT: long   ; count of number of points to skip
;                                ; before first good point
;                                ; FIRST_VALID_POINT = 0
;                                ; for normal waveforms.
;
<128>      LAST_VALID_PNT: long    ; index of last good data point
;                                ; in record before padding (blanking)
;                                ; was started.
;                                ; LAST_VALID_POINT = WAVE_ARRAY_COUNT-1
;                                ; except for aborted sequence
;                                ; and rollmode acquisitions
;
<132>      FIRST_POINT: long       ; for input and output, indicates
;                                ; the offset relative to the
;                                ; beginning of the trace buffer.
;                                ; Value is the same as the FP parameter
;                                ; of the WFSU remote command.
;
<136>      SPARSING_FACTOR: long   ; for input and output, indicates
;                                ; the sparsing into the transmitted
;                                ; data block.
;                                ; Value is the same as the SP parameter
;                                ; of the WFSU remote command.

```

## 9300 &amp; LC Series

```

;
<140>      SEGMENT_INDEX: long      ; for input and output, indicates the
;                                     ; index of the transmitted segment.
;                                     ; Value is the same as the SN parameter
;                                     ; of the WFSU remote command.
;
<144>      SUBARRAY_COUNT: long    ; for Sequence, acquired segment count,
;                                     ; between 0 and NOM_SUBARRAY_COUNT
;
<148>      SWEEPS_PER_ACQ: long    ; for Average or Extrema,
;                                     ; number of sweeps accumulated
;                                     ; else 1
;
<152>      POINTS_PER_PAIR: word   ; for Peak Detect waveforms (which always
;                                     ; include data points in DATA_ARRAY_1 and
;                                     ; min/max pairs in DATA_ARRAY_2).
;                                     ; Value is the number of data points for
;                                     ; each min/max pair.
;
<154>      PAIR_OFFSET: word       ; for Peak Detect waveforms only
;                                     ; Value is the number of data points by
;                                     ; which the first min/max pair in
;                                     ; DATA_ARRAY_2 is offset relative to the
;                                     ; first data value in DATA_ARRAY_1.
;
<156>      VERTICAL_GAIN: float
;
<160>      VERTICAL_OFFSET: float  ; to get floating values from raw data :
;                                     ; VERTICAL_GAIN * data - VERTICAL_OFFSET
;
<164>      MAX_VALUE: float        ; maximum allowed value. It corresponds
;                                     ; to the upper edge of the grid.
;
<168>      MIN_VALUE: float        ; minimum allowed value. It corresponds
;                                     ; to the lower edge of the grid.
;
<172>      NOMINAL_BITS: word      ; a measure of the intrinsic precision
;                                     ; of the observation: ADC data is 8 bit
;                                     ; averaged data is 10-12 bit, etc.
;
<174>      NOM_SUBARRAY_COUNT: word ; for Sequence, nominal segment count
;                                     ; else 1
;
<176>      HORIZ_INTERVAL: float   ; sampling interval for time domain
;                                     ; waveforms
;
<180>      HORIZ_OFFSET: double    ; trigger offset for the first sweep of
;                                     ; the trigger, seconds between the
;                                     ; trigger and the first data point

```

```

;
<188>      PIXEL_OFFSET: double      ; needed to know how to display the
;                                         ; waveform
;
<196>      VERTUNIT: unit_definition ; units of the vertical axis
;
<244>      HORUNIT: unit_definition ; units of the horizontal axis
;
<292>      RESERVED3: word
<294>      RESERVED4: word          ; 2 expansion entries
;
<296>      TRIGGER_TIME: time_stamp ; time of the trigger
;
<312>      ACQ_DURATION: float       ; duration of the acquisition (in sec)
;                                         ; in multi-trigger waveforms.
;                                         ; (e.g. sequence, RIS, or averaging)
;
<316>      RECORD_TYPE: enum
;          _0      single_sweep
;          _1      interleaved
;          _2      histogram
;          _3      graph
;          _4      filter_coefficient
;          _5      complex
;          _6      extrema
;          _7      sequence_obsolete
;          _8      centered_RIS
;          _9      peak_detect
;          endenum
;
<318>      PROCESSING_DONE: enum
;          _0      no_processing
;          _1      fir_filter
;          _2      interpolated
;          _3      sparsed
;          _4      autoscaled
;          _5      no_result
;          _6      rolling
;          _7      cumulative
;          endenum
;
<320>      RESERVED5: word          ; expansion entry
;
<322>      RIS_SWEEPS: word         ; for RIS, the number of sweeps
;                                         ; else 1
;
; The following variables describe the basic acquisition
; conditions used when the waveform was acquired

```



```
;
<324>      TIMEBASE: enum
          _0      1_ps/div
          _1      2_ps/div
          _2      5_ps/div
          _3      10_ps/div
          _4      20_ps/div
          _5      50_ps/div
          _6      100_ps/div
          _7      200_ps/div
          _8      500_ps/div
          _9      1_ns/div
          _10     2_ns/div
          _11     5_ns/div
          _12     10_ns/div
          _13     20_ns/div
          _14     50_ns/div
          _15     100_ns/div
          _16     200_ns/div
          _17     500_ns/div
          _18     1_us/div
          _19     2_us/div
          _20     5_us/div
          _21     10_us/div
          _22     20_us/div
          _23     50_us/div
          _24     100_us/div
          _25     200_us/div
          _26     500_us/div
          _27     1_ms/div
          _28     2_ms/div
          _29     5_ms/div
          _30     10_ms/div
          _31     20_ms/div
          _32     50_ms/div
          _33     100_ms/div
          _34     200_ms/div
          _35     500_ms/div
          _36     1_s/div
          _37     2_s/div
          _38     5_s/div
          _39     10_s/div
          _40     20_s/div
          _41     50_s/div
          _42     100_s/div
          _43     200_s/div
          _44     500_s/div
          _45     1_ks/div
          _46     2_ks/div
```



```

    _47  5_ks/div
    _100 EXTERNAL
endenum

;
<326>    VERT_COUPLING: enum
    _0    DC_50_ohms
    _1    ground
    _2    DC_1MOhm
    _3    ground
    _4    AC,_1MOhm
endenum

;
<328>    PROBE_ATT: float
;
<332>    FIXED_VERT_GAIN: enum
    _0    1_uV/div
    _1    2_uV/div
    _2    5_uV/div
    _3    10_uV/div
    _4    20_uV/div
    _5    50_uV/div
    _6    100_uV/div
    _7    200_uV/div
    _8    500_uV/div
    _9    1_mV/div
    _10   2_mV/div
    _11   5_mV/div
    _12   10_mV/div
    _13   20_mV/div
    _14   50_mV/div
    _15   100_mV/div
    _16   200_mV/div
    _17   500_mV/div
    _18   1_V/div
    _19   2_V/div
    _20   5_V/div
    _21   10_V/div
    _22   20_V/div
    _23   50_V/div
    _24   100_V/div
    _25   200_V/div
    _26   500_V/div
    _27   1_kV/div
endenum

;
<334>    BANDWIDTH_LIMIT: enum
    _0    off
    _1    on

```

```

                                endenum
;
<336>      VERTICAL_VERNIER: float
;
<340>      ACQ_VERT_OFFSET: float
;
<344>      WAVE_SOURCE: enum
              _0      CHANNEL_1
              _1      CHANNEL_2
              _2      CHANNEL_3
              _3      CHANNEL_4
              _9      UNKNOWN
                                endenum
;
/00      ENDBLOCK
;
=====
;
USERTEXT: BLOCK
;
; Explanation of the descriptor block USERTEXT at most 160 bytes long.
;
;
< 0>      TEXT: text              ; a list of ASCII characters
;
/00      ENDBLOCK
;
=====
;
DATA_ARRAY_1: ARRAY
;
; Explanation of the data array DATA_ARRAY_1.
; This main data array is always present. It is the only data array for
; most waveforms.
; The data item is repeated for each acquired or computed data point
; of the first data array of any waveform.
;
< 0>      MEASUREMENT: data        ; the actual format of a data is
                                   ; given in the WAVEDESC descriptor
                                   ; by the COMM_TYPE variable.
;
/00      ENDARRAY
;
=====
;
DATA_ARRAY_2: ARRAY
;
; Explanation of the data array DATA_ARRAY_2.
; This is an optional secondary data array for special types of waveforms:

```

```

;      Complex FFT      imaginary part      (real part in DATA_ARRAY_1)
;      Extrema          floor trace         (roof trace in DATA_ARRAY_1)
;      Peak Detect      min/max pairs       (data values in DATA_ARRAY_1)
; In the first 2 cases, there is exactly one data item in DATA_ARRAY_2 for
; each data item in DATA_ARRAY_1.
; In Peak Detect waveforms, there may be fewer data values in DATA_ARRAY_2,
; as described by the variable POINTS_PER_PAIR.
;
< 0>      MEASUREMENT: data      ; the actual format of a data is
;                                ; given in the WAVEDESC descriptor
;                                ; by the COMM_TYPE variable.

;
/00      ENDARRAY
;
;=====
;
TRIGTIME: ARRAY
;
; Explanation of the trigger time array TRIGTIME.
; This optional time array is only present with SEQNCE waveforms.
; The following data block is repeated for each segment which makes up
; the acquired sequence record.
;
< 0>      TRIGGER_TIME: double   ; for sequence acquisitions,
;                                ; time in seconds from first
;                                ; trigger to this one

;
< 8>      TRIGGER_OFFSET: double ; the trigger offset is in seconds
;                                ; from trigger to zeroth data point

;
/00      ENDARRAY
;
;=====
;
RISTIME: ARRAY
;
; Explanation of the random-interleaved-sampling (RIS) time array RISTIME.
; This optional time array is only present with RIS waveforms.
; This data block is repeated for each sweep which makes up the RIS record
;
< 0>      RIS_OFFSET: double     ; seconds from trigger to zeroth
;                                ; point of segment

;
/00      ENDARRAY
;
;=====
;
SIMPLE: ARRAY

```

```
;
; Explanation of the data array SIMPLE.
; This data array is identical to DATA_ARRAY_1. SIMPLE is an accepted
; alias name for DATA_ARRAY_1.
;
< 0>          MEASUREMENT: data          ; the actual format of a data is
                                           ; given in the WAVEDESC descriptor
                                           ; by the COMM_TYPE variable.
;
/00          ENDARRAY
;
;=====
;
DUAL: ARRAY
;
; Explanation of the DUAL array.
; This data array is identical to DATA_ARRAY_1, followed by DATA_ARRAY_2.
; DUAL is an accepted alias name for the combined arrays DATA_ARRAY_1 and
; DATA_ARRAY_2 (e.g. real and imaginary parts of an FFT).
;
< 0>          MEASUREMENT_1: data        ; data in DATA_ARRAY_1.
;
< 0>          MEASUREMENT_2: data        ; data in DATA_ARRAY_2.
;
/00          ENDARRAY
;
;
000000          ENDTEMPLATE
```

**A**

Addresses, 2-2  
 ALL\_STATUS?, ALST?, Query, 13  
 ARM\_ACQUISITION, ARM, Command, 14  
 ATTENUATION, ATTN, Command/Query, 15  
 AUTO\_CALIBRATE, ACAL, Command/Query, 16  
 AUTO\_SCROLL, ASCR, Command/Query, 17  
 AUTO\_SETUP, ASET, Command, 18

**B**

BANDWIDTH\_LIMIT, BWL, Command/Query, 19  
 BASICA, 2-5, 4-5, A-2  
 Binary blocks, 4-8  
 Block Data, 1-8  
 Buffers, 2-3  
 BUZZER, BUZZ, Command, 21

**C**

CAL?, Query, 22  
 CAL\_OUTPUT, COUT, Command/Query, 23  
 CALL\_HOST, CHST, Command/Query, 25  
 Character data, 1-6  
 CLEAR\_MEMORY, CLM, Command, 26  
 CLEAR\_SWEEPS, CLSW, Command, 27  
 CLS, Command, 28  
 CMR (Command Error Status Register), 5-1, 5-3, 5-5, 5-6  
 CMR?, Query, 29  
 COLOR, COLR, Command/Query, 31

COLOR\_SCHEME, CSCH, Command/Query, 33  
 Colors  
   list of colors and their short form names, 31  
 COMBINE\_CHANNELS, COMB, Command/Query, 34  
 COMM\_FORMAT, CFMT, Command/Query, 35  
 COMM\_HEADER, CHDR, Command/Query, 37  
 COMM\_HELP, CHLP, Command/Query, 38  
 COMM\_HELP\_LOG?, CHL?, Query, 39  
 COMM\_NET, CONET, Command/Query, 40  
 COMM\_ORDER, CORD, Command/Query, 41  
 COMM\_RS232, CORS, Command/Query, 42  
 Command Error Status Register.  
   see CMR  
 Commands and Queries, 1-2, 1-3  
   How they are described, 1  
   Notation, 2  
   Overview, 1  
   When they can be used, 1  
 Configuring  
   Printing, 2-19  
 Continuous Polling, 2-13  
 Controller Timeout, 1-3, 2-3, 2-10, 2-14  
 COUPLING, CPL, Command/Query, 45  
 CURSOR\_MEASURE, CRMS, Command/Query, 46  
 CURSOR\_READOUT, ROUT, Command/Query, 49  
 CURSOR\_SET, CRST, Command/Query, 50  
 CURSOR\_VALUE?, CRVA?, Query, 52

**D****Data**

- Arrays, 4-1, 4-2
- ASCII forms, 1-6
- Blocks, 4-1
- Formatting, 4-5, 4-13
- HEX mode, 3-1, 3-5, 4-13
- Horizontal position, 4-10
- Interpretation, 4-6, 4-9
- Sparsing, 4-13
- Values, 4-4, 4-8
- Vertical reading, 4-9
- DATA\_POINTS, DPNT, Command/Query, 53
- DATE, Command/Query, 54
- DDR (Device Dependent Error Status Register), 5-7
- DDR?, Query, 56
- DEFINE, DEF, Command/Query, 56
- DELETE\_FILE, DELF, Command, 62
- Descriptor
  - Block, 4-2
  - Values, 4-4, 4-8
- Device Dependent Error Status Register. see DDR
- DIRECTORY, DIR, Command/Query, 63
- DISPLAY, DISP, Command/Query, 65
- DOT\_JOIN, DTJN, Command/Query, 66
- DUAL\_ZOOM, DZOM, Command/Query, 67

**E**

- Error Messages, 1-2
- ENABLE\_KEY, EKEY, Command/Query, 68
- ESE (Standard Event Status Enable Register), 2-12, 5-1, 5-3
- ESE, Command/Query, 69

- ESR (Standard Event Status Register), 2-12, 5-1, 5-3, 5-5
- ESR?, Query, 70
- Execution Error Status Register. see EXR
- EXR (Execution Error Status Register), 5-1, 5-7
- EXR?, Query, 73

**F**

- FAT\_CURSOR, FATC, Command/Query, 76
- FILENAME, FLNM, Command/Query, 77
- FIND\_CTR\_RANGE, FCR, Command, 78
- FORMAT\_CARD, FCRD, Command/Query, 79
- FORMAT\_FLOPPY, FFLP, Command/Query, 81
- FORMAT\_HDD, FHDD, Command/Query, 83
- FORMAT\_VDISK, FVDISK, Command/Query, 85
- FULL\_SCREEN, FSCR, Command/Query, 87
- FUNCTION\_RESET, FRST, Command, 88

**G**

- GLOBAL\_BWL, GBWL, Command/Query, 89
- GPIO
  - Addresses, 2-2
  - ATN (ATteNtion), 2-3
  - Data lines, 2-2
  - DCL (Device CLear), 2-4
  - EOI (End Or Identify), 1-3, 2-3
  - GET (Group Execute Trigger), 2-4, 2-10
  - GTL (Go To Local), 2-5, 2-9
  - Handshake lines, 2-3
  - Hard copies, 2-19

## Index

Hardcopy, 2-19, 2-20  
Hardware configuration, 2-6  
IEEE 488.1, 2-3  
IEEE 488.2, 2-4  
IFC (InterFace Clear), 2-3, 2-5  
INE (Internal State Change Enable Register), 2-11  
INR (Internal State Change Status Register), 2-11, 2-13  
Listener address, 2-16, 2-21  
LLO (Local LOut), 2-5  
MLA (Listen address), 2-2  
MTA (Talker address), 2-2  
Overview, 1-1  
Polling, 2-13  
PRE (Parallel Poll Enable Register), 2-15  
**Printing**, 2-20  
Program for IBM PC, A-2, A-5  
Programming service requests, 2-10  
Programming transfers, 2-5  
REN (Remote ENable), 2-3, 2-4  
RQS (ReQuest for Service), 2-14  
SDC (Selected Device Clear), 2-4, 2-9  
Signals, 2-2  
Software configuration, 2-6  
SRE (Service Request Enable Register), 2-11  
SRQ (Service Request), 2-10, 2-11, 2-12  
SRQ (Service ReQuest), 2-3  
Standard, 1-2  
Structure, 2-1  
Talker address, 2-16, 2-21  
Transfers, 2-5  
UNL (Universal unlisten), 2-2, 2-17, 2-21  
UNT (Universal untalk), 2-2, 2-17, 2-21

GRID, Command/Query, 90

## H

Hard copies. see GPIB  
HARDCOPY\_SETUP, HCSU, 91  
HARDCOPY\_TRANSMIT, HCTR, Command, 94  
Header, 1-4  
Header Path, 1-5  
Help Messages, 1-2  
HOR\_MAGNIFY, HMAG, Command/Query, 95  
HOR\_POSITION, HPOS, Command/Query, 96

## I

IDN?, Query, 98  
IEEE 488.1, 1-2  
IEEE 488.2, 1-2, 5-1, 5-5  
IEEE Standards. see GPIB  
INE (Internal State Change Enable Register), 2-11, 5-3, 5-6  
INE (Internal State Change Enable Register), 5-1  
INE, Command/Query, 99  
INR (Internal State Change Status Register), 2-13  
INR (Internal State Change Status Register), 2-11, 5-1, 5-6  
INR?, Query, 100  
INSPECT? Queries, 4-4  
INSPECT?, INSP?, Query, 102  
INTENSITY, INTS, Command/Query, 104  
Interface Capabilities, 2-2  
Interface messages, 2-1  
INTERLEAVED, ILVD, Command/Query, 105  
Internal State Change Enable Register. see INE  
Internal State Change Status Register. see INR  
IST Polling, 2-17, 5-3, 5-6  
IST?, Query, 106



**K**

KEY, Command, 107

**L**

Line Splitting, see RS-232-C

Listeners, 2-1

Logical Data Blocks, 4-1

LOGO

Command/Query, 108

**M**

MASK

Command/Query, 109

MATH\_LIMITS, MLIM,

Command/Query, 111

MEASURE\_GATE, MGAT,

Command/Query, 101

MEMORY\_SIZE, MSIZ,

Command/Query, 102

MESSAGE, MSG,

Command/Query, 103

MULTI\_ZOOM, MZOM,

Command/Query, 104

Multipliers, 1-7

**N**

Notation, 2

Numeric Data, 1-6

**O**

OFFSET, OFST,

Command/Query, 116

OFFSET\_CONSTANT, OFCT,

Command/Query, 117

OPC, Command/Query, 118

OPT?, Query, 119

**P**

PANEL\_SETUP, PNSU,

Command/Query, 121

Parallel Poll Enable Register, see  
PRE

Parallel Polling, 2-15

Parameter measurements, 44

PARAMETER\_CLR, PACL,

Command, 122

PARAMETER\_CUSTOM, PACU,

Command/Query, 123

PARAMETER\_DELETE, PADL,

Command, 127

PARAMETER\_STATISTICS?,

PAST?, Query, 128

PARAMETER\_VALUE?, PAVA?,

Query, 129

PASS\_FAIL\_CONDITION,

PFCO, Command/Query, 132

PASS\_FAIL\_COUNTER, PFCT,

Command/Query, 134

PASS\_FAIL\_DO, PFDO,

Command/Query, 135

PASS\_FAIL\_MASK, PFMS,

Command, 137

PASS\_FAIL\_STATUS?, PFST?,

Query, 138

PEAK\_DETECT, PDET,

Command/Query, 139

PER\_CURSOR\_SET, PECS,

Command/Query, 140

PER\_CURSOR\_VALUE?,

PECV?, Query, 142

PERSIST, PERS,

Command/Query, 143

PERSIST\_COLOR, PECL,

Command/Query, 144

PERSIST\_LAST, PELT,

Command/Query, 145

PERSIST\_SAT, PESA,

Command/Query, 146

PERSIST\_SETUP, PESU,

Command/Query, 147

Polling, 2-13

Continuous, 2-13

IST Polling, 2-17

Parallel, 2-15

Serial, 2-14

## Index

PRE (Parallel Poll Enable Register), 2-15, 5-3, 5-6  
 PRE, Command/Query, 148  
 PROBE\_CAL?, PRCA?, Query, 149  
 PROBE\_DEGAUSS?, PRDG?, Query, 150  
 PROBE\_INFOTEXT, PRIT, Query, 151  
 PROBE\_NAME?, PRNA?, Query, 152  
 Program Messages, 1-2, 1-3, 2-1

## Q

Quotation marks  
   their use in command notation, 1-7

## R

RCL, Command, 154  
 REAR\_CALIBRATION, ROUT, Command/Query, 153  
 RECALL, REC, Command, 155  
 RECALL\_PANEL, RCPN, Command, 156  
 RERERENCE\_CLOCK, RCLK, Command/Query, 157  
 Response Messages, 1-8, 1-9  
 RIS Acquisition Times (RISTIME), 4-2  
 RISTIME, 4-2, 4-11  
 RQS (ReQuest for Service), 2-14  
 RS-232-C  
   Configuration, 3-1  
   Echo, 3-2  
   Editing, 3-3  
   Handshake control, 3-2  
   Immediate commands, 3-2  
   Line splitting, 3-4  
   Message terminators, 3-3  
   Overview, 1-1  
   Simulating GPIB Commands, 3-6  
   SRQ (Service ReQuest), 3-4

RST, Command, 158

## S

SAMPLE\_CLOCK, SCLK, Command/Query, 159  
 SAV, Command, 160  
 SCREEN, Command, 161  
 SCREEN\_DUMP, SCDP, Command/Query, 162  
 SCREEN\_SAVE, SCSVG, Command/Query, 163  
 SELECT, SEL, Command/Query, 164  
 SEQUENCE, SEQ, Command/Query, 165  
 Serial Polling, 2-14  
 Service Request Enable Register.  
   see SRE  
 Service Request Reporting, 5-1  
 Service requests, 3-4  
 SIMPLE, 4-2  
 SKEY, Command, 167  
 SLEEP, Command, 168  
 STITLE, Command, 169  
 SRE (Service Request Enable Register), 2-11, 5-1, 5-3, 5-5  
 SRE, Command/Query, 170  
 SRQ (Service Request), 2-10, 2-12, 3-4, 5-3  
 Standard Event Status Register.  
   see ESR  
 Status Byte Register. see STB  
 Status Register Reporting, 5-1  
 STB (Status Byte Register), 2-10, 5-4  
 STB?, Query, 171  
 STOP, Command, 173  
 STORE, STO, Command, 174  
 STORE\_PANEL, STPN, Command, 175  
 STORE\_SETUP, STST, Command/Query, 176  
 STORE\_TEMPLATE, STTM, Command, 177  
 String Data, 1-7

**T**

Talker, 2-1, 2-19  
 Template, 4-1, 4-4, 4-9, 4-10, B-1  
 TIME\_DISPLAY, TDISP,  
     Command/Query, 178  
 TEMPLATE?, TMPL?, Query, 179  
 Terminators, 1-3, 3-3, 4-8  
 TIME\_DIV, TDIV,  
     Command/Query, 180  
 TRACE, TRA,  
     Command/Query, 181  
 TRACE\_LABEL, TRLB,  
     Command/Query, 182  
 TRACE\_OPACITY, TOPA,  
     Command/Query, 183  
 TRACE\_ORDER, TORD,  
     Command/Query, 184  
 TRANSFER\_FILE,  
     Command/Query, 185  
 TRG, Command, 186  
 TRIG\_COUPLING, TRCP,  
     Command/Query, 187  
 TRIG\_DELAY, TRDL,  
     Command/Query, 188  
 TRIG\_LEVEL, TRLV,  
     Command/Query, 189  
 TRIG\_LEVEL\_2, TRLV2,  
     Command/Query, 190  
 TRIG\_MODE, TRMD,  
     Command/Query, 191  
 TRIG\_PATTERN, TRPA,  
     Command/Query, 192  
 TRIG\_SELECT TRSE,  
     Command/Query, 194  
 TRIG\_SLOPE, TRSL,  
     Command/Query, 197  
 TRIG\_WINDOW, TRWI,  
     Command/Query, 198  
 Trigger Times (TRIGTIME), 4-2  
 TRIGTIME, 4-2, 4-11  
 TST?, Query, 199

**U**

URR (User Request Status  
     Register), 5-7  
 URR?, Query, 200  
 User Request Status Register.  
     see URR  
 USERTEXT, 4-2

**V**

VERT\_MAGNIFY, VMAG,  
     Command/Query, 201  
 VERT\_POSITION, VPOS,  
     Command/Query, 202  
 VOLT\_DIV, VDIV,  
     Command/Query, 203

**W**

WAI, Command, 204  
 WAIT, Command, 205  
 Warning Messages, 1-2  
 WAVEDESC. see Descriptor  
 WAVEFORM  
     Command, 4-12  
     Query, 4-6, 4-13  
     Transfer optimization, 4-13  
 Waveform Template, B-1  
 WAVEFORM, WF,  
     Command/Query, 206  
 WAVEFORM\_SETUP, WFSU,  
     Command/Query, 208  
 WAVEFORM\_TEXT, WFTX,  
     Command/Query, 210

**X**

XY\_ASSIGN?, XYAS?,  
     Query, 211  
 XY\_CURSOR\_ORIGIN, XYCO,  
     Command/Query, 212  
 XY\_CURSOR\_SET, XYCS,  
     Command/Query, 213  
 XY\_CURSOR\_VALUE?, XYCV?,  
     Query, 215

## Index

---

XY\_DISPLAY, XYDS,  
Command/Query, 217  
XY\_RENDER, XYRD,  
Command/Query, 218

XY\_SATURATION, XYSA,  
Command/Query, 219