PART TWO

COMMANDS

Part Two lists and describes the commands and queries you will need to remotely operate your Waverunner oscilloscope.

PART TWO: COMMANDS

In this part of the manual, you'll find the commands and queries.

- ***To run Waverunner remotely.***

Use Waverunner Commands and Queries

This part of the manual lists and describes the remote control commands and queries recognized by Waverunner. You can execute all of them in either local or remote state. Where commands or queries for special options are not included, you will find them in those options' dedicated *Operator's Manuals*.

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

Queries perform actions such as obtaining information. They are recognized by **?** following their headers. Many commands can be used as queries with the question mark added.

A brief explanation of the operation performed by the command or query is followed the formal syntax, with the full-name header given in lower-case characters and the short form derived from it in upper-case characters (e.g., DoT_JoiN for DTJN). 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 calls to the related interface subroutines in BASIC. The device name of the oscilloscope is defined as **SCOPE%**.

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

COMMAND NOTATION

The following notation is used in the commands:

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

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

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

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

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

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

Table of Commands and Queries — By Short Form...

PAGE NO.	SHORT FORM	LONG FORM	SUBSYSTEM (CATEGORY)	WHAT THE COMMAND OR QUERY DOES
66	ACAL	AUTO_CALIBRATE	MISCELLANEOUS	Enables or disables automatic calibration.
63	ALST?	ALL_STATUS?	STATUS	Reads and clears the contents of all status registers.
64	ARM	ARM_ACQUISITION	ACQUISITION	Changes acquisition state from "stopped" to "single".
67	ASCR	AUTO_SCROLL	DISPLAY	Controls the Auto Scroll viewing feature.
68	ASET	AUTO_SETUP	ACQUISITION	Adjusts vertical, timebase and trigger parameters.
65	ATTN	ATTENUATION	ACQUISITION	Selects the vertical attenuation factor of the probe.
71	BUZZ	BUZZER	MISCELLANEOUS	Controls the built-in piezo-electric buzzer.
69	BWL	BANDWIDTH_LIMIT	ACQUISITION	Enables/disables bandwidth-limiting low-pass filter.
72	*CAL?	*CAL?	MISCELLANEOUS	Performs complete internal calibration of oscilloscope.
84	CFMT	COMM_FORMAT	COMMUNICATION	Selects the format for sending waveform data.
86	CHDR	COMM_HEADER	COMMUNICATION	Controls formatting of query responses.
87	CHLP	COMM_HELP	COMMUNICATION	Controls operational level of the RC Assistant.
88	CHL	COMM_HELP_LOG	COMMUNICATION	Returns the contents of the RC Assistant log.
74	CHST	CALL_HOST	DISPLAY	Allows manual generation of a service request (SRQ).
75	CLM	CLEAR_MEMORY	FUNCTION	Clears the specified memory.
77	*CLS	*CLS	STATUS	Clears all status data registers.
76	CLSW	CLEAR_SWEEPS	FUNCTION	Restarts the cumulative processing functions.
78	CMR?	CMR?	STATUS	Reads and clears the Command error Register (CMR).
80	COLR	COLOR	DISPLAY	Selects color of individual on-screen objects.
89	CORD	COMM_ORDER	COMMUNICATION	Controls the byte order of waveform data transfers.
91	CORS	COMM_RS232	COMMUNICATION	Sets remote control parameters of the RS-232-C port.
73	COUT	CAL_OUTPUT	MISCELLANEOUS	Sets signal type put out at the CAL connector.
94	CPL	COUPLING	ACQUISITION	Selects the specified input channel's coupling mode.
95	CRMS	CURSOR_MEASURE	CURSOR	Specifies the type of cursor/parameter measurement.
98	CRST?	CURSOR_SET?	CURSOR	Allows positioning of any one of eight cursors.
100	CRVA?	CURSOR_VALUE?	CURSOR	Returns trace values measured by specified cursors.
83	CSCH	COLOR_SCHEME	DISPLAY	Selects the display color scheme.
103	DATE	DATE	MISCELLANEOUS	Changes the date/time of the internal real-time clock.
104	DDR?	DDR?	STATUS	Reads, clears the Device Dependent Register (DDR).
106	DEF	DEFINE	FUNCTION	Specifies math expression for function evaluation.
112	DELF	DELETE_FILE	MASS STORAGE	Deletes files from mass storage.
113	DIR	DIRECTORY	MASS STORAGE	Creates and deletes file directories.
216	DISP	DISPLAY	DISPLAY	Controls the display screen.
102	DPNT	DATA_POINTS	DISPLAY	Controls bold/single pixel display of sample points.
116	DTJN	DOT_JOIN	DISPLAY	Controls the interpolation lines between data points.
117	DZOM	DUAL_ZOOM	DISPLAY	Sets horizontal magnification and positioning.
118	EKEY	ENABLE_KEY	DISPLAY	Allows use of the KEY command in local mode.

PAGE NO.	SHORT FORM	LONG FORM	SUBSYSTEM (CATEGORY)	WHAT THE COMMAND OR QUERY DOES
119	*ESE	*ESE	STATUS	Sets the Standard Event Status Enable register(ESE).
120	*ESR?	*ESR?	STATUS	Reads, clears the Event Status Register (ESR).
123	EXR?	EXR?	STATUS	Reads, clears the EXecution error Register (EXR).
125	FATC	FAT_CURSOR	DISPLAY	Controls width of cursors.
127	FCR	FIND_CTR_RANGE	FUNCTION	Automatically sets the center and width of a histogram.
128	FCRD	FORMAT_CARD	MISCELLANEOUS	Formats the memory card.
130	FFLP	FORMAT_FLOPPY	MISCELLANEOUS	Formats a floppy disk.
132	FHDD	FORMAT_HDD	MASS STORAGE	Formats the removable hard disk.
126	FLNM	FILENAME	MASS STORAGE	Changes default filenames.
135	FRST	FUNCTION_RESET	FUNCTION	Resets a waveform-processing function.
134	FSCR	FULL_SCREEN	DISPLAY	Selects magnified view format for the grid.
136	GBWL	GLOBAL_BWL	ACQUISITION	Enables/disables the Global Bandwidth Limit.
137	GRID	GRID	DISPLAY	Specifies single-, dual- or quad-mode grid display.
138	HCSU	HARDCOPY_SETUP	HARD COPY	Configures the hardcopy driver.
141	HCTR	HARDCOPY_TRANSMIT	HARD COPY	Sends string of ASCII characters to hardcopy unit.
142	HMAG	HOR_MAGNIFY	DISPLAY	Horizontally expands the selected expansion trace.
143	HPOS	HOR_POSITION	DISPLAY	Horizontally positions intensified zone's center.
145	*IDN?	*IDN?	MISCELLANEOUS	For identification purposes.
152	ILVD	INTERLEAVED	ACQUISITION	Enables/disables random interleaved sampling (RIS).
146	INE	INE	STATUS	Sets the Internal state change Enable register (INE).
147	INR?	INR?	STATUS	Reads, clears INternal state change Register (INR).
149	INSP?	INSPECT?	WAVEFORM TRANSFER	Allows acquired waveform parts to be read.
151	INTS	INTENSITY	DISPLAY	Sets the grid or trace/text intensity level.
153	IST?	IST?	STATUS	Reads the current state of the IEEE 488.
154	KEY	KEY	DISPLAY	Displays a string in the menu field.
155	LOGO	LOGO	DISPLAY	Displays LeCroy logo at top of grid.
156	MASK	MASK	CURSOR	Invokes Polymask draw and fill tools.
158	MGAT	MEASURE_GATE	DISPLAY	Controls highlighting of the measurement gate region.
160	MSG	MESSAGE	DISPLAY	Displays a string of characters in the message field.
159	MSIZ	MEMORY_SIZE	ACQUISITION	Selects max. memory length.
161	MZOM	MULTI_ZOOM	DISPLAY	Sets horizontal magnification and positioning.
162	OFST	OFFSET	ACQUISITION	Allows output channel vertical offset adjustment.
163	*OPC	*OPC	STATUS	Sets the OPC bit in the Event Status Register (ESR).
164	*OPT?	*OPT?	MISCELLANEOUS	Identifies oscilloscope options.
167	PACL	PARAMETER_CLR	CURSOR	Clears all current parameters in Custom, Pass/Fail.
168	PACU	PARAMETER_CUSTOM	CURSOR	Controls parameters with customizable qualifiers.
171	PADL	PARAMETER_DELETE	CURSOR	Deletes a specified parameter in Custom, Pass/Fail.
172	PAST?	PARAMETER_STATISTICS?	CURSOR	Returns current statistics parameter values.
173	PAVA?	PARAMETER_VALUE?	CURSOR	Returns current parameter, mask test values.

PAGE NO.	SHORT FORM	LONG FORM	SUBSYSTEM (CATEGORY)	WHAT THE COMMAND OR QUERY DOES
183	**PECS**	**PER_CURSOR_SET**	CURSOR	Positions independent cursors.
185	**PECV?**	**PER_CURSOR_VALUE?**	CURSOR	Returns values measured by cursors.
188	**PELT**	**PERSIST_LAST**	DISPLAY	Shows the last trace drawn in a persistence data map.
186	**PERS**	**PERSIST**	DISPLAY	Enables or disables the persistence display mode.
187	**PECL**	**PERSIST_COLOR**	DISPLAY	Controls color rendering method of persistence traces.
189	**PESA**	**PERSIST_SAT**	DISPLAY	Sets the color saturation level in persistence.
190	**PESU**	**PERSIST_SETUP**	DISPLAY	Selects display persistence duration.
176	**PFCO**	**PASS_FAIL_CONDITION**	CURSOR	Adds a Pass/Fail test condition or custom parameter.
178	**PFCT**	**PASS_FAIL_COUNTER**	CURSOR	Resets the Pass/Fail acquisition counters.
179	**PFDO**	**PASS_FAIL_DO**	CURSOR	Defines desired outcome, actions after Pass/Fail test.
181	**PFMS**	**PASS_FAIL_MASK**	CURSOR	Generates tolerance mask on a trace and stores it.
182	**PFST?**	**PASS_FAIL_STATUS?**	CURSOR	Returns the Pass/Fail test for a given line number.
166	**PNSU**	**PANEL_SETUP**	SAVE/RECALL	Complements the *SAV/*RST commands.
191	***PRE**	***PRE**	STATUS	Sets the PaRallel poll Enable register (PRE).
192	**PRCA?**	**PROBE_CAL?**	PROBES	Performs auto-calibration of connected current probe.
193	**PRDG?**	**PROBE_DEGAUSS?**	PROBES	Degausses, calibrates connected current probe.
194	**PRNA**	**PROBE_NAME?**	PROBES	Names the probe connected to the oscilloscope.
195	***RCL**	***RCL**	SAVE/RECALL	Recalls one of five non-volatile panel setups.
198	**RCPN**	**RECALL_PANEL**	SAVE/RECALL	Recalls a front panel setup from mass storage.
197	**REC**	**RECALL**	WAVEFORM TRANSFER	Recalls a file from mass storage to internal memory.
196	**ROUT**	**REAR_OUTPUT**	MICELLANEOUS	Sets the type of signal put out at rear BNC connector.
199	***RST**	***RST**	SAVE/RECALL	Initiates a device reset.
201	***SAV**	***SAV**	SAVE/RECALL	Stores current state in non-volatile internal memory.
202	**SCDP**	**SCREEN_DUMP**	HARD COPY	Causes a screen dump to the hardcopy device.
200	**SCLK**	**SAMPLE_CLOCK**	ACQUISITION	Allows control of an external timebase.
203	**SCSV**	**SCREEN_SAVE**	DISPLAY	Controls the automatic screen saver.
204	**SEL**	**SELECT**	DISPLAY	Selects the specified trace for manual display control.
205	**SEQ**	**SEQUENCE**	ACQUISITION	Sets the conditions for the sequence mode acquisition.
207	**SLEEP**	**SLEEP**	MICELLANEOUS	Makes the scope wait before it interprets new commands
208	***SRE**	***SRE**	STATUS	Sets the Service Request Enable register (SRE).
209	***STB?**	***STB?**	STATUS	Reads the contents of the IEEE 488.
212	**STO**	**STORE**	WAVEFORM TRANSFER	Stores a trace in internal memory or mass storage.
211	**STOP**	**STOP**	ACQUISITION	Immediately stops signal acquisition.
213	**STPN**	**STORE_PANEL**	SAVE/RECALL	Stores front panel setup to mass storage.
214	**STST**	**STORE_SETUP**	WAVEFORM TRANSFER	Controls the way in which traces are stored.
215	**STTM**	**STORE_TEMPLATE**	WAVEFORM TRANSFER	Stores the waveform template to mass storage.

PAGE NO.	SHORT FORM	LONG FORM	SUBSYSTEM (CATEGORY)	WHAT THE COMMAND OR QUERY DOES
218	TDIV	TIME_DIV	ACQUISITION	Modifies the timebase setting.
216	TDISP	TDISP	MISCELLANEOUS	Changes time display from current to trigger or none.
217	TMPL?	TEMPLATE?	WAVEFORM TRANSFER	Produces a complete waveform template copy.
219	TRA	TRACE	DISPLAY	Enables or disables the display of a trace.
220	TOPA	TRACE_OPACITY	DISPLAY	Controls the opacity of the trace color.
222	TRCP	TRIG_COUPLING	ACQUISITION	Sets the coupling mode of the specified trigger source.
223	TRDL	TRIG_DELAY	ACQUISITION	Sets the time at which the trigger is to occur.
221	*TRG	*TRG	ACQUISITION	Executes an ARM command.
225	TRLV	TRIG_LEVEL	ACQUISITION	Adjusts the trigger level of the specified trigger source.
226	TRMD	TRIG_MODE	ACQUISITION	Specifies the trigger mode.
227	TRSE	TRIG_SELECT	ACQUISITION	Selects the condition that will trigger acquisition.
230	TRSL	TRIG_SLOPE	ACQUISITION	Sets the trigger slope of the specified trigger source.
231	TRWI	TRIG_WINDOW	ACQUISITION	Sets window amplitude on current Edge trigger source.
232	*TST?	*TST?	MISCELLANEOUS	Performs an internal self-test.
233	URR?	URR?	STATUS	Reads, clears User Request status Register (URR).
237	VDIV	VOLT_DIV	ACQUISITION	Sets the vertical sensitivity.
235	VMAG	VERT_MAGNIFY	DISPLAY	Vertically expands the specified trace.
236	VPOS	VERT_POSITION	DISPLAY	Adjusts the vertical position of the specified trace.
238	*WAI	*WAI	STATUS	Required by the IEEE 488.
239	WAIT	WAIT	ACQUISITION	Prevents new analysis until current is completed.
240	WF	WAVEFORM	WAVEFORM TRANSFER	Transfers a waveform from controller to scope.
243	WFSU	WAVEFORM_SETUP	WAVEFORM TRANSFER	Specifies amount of waveform data to go to controller.
245	WFTX	WAVEFORM_TEXT	WAVEFORM TRANSFER	Documents acquisition conditions.
246	XYAS?	XY_ASSIGN?	DISPLAY	Returns traces currently assigned to the XY display.
247	XYCO	XY_CURSOR_ORIGIN	CURSOR	Sets origin position of absolute cursor measurements.
248	XYCS	XY_CURSOR_SET	CURSOR	Allows positioning of XY voltage cursors.
250	XYCV?	XY_CURSOR_VALUE?	CURSOR	Returns the current values of the X vs Y cursors.
252	XYDS	XY_DISPLAY	DISPLAY	Enables or disables the XY display mode.
253	XYRD	XY_RENDER	DISPLAY	Controls XY plot: smooth or disconnected points.
254	XYSA	XY_SATURATION	DISPLAY	Sets persistence color saturation level in XY display.

Table of Commands and Queries — By Subsystem...

PAGE NO.	SHORT FORM	LONG FORM	WHAT THE COMMAND OR QUERY DOES
ACQUISITION — TO CONTROL WAVEFORM CAPTURE			
64	ARM	ARM_ACQUISITION	Changes acquisition state from "stopped" to "single".
68	ASET	AUTO_SETUP	Adjusts vertical, timebase and trigger parameters for signal display.
65	ATTN	ATTENUATION	Selects the vertical attenuation factor of the probe.
69	BWL	BANDWIDTH_LIMIT	Enables or disables the bandwidth-limiting low-pass filter.
136	GBWL	GLOBAL_BWL	Enables/disables the Global Bandwidth Limit
152	ILVD	INTERLEAVED	Enables or disables random interleaved sampling (RIS).
159	MSIZ	MEMORY_SIZE	Allows selection of maximum memory length (M- and L-models only).
162	OFST	OFFSET	Allows vertical offset adjustment of the specified input channel.
200	SCLK	SAMPLE_CLOCK	Allows control of an external timebase.
205	SEQ	SEQUENCE	Sets the conditions for Sequence-mode acquisition.
211	STOP	STOP	Immediately stops signal acquisition.
218	TDIV	TIME_DIV	Modifies the timebase setting.
222	TRCP	TRIG_COUPLING	Sets the coupling mode of the specified trigger source.
223	TRDL	TRIG_DELAY	Sets the time at which the trigger is to occur.
221	*TRG	*TRG	Executes an ARM command.
225	TRLV	TRIG_LEVEL	Adjusts the level of the specified trigger source.
226	TRMD	TRIG_MODE	Specifies Trigger mode.
227	TRSE	TRIG_SELECT	Selects the condition that will trigger acquisition.
230	TRSL	TRIG_SLOPE	Sets the slope of the specified trigger source.
231	TRWI	TRIG_WINDOW	Sets the window amplitude in volts on the current Edge trigger source.
237	VDIV	VOLT_DIV	Sets the vertical sensitivity in volts/div.
239	WAIT	WAIT	Prevents new command analysis until current acquisition completion.
COMMUNICATION — TO SET COMMUNICATION CHARACTERISTICS			
84	CFMT	COMM_FORMAT	Selects the format to be used for sending waveform data.
86	CHDR	COMM_HEADER	Controls formatting of query responses.
87	CHLP	COMM_HELP	Controls operational level of the RC Assistant.
88	CHL	COMM_HELP_LOG	Returns the contents of the RC Assistant log.
89	CORD	COMM_ORDER	Controls the byte order of waveform data transfers.
91	CORS	COMM_RS232	Sets remote control parameters of the RS-232-C port.
CURSOR — TO PERFORM MEASUREMENTS			
95	CRMS	CURSOR_MEASURE	Specifies the type of cursor or parameter measurement for display.
98	CRST?	CURSOR_SET?	Allows positioning of any one of eight independent cursors.
100	CRVA?	CURSOR_VALUE?	Returns the values measured by the specified cursors for a given trace.
156	MASK	MASK	Invokes Polymask draw and fill tools.
167	PACL	PARAMETER_CLR	Clears all current parameters in Custom and Pass/Fail modes.
171	PADL	PARAMETER_DELETE	Deletes a specified parameter in Custom and Pass/Fail modes.

PAGE NO.	SHORT FORM	LONG FORM	WHAT THE COMMAND OR QUERY DOES
172	PAST?	PARAMETER_STATISTICS?	Returns current statistics values for the specified pulse parameter.
173	PAVA?	PARAMETER_VALUE?	Returns current value(s) of parameter(s) and mask tests.
183	PECS	PER_CURSOR_SET	Allows positioning of any one of six independent cursors.
185	PECV?	PER_CURSOR_VALUE?	Returns the values measured by specified cursors for a given trace.
176	PFCO	PASS_FAIL_CONDITION	Adds a Pass/Fail test condition or custom parameter to display.
178	PFCT	PASS_FAIL_COUNTER	Resets the Pass/Fail acquisition counters.
179	PFDO	PASS_FAIL_DO	Defines the desired outcome and actions following a Pass/Fail test.
181	PFMS	PASS_FAIL_MASK	Generates a tolerance mask around a chosen trace and stores it.
182	PFST?	PASS_FAIL_STATUS?	Returns the Pass/Fail test for a given line number.
247	XYCO	XY_CURSOR_ORIGIN	Sets position of origin for absolute cursor measurements on XY display.
248	XYCS	XY_CURSOR_SET	Allows positioning of any one of six independent XY voltage cursors.
250	XYCV?	XY_CURSOR_VALUE?	Returns current values of X vs Y cursors.
		DISPLAY — TO DISPLAY WAVEFORMS	
67	ASCR	AUTO_SCROLL	Controls the Auto Scroll viewing feature.
74	CHST	CALL_HOST	Allows manual generation of a service request (SRQ).
80	COLR	COLOR	Selects color of individual objects such as traces, grids or cursors.
83	CSCH	COLOR_SCHEME	Selects the display color scheme.
102	DPNT	DATA_POINTS	Controls display of sample points in single display pixels or bold.
216	DISP	DISPLAY	Controls the oscilloscope display screen.
116	DTJN	DOT_JOIN	Controls the interpolation lines between data points.
117	DZOM	DUAL_ZOOM	Sets horizontal magnification and positioning for all expanded traces.
118	EKEY	ENABLE_KEY	Allows use of the KEY command in local mode.
125	FATC	FAT_CURSOR	Controls width of cursors.
134	FSCR	FULL_SCREEN	Selects magnified view format for the grid.
137	GRID	GRID	Specifies grid display in single, dual or quad mode.
142	HMAG	HOR_MAGNIFY	Horizontally expands the selected expansion trace.
143	HPOS	HOR_POSITION	Horizontally positions the intensified zone's center on the source trace.
151	INTS	INTENSITY	Sets grid or trace/text intensity level.
154	KEY	KEY	Displays a string in the menu field.
155	LOGO	LOGO	Displays LeCroy logo at top of grid.
158	MGAT	MEASURE_GATE	Controls highlighting of the region between the parameter cursors.
160	MSG	MESSAGE	Displays a string of characters in the message field.
161	MZOM	MULTI_ZOOM	Sets horizontal magnification and positioning for all expanded traces.
186	PERS	PERSIST	Enables or disables the Persistence Display mode.
187	PECL	PERSIST_COLOR	Controls color rendering method of persistence traces.
188	PELT	PERSIST_LAST	Shows the last trace drawn in a persistence data map.
189	PESA	PERSIST_SAT	Sets the color saturation level in persistence.
190	PESU	PERSIST_SETUP	Selects display persistence duration in Persistence mode.
203	SCSV	SCREEN_SAVE	Controls the automatic screen saver.

PAGE NO.	SHORT FORM	LONG FORM	WHAT THE COMMAND OR QUERY DOES
204	SEL	SELECT	Selects the specified trace for manual display control.
219	TRA	TRACE	Enables or disables the display of a trace.
220	TOPA	TRACE_OPACITY	Controls the opacity of the trace color.
235	VMAG	VERT_MAGNIFY	Vertically expands the specified trace.
236	VPOS	VERT_POSITION	Adjusts the vertical position of the specified trace.
246	XYAS?	XY_ASSIGN?	Returns the traces currently assigned to the XY display.
252	XYDS	XY_DISPLAY	Enables or disables the XY display mode.
253	XYRD	XY_RENDER	Controls XY plot: smooth or disconnected points.
254	XYSA	XY_SATURATION	Sets persistence color saturation level in XY display.
FUNCTION — TO PERFORM WAVEFORM MATHEMATICAL OPERATIONS			
75	CLM	CLEAR_MEMORY	Clears the specified memory.
76	CLSW	CLEAR_SWEEPS	Restarts the cumulative processing functions.
106	DEF	DEFINE	Specifies the mathematical expression to be evaluated by a function.
127	FCR	FIND_CTR_RANGE	Automatically sets the center and width of a histogram.
135	FRST	FUNCTION_RESET	Resets a waveform processing function.
HARD COPY — TO PRINT THE CONTENTS OF THE DISPLAY			
138	HCSU	HARDCOPY_SETUP	Configures the hardcopy driver.
141	HCTR	HARDCOPY_TRANSMIT	Sends a string of unmodified ASCII characters to the hardcopy unit.
202	SCDP	SCREEN_DUMP	Causes a screen dump to the hardcopy device.
MASS STORAGE — TO CREATE AND DELETE FILE DIRECTORIES			
112	DELF	DELETE_FILE	Deletes files from the currently selected directory on mass storage.
113	DIR	DIRECTORY	Creates and deletes file directories on mass-storage devices.
128	FCRD	FORMAT_CARD	Formats the memory card.
130	FFLP	FORMAT_FLOPPY	Formats a floppy disk in the Double- or High-Density format.
132	FHDD	FORMAT_HDD	Formats the removable hard disk.
126	FLNM	FILENAME	Changes the default filename of any stored trace, setup or hard copy.
MISCELLANEOUS — TO CALIBRATE AND TEST			
66	ACAL	AUTO_CALIBRATE	Enables or disables automatic calibration.
71	BUZZ	BUZZER	Controls the built-in piezoelectric buzzer.
72	*CAL?	*CAL?	Performs a complete internal calibration of the oscilloscope.
73	COUT	CAL_OUTPUT	Sets the type of signal put out at the CAL connector.
103	DATE	DATE	Changes the date/time of the oscilloscope's internal real-time clock.
145	*IDN?	*IDN?	Used for identification purposes.
164	*OPT?	*OPT?	Identifies oscilloscope options.
196	ROUT	REAR_OUTPUT	Sets the type of signal put out at rear BNC connector.
207	SLEEP	SLEEP	Makes the scope wait before it interprets new commands
216	TDISP	TDISP	Changes time display from current to trigger or none.
232	*TST?	*TST?	Performs an internal self-test.

PAGE NO.	SHORT FORM	LONG FORM	WHAT THE COMMAND OR QUERY DOES
PROBES — TO USE PROBES			
192	PRCA?	PROBE_CAL?	Performs a complete calibration of the connected current probe.
193	PRDG	PROBE_DEGAUSS?	Degausses and calibrates the connected current probe.
194	PRNA	PROBE_NAME?	Gives an identification of the probe connected to the oscilloscope.
SAVE/RECALL SETUP — TO PRESERVE AND RESTORE FRONT PANEL SETTINGS			
166	PNSU	PANEL_SETUP	Complements the *SAV/*RST commands.
195	*RCL	*RCL	Recalls one of five non-volatile panel setups.
198	RCPN	RECALL_PANEL	Recalls a front panel setup from mass storage.
199	*RST	*RST	Initiates a device reset.
201	*SAV	*SAV	Stores the current state in non-volatile internal memory.
213	STPN	STORE_PANEL	Stores the complete front panel setup on a mass-storage file.
STATUS — TO OBTAIN STATUS INFORMATION AND SET UP SERVICE REQUESTS			
63	ALST?	ALL_STATUS?	Reads and clears the contents of all (but one) of the status registers.
77	*CLS	*CLS	Clears all the status data registers.
78	CMR?	CMR?	Reads and clears the contents of the CoMmand error Register (CMR).
104	DDR?	DDR?	Reads and clears the Device-Dependent error Register (DDR).
119	*ESE	*ESE	Sets the standard Event Status Enable (ESE) register.
120	*ESR?	*ESR?	Reads and clears the Event Status Register (ESR).
123	EXR?	EXR?	Reads and clears the EXecution error Register (EXR).
146	INE	INE	Sets the INternal state change Enable register (INE).
147	INR?	INR?	Reads and clears the INternal state change Register (INR).
153	IST?	IST?	Individual STatus reads the current state of IEEE 488.
163	*OPC	*OPC	Sets to true the OPC bit (0) in the Event Status Register (ESR).
191	*PRE	*PRE	Sets the PaRallel poll Enable register (PRE).
208	*SRE	*SRE	Sets the Service Request Enable register (SRE).
209	*STB?	*STB?	Reads the contents of IEEE 488.
233	URR?	URR?	Reads and clears the User Request status Register (URR).
238	*WAI	*WAI	WAIt to continue — required by IEEE 488.
WAVEFORM TRANSFER — TO PRESERVE AND RESTORE WAVEFORMS			
149	INSP?	INSPECT?	Allows acquired waveform parts to be read.
197	REC	RECALL	Recalls a waveform file from mass storage to internal memories M1–4.
212	STO	STORE	Stores a trace in one of the internal memories M1–4 or mass storage.
214	STST	STORE_SETUP	Controls the way in which traces are stored.
215	STTM	STORE_TEMPLATE	Stores the waveform template in a mass-storage device.
217	TMPL?	TEMPLATE?	Produces a copy of the template describing a complete waveform.
240	WF	WAVEFORM	Transfers a waveform from the controller to the oscilloscope.
243	WFSU	WAVEFORM_SETUP	Specifies amount of waveform data for transmission to controller.
245	WFTX	WAVEFORM_TEXT	Documents the conditions under which a waveform has been acquired

STATUS

ALL_STATUS?, ALST?

Query

DESCRIPTION

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

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

QUERY SYNTAX

```
ALl_STatus?
```

RESPONSE FORMAT

ALl_STatus STB,<value>,ESR,<value>,INR,<value>,
DDR,<value>,CMR,<value>,EXR,<value>,URR,<value>

<value> : = 0 to 65535

EXAMPLE (GPIB)

The following reads the contents of all the status registers:

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

Response message:

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

RELATED COMMANDS

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

ACQUISITION

ARM_ACQUISITION, ARM

Command

DESCRIPTION

The ARM_ACQUISITION command enables the signal acquisition process by changing the acquisition state (trigger mode) from "stopped" to "single".

COMMAND SYNTAX

```
ARM_acquisition
```

EXAMPLE

The following enables signal acquisition:

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

RELATED COMMANDS

STOP, *TRG, TRIG_MODE, WAIT

ACQUISITION

ATTENUATION, ATTN

Command/Query

DESCRIPTION

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

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

COMMAND SYNTAX

<channel> : ATTeNuation <attenuation>

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

QUERY SYNTAX

<channel> : ATTeNuation?

RESPONSE FORMAT

<channel> : ATTeNuation <attenuation>

AVAILABILITY

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

EXAMPLE (GPIB)

The following 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 your Waverunner oscilloscope. At power-up, auto-calibration is turned ON, i.e. all input channels are periodically calibrated for the current input amplifier and timebase settings.

The automatic calibration can be disabled by issuing the command ACAL OFF. Whenever convenient, a *CAL? query may be issued to fully calibrate the oscilloscope. When the oscilloscope is returned to local control, the periodic calibrations are resumed.

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 disables auto-calibration:

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

RELATED COMMANDS

*CAL?

DISPLAY

AUTO_SCROLL, ASCR

Command/Query

DESCRIPTION

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.

COMMAND SYNTAX

Auto_SCRoll <action>,<mode>,<speed>

<action> : = {PLAY, REVERSE, STOP}

<mode> : = {DIV_S, DIV_U}

<speed> : = 0.01 to 10.00 for DIV_U mode;
0.10 to 10.00 for DIV_S mode.

QUERY SYNTAX

Auto_SCRoll?

RESPONSE FORMAT

Auto_SCRoll <action>,<mode>,<speed>

EXAMPLE (GPIB)

The following activates Auto Scroll and start scrolling the data to the right at a rate of 2 s/div.:

```
CMD$="ASCR PLAY,DIV_S,2": CALL IBWRT(SCOPE%,CMD$)
```

RELATED COMMANDS

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, C3, C4}

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

AVAILABILITY

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

EXAMPLE

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

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

ACQUISITION

BANDWIDTH_LIMIT, BWL

Command/Query

DESCRIPTION

BANDWIDTH_LIMIT enables or disables the bandwidth-limiting low-pass filter. When Global_BWL (see page 136) 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, 200MHZ}

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

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>]]]

AVAILABILITY

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

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 turns off Global_BWL):

```
CMD$="GBWL OFF": CALL IBWRT(SCOPE%,CMD$)
CMD$="BWL C1,ON": CALL IBWRT(SCOPE%,CMD$)
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 you select by turning the buzzer ON or OFF.

NOTE: This command is only able to be used in oscilloscopes fitted with the CLBZ hard option.

COMMAND SYNTAX

BUZZer <state>

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

EXAMPLE (GPIB)

Sending the following 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 causes the oscilloscope to perform an internal self-calibration and generates a response that indicates whether or not your Waverunner oscilloscope completed the calibration without error. This internal calibration sequence is the same as that which occurs at power-up. At the end of the calibration, after the response has indicated how the calibration terminated, the oscilloscope returns to the state it was in just prior to the calibration cycle.

QUERY SYNTAX

*CAL?

RESPONSE FORMAT

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

EXAMPLE (GPIB)

The following forces a self-calibration:

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

Response message (if no failure): *CAL 0

RELATED COMMANDS

AUTO_CALIBRATE

MISCELLANEOUS

CAL_OUTPUT, COUT

Command/Query

DESCRIPTION

The CAL_OUTPUT command is used to set the type of signal put out at the Waverunner front panel CAL connector.

COMMAND SYNTAX

Cal_OUTput <mode>[, <level>[, <rate>]]

<mode> : = {CALSQ, LEVEL}
<level> : = –1.0 to 1.00 V into 1 MΩ
<rate> : = 500 Hz to 1 MHz.

QUERY SYNTAX

Cal_OUTput?

RESPONSE FORMAT

Cal_OUTput <mode>,<level>[,<rate>]

EXAMPLE (GPIB)

The following sets the calibration signal to give a 0–0.2 volt pulse of 25 ns width at a 10 kHz rate:

```
CMD$="COUT PULSE,0.2 V,10 kHz":
CALL IBWRT(SCOPE%,CMD$)
```

ADDITIONAL INFORMATION

NOTATION	
CALSQ	Provides a square signal
LEVEL	Provides a DC signal at the requested level

DISPLAY

CALL_HOST, CHST

Command/Query

DESCRIPTION

The CALL_HOST command allows you 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 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

FUNCTION

CLEAR_MEMORY, CLM
Command

DESCRIPTION

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

COMMAND SYNTAX

CLear_Memory < memory>

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

EXAMPLE (GPIB)

The following clears the memory M2.

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

RELATED COMMANDS

STORE

FUNCTION

CLEAR_SWEEPS, CLSW
Command

DESCRIPTION

The CLEAR_SWEEPS command restarts the cumulative processing functions: summed or continuous average, extrema, FFT power average, histogram, pulse parameter statistics, Pass/Fail counters, and persistence.

COMMAND SYNTAX

```
CLear SWeeps
```

EXAMPLE (GPIB)

The following example will restart the cumulative processing:

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

RELATED COMMANDS

DEFINE, INR

STATUS

*CLS
Command

DESCRIPTION

The *CLS command clears all status data registers.

COMMAND SYNTAX

```
*CLS
```

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

ALL_STATUS, CMR, DDR, *ESR, EXR, *STB, URR

STATUS

CMR?
Query

DESCRIPTION

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

QUERY SYNTAX

CMR?

RESPONSE FORMAT

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

EXAMPLE (GPIB)

The following reads the contents of the CMR register:

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

RELATED COMMANDS

ALL_STATUS?, *CLS

ADDITIONAL INFORMATION

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, COLR
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 the user schemes U1, U2, U3 or 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, LTGEEN, JADE, LMGREEN, APGREEN, EMGREEN, GRGREEN, OCSPRAY, ICEBLUE, PASTBLUE, PALEBLUE, SKYBLUE, ROYLBLUE, 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>]

EXAMPLE (GPIB)

The following 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

ADDITIONAL INFORMATION

NOTATION			
<color>	**Color**	**<color>**	**Color**
WHITE	White	OCSPRAY	Ocean Spray
CYAN	Cyan	ICEBLUE	Ice Blue
YELLOW	Yellow	PASTBLUE	Pastel Blue
GREEN	Green	PALEBLUE	Pale Blue
MAGENTA	Magenta	SKYBLUE	Sky Blue
BLUE	Blue	ROYLBLUE	Royal Blue
RED	Red	DEEPBLUE	Deep Blue
LTGRAY	Light Gray	NAVY	Navy
GRAY	Gray	PLUM	Plum
SLGRAY	Slate Gray	PURPLE	Purple
CHGRAY	Charcoal Gray	AMETHYST	Amethyst
DKCYAN	Dark Cyan	FUCHSIA	Fuchsia
CREAM	Cream	RASPB	Raspberry
SAND	Sand	NEONPINK	Neon Pink
AMBER	Amber	PALEPINK	Pale Pink
OLIVE	Olive	PINK	Pink
LTGREEN	Light Green	VERMIL	Vermilion
JADE	Jade	ORANGE	Orange
LMGREEN	Lime Green	CERISE	Cerise
APGREEN	Apple Green	KHAKI	Khaki

EMGREEN	Emerald Green	BROWN	Brown
GRGREEN	Grass Green	BLACK	Black
<object>	**Display Object**	**<object>**	**Display Object**
BACKGND	Background	CURSOR	cursors
C1..C4	Channel Traces	WARNING	Warning Messages
TA..TD	Function Traces	NEUTRAL	Neutral color
GRID	Grid lines	OVERLAYS	Menu background color in FULL SCREEN

DISPLAY

COLOR_SCHEME, CSCH

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>

EXAMPLE (GPIB)

The following selects the user color scheme U2:

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

RELATED COMMANDS

COLOR, PERSIST_COLOR

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 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>

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. There are three response formats: LONG, in which responses start with the long form of the header word; SHORT, where responses start with the short form of the header word; and OFF, for which headers are omitted from the response and suffix units in numbers are suppressed.

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

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

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

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

COMMAND SYNTAX

Comm_HeaDeR <mode>

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

QUERY SYNTAX

Comm_HeaDeR?

RESPONSE FORMAT

Comm_HeaDeR <mode>

EXAMPLE (GPIB)

The following code sets the response header format to SHORT:

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

RELATED COMMANDS

COMM_HELP_LOG

COMMUNICATION

COMM_HELP, CHLP

Command/Query

DESCRIPTION

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

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

DESCRIPTION

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

QUERY SYNTAX

Comm_HeLP_Log? [CLR]

RESPONSE FORMAT

Comm_Help_Log <string containing the logged text>

EXAMPLE (GPIB)

The following reads the remote control log and prints it:

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

RELATED COMMANDS

COMM_HELP

COMMUNICATION

COMM_ORDER, CORD

Command/Query

DESCRIPTION

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

The COMM_ORDER? query returns the byte transmission order in current use.

COMMAND SYNTAX

Comm_ORDer <mode>

<mode> : = {HI, LO}

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

QUERY SYNTAX

Comm_ORDer?

RESPONSE FORMAT

Comm_ORDer <mode>

EXAMPLE

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

TYPE	CORD HI	CORD LO
Word	<MSB><LSB>	<LSB><MSB>
Long or 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 COMM_RS232 command 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 and query is only valid when you control the oscilloscope remotely using the Waverunner RS-232-C port.

PARAMETERS

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.

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.

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.

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.

SRQ string: This string is sent each time the oscilloscope signals an SRQ to the host computer.

Some COMM_RS232 parameters require ASCII strings as actual arguments. In order to facilitate the embedding of non-printable characters into such strings, escape sequences can be used. The back-slash character ('\') is used as an escape character. The following escape sequences are recognized:

"\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":	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	
EI	End input character
EO	End output string
LL	Line length
LS	Line separator
SRQ	SRQ service request

COMMAND SYNTAX

COmm_RS232 EI,<ei_char>,EO,'<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

COmm_RS232?

RESPONSE FORMAT

COmm_RS232 EI,<ei_char>,EO,"p <eo_string>",LL,<line_length>,LS,<line_sep>,SRQ,"<srq_string>"

EXAMPLE

After executing the command

```
COMM_RS232 EI,3,EO, "\r\nEND\r\n"
```

the oscilloscope 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`, `C3`, `C4`, `EX`, `EX10`}

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

QUERY SYNTAX

<channel> : `CouPLing?`

RESPONSE FORMAT

<channel> : `CouPLing` <coupling>

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

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

AVAILABILITY

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

EXAMPLE (GPIB)

The following sets the coupling of Channel 2 to 50 Ω 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	
ABS	absolute reading of relative cursors
CUST	custom parameters
FAIL	Pass/Fail: fail
HABS	horizontal absolute cursors
HPAR	standard time parameters
HREL	horizontal relative cursors
OFF	cursors and parameters off
PARAM	synonym for VPAR
PASS	Pass/Fail: pass
SHOW	custom parameters (old form)
STAT	parameter statistics
VABS	vertical absolute cursors
VPAR	standard voltage parameters
VREL	vertical relative cursors

NOTE: The PARAM mode is turned OFF when 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}

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

The 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 switches on the vertical relative cursors:

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

The following determines which cursor is currently turned on:

```
CMDS$="CRMS?": CALL IBWRT(SCOPE%,CMD$):
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 these four forms:

```
CURSOR_MEASURE HABS
CURSOR_MEASURE HREL
CURSOR_MEASURE VABS
CURSOR_MEASURE VREL
```

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

```
CURSOR_MEASURE HABS

CURSOR_MEASURE HREL
CURSOR_MEASURE VABS
CURSOR_MEASURE VREL
CURSOR_MEASURE FAIL
```

To select parameters in the Custom mode, and to modify the test conditions in the Pass/Fail mode, use the command:

```
PASS_FAIL_CONDITION
```

CURSOR

CURSOR_SET, CRST

Command/Query

DESCRIPTION

The CURSOR_SET command allows you 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.

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 124). To change only the trace without repositioning the cursors, the CURSOR_SET command can be given with no argument (for example, TB:CRST).

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

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

COMMAND SYNTAX

<trace> : CuRsor_SeT <cursor>,<position>[,<cursor>,<position>,<cursor> ,<position>]

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

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

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

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

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

The suffix DIV is optional.

QUERY SYNTAX

<trace> : `CuRsor_SeT?` [<cursor>,...<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.

AVAILABILITY

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

EXAMPLE (GPIB)

The following 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	
HABS	horizontal absolute
HREL	horizontal relative

QUERY SYNTAX

<trace> : CuRsor_VAlue? [<mode>,...<mode>]

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

<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>

<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.

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

DISPLAY

DATA_POINTS, DPNT

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>

EXAMPLE (GPIB)

The following highlights the waveform sample points:

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

MISCELLANEOUS

DATE

Command/Query

DESCRIPTION

The DATE command changes the date/time of the oscilloscope's internal real-time clock.

The DATE? query returns the current date/time setting.

COMMAND SYNTAX

DATE <day>,<month>,<year>,<hour>,<minute>,<second>

<day> : = 1 to 31

<month> : = {JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC}

<year> : = 1990 to 2089

<hour> : = 0 to 23

<minute> : = 0 to 59

<second> : = 0 to 59

TIP: You need only specify those DATE parameters up to and including the parameter to be changed in order to change the `year` setting: `day`, `month` and `year`. The time settings will remain unchanged. But to change the `second` setting, all the DATE parameters must be specified with the required settings.

QUERY SYNTAX

DATE?

RESPONSE FORMAT

DATE <day>,<month>,<year>,<hour>,<minute>,<second>

EXAMPLE (GPIB)

This will change the date to January 1, 1997 and the time to 1:21:16 p.m. (13:21:16 in 24-hour notation):

```
CMD$="DATE 1,JAN,1997,13,21,16": CALL
IBWRT(SCOPE%,CMD$)
```

STATUS

DDR?
Query

DESCRIPTION

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

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

QUERY SYNTAX

DDR?

RESPONSE FORMAT

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

AVAILABILITY

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

EXAMPLE (GPIB)

The following 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 math tools and zoom in the standard oscilloscope as well as those in the Extended Math and WaveAnalyzer options. See the *Operator's Manual* for more about Waverunner Math Tools.

COMMAND SYNTAX

<function> : DEFine EQN,'<equation>'
[,<param_name>,<value>,...]

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

Space (blank) characters inside equations are optional.

QUERY SYNTAX

<function> : DEFine?

RESPONSE FORMAT

<function> : DEFine EQN,'<equation>'[,MAXPTS,<max_points>]
[,SWEEPS,<max_sweeps>][,WEIGHT,<weight>][,BITS,<bits>]

FUNCTION PARAMETERS

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

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

FUNCTION EQUATIONS AND NAMES	
NOTE: These are available according to the options installed in your Waverunner oscilloscope. See Chapter 5 of the Operator's Manual for math and waveform processing options.	
–<source>	Negation
+<source>	Identity
<source>	
<source1> - <source2>	Subtraction
<source1> + <source2>	Addition
<source1>/<source2>	Ratio
<source1><source2>	Multiplication
1/<source>	Reciprocal
ABS(<source>)	Absolute Value
AVGC(<source>)	Continuous Average
AVGS(<source>)	Average Summed
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)

NOTE: For FFT functions, the source must be a time-domain, single-segment waveform.	
FFT(<source>)	Fast Fourier Transform (complex result)
FLOOR(EXTR(<source>))	Floor (Extrema source only)
HIST(<custom_line>)	Histogram of parameter on custom line
IMAG(FFT(<source>))	Imaginary part of complex result
INTG(<source>[{+,–} <addend>])	Integral
LN(<source>)	Logarithm base e
LOG10(<source>)	Logarithm base 10
MAG(AVGP(<function>))	FFT power average of magnitude
NOTE: For FFT Average functions, the source waveform must also be defined as an FFT function.	
MAG(FFT(<source>))	Magnitude of complex result
PHASE(FFT(<source>))	Phase angle (degrees) of complex result
PS(AVGP(<function>))	FFT Average of power spectrum
PS(FFT(<source>))	Power spectrum
PSD(AVGP(<function>))	FFT power average of power density
PSD(FFT(<source>))	Power density
REAL(FFT(<source>))	Real part of complex result
RESC([{+,–}][<multiplier>*]<source>[{+,–} <addend>])	Rescale
ROOF(EXTR(<source>))	Roof (Extrema source only)
SINX(<source>)	Sin(x)/x interpolator
SQR(<source>)	Square
SQRT(<source>)	Square Root
ZOOMONLY (<extended_source>)	Zoom only (No Math)

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

SOURCE VALUES

<sourceN> : = {TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, C3, C4}

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

<custom_line> : = {CUST1, CUST2, CUST3, CUST4, CUST5}

<extended_source> : = {C1, C2, C3,C4, TA, TB, TC, TD, M1, M2, M3, M4}

VALUES TO DEFINE NUMBER OF POINTS/SWEEPS:

<max_points> : = 50 to 10 000 000

<max sweeps> : = 1 to 1000

<max_sweeps> : = 1 to 1 000 000 (with WaveAnalyzer only)

<max_sweeps> : = 1 to 50 000

VALUES FOR RESCALE FUNCTION:

<addend> : = 0.0 to 1e15

<multiplier> : = 0.0 to 1e15

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

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

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 WINDOWS:

<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

HISTOGRAM VALUES

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

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

 : = –1e15 to 1e15

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

<vert_scale> : = {LIN, LOG, CONSTMAX}

HISTOGRAM NOTATION	
LIN	Use linear vertical scaling for histogram display
LOG	Use log vertical scaling for histogram display
CONSTMAX	Use constant maximum linear scaling for histogram display

PRML CORRELATION VALUES

<length> : = 0 to 10 divisions

<start> : = 0 to 10 divisions

AVAILABILITY

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

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

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

NOTE: The pair SWEEPS,<max_sweeps> applies only to the summed averaging (AVGS).

EXAMPLES (GPIB)

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

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

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

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

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

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

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

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

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

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

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

CMD$="TC:DEF
EQN,'HIST(CUST2)',VERT,LIN,MAXBINS,100":

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

RELATED COMMANDS

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

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, FLPY, HDD}

<filename> : = An alphanumeric string of up to eight characters, followed by a dot and an extension of up to three characters.

AVAILABILITY

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

<device> : HDD available only when removable Hard Disk Drive option is fitted.

EXAMPLE (GPIB)

The following 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> : = {CARD, FLPY, HDD}

<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.

AVAILABILITY

<device> : CARD available only with the Memory Card option installed.

<device> : HDD available only with the removable Hard Disk option installed.

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 of04-MAR-1994
  10:46:20 on Memory Card
SC1         000    2859         19-DEC-199416:33:06
SC1         001    2859         19-DEC-199416:34:32
TEST5       002    20359        12-MAR-199413:34:12
3 File(s) 1948672 bytes free
"
```

DISPLAY

DISPLAY, DISP
Command/Query

DESCRIPTION

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

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

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

COMMAND SYNTAX

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

QUERY SYNTAX

DISPlay?

RESPONSE FORMAT

DISPlay <state>

EXAMPLE (GPIB)

The following turns off the display:

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

DISPLAY

DOT_JOIN, DTJN

Command/Query

DESCRIPTION

The DOT_JOIN command controls the interpolation lines between data points.

COMMAND SYNTAX

DoT_JoiN <state>

<state> : = {ON, OFF}

QUERY SYNTAX

DoT_JoiN?

RESPONSE FORMAT

DoT_JoiN <state>

EXAMPLE (GPIB)

The following turns off the interpolation lines:

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

DISPLAY

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 turns dual zoom on:

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

RELATED COMMANDS

`HOR_MAGNIFY, HOR_POSITION, MULTI_ZOOM`

STATUS

ENABLE_KEY
Command/Query

DESCRIPTION

The ENABLE_KEY command allows you to assign menus to the lower six menu buttons for use in local mode.

COMMAND SYNTAX

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

QUERY SYNTAX

ENABLE_KEY?

RESPONSE FORMAT

ENABLE_KEY <state>

EXAMPLE (GPIB)

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

RELATED COMMANDS

KEY

STATUS

*ESE

Command/Query

DESCRIPTION

The *ESE command sets the Standard Event Status Enable register (ESE). This command allows one or more events in the ESR register to be reflected in the ESB summary message bit (bit 5) of the STB register. For an overview of the ESB defined events refer to the ESR table on page 121.

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

COMMAND SYNTAX

*ESE <value>

<value> : = 0 to 255

QUERY SYNTAX

*ESE?

RESPONSE FORMAT

*ESE <value>

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

*ESR

STATUS

*ESR?
Query

DESCRIPTION

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

QUERY SYNTAX

```
*ESR?
```

RESPONSE FORMAT

```
*ESR <value>
```

<value> : = 0 to 255

EXAMPLE (GPIB)

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

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

Response message:

```
*ESR 0
```

RELATED COMMANDS

ALL_STATUS, *CLS, *ESE

ADDITIONAL INFORMATION

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

NOTE: (refer to table above)

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 can 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?
Query

DESCRIPTION

The EXR? query reads and clears the contents of the EXecution error Register (EXR). The EXR register specifies the type of the last error detected during execution. Refer to the table next page.

QUERY SYNTAX

```
EXR?
```

RESPONSE FORMAT

```
EXR <value>
```
<value> : = 21 to 64

EXAMPLE (GPIB)

The following reads the contents of the EXR register:

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

Response message (if no fault):

```
EXR 0
```

RELATED COMMANDS

```
ALL_STATUS, *CLS
```

ADDITIONAL INFORMATION

EXECUTION ERROR STATUS REGISTER STRUCTURE (EXR)	
Value	**Description**
21	Permission error. The command cannot be executed in local mode.
22	Environment error. The oscilloscope is not configured to correctly process a command. For instance, the oscilloscope cannot be set to RIS at a slow timebase.
23	Option error. The command applies to an option which has not been installed.
24	Unresolved parsing error.
25	Parameter error. Too many parameters specified.
26	Non-implemented command.
30	Hex data error. A non-hexadecimal character has been detected in a hex data block.
31	Waveform error. The amount of data received does not correspond to descriptor indicators.
32	Waveform descriptor error. An invalid waveform descriptor has been detected.
33	Waveform text error. A corrupted waveform user text has been detected.
34	Waveform time error. Invalid RIS or TRIG time data has been detected.
35	Waveform data error. Invalid waveform data have been detected.
36	Panel setup error. An invalid panel setup data block has been detected.
50	No mass storage present when user attempted to access it. *
51	Mass storage not formatted when user attempted to access it. *
53	Mass storage was write protected when user attempted to create, 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. *

* Only with memory card or removable hard disk option.

DISPLAY

FAT_CURSOR, FATC

Command/Query

DESCRIPTION

FAT_CURSOR controls the width of the cursors.

COMMAND SYNTAX

FAT_CURSOR,FATC <state>
<state> : = {ON, OFF}

QUERY SYNTAX

FAT_CURSOR?

RESPONSE FORMAT

FAT_CURSOR <state>

EXAMPLE (GPIB)

The following sets the cursor appearance to fat:

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

MASS STORAGE

FILENAME, FLNM

Command/Query

DESCRIPTION

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.

COMMAND SYNTAX

FiLeNaMe TYPE,<type>,FILE,'<filename>'

<type> : = {C1, C2, C3, C4, TA, TB, TC, TD, SETUP, HCOPY }

<filename> : = For C1 to TD, an alphanumeric string of up to eight characters forming a legal DOS filename. Up to five characters for SETUP and HCOPY.

NOTE: No extension can be specified, as the oscilloscope automatically does this.

QUERY SYNTAX

FiLeNaMe? TYPE,<type>
<type> : = {ALL, C1, C2, C3, C4, TA, TB, TC, TD, SETUP, HCOPY}

RESPONSE FORMAT

FiLeNaMe
TYPE,<type>,FILE,"<filename>"[,TYPE,<type>,FILE,"<filename>"...]

AVAILABILITY

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

EXAMPLE (GPIB)

The following designates channel 1 waveform files as TESTPNT6.xxx" where xxx is a numeric extension assigned by the scope:

```
CMD$="FLNM TYPE,C1, FILE, 'TESTPNT6'":
CALL IBWRT(SCOPE%,CMD$)
```

RELATED COMMANDS

DIRECTORY, FORMAT_CARD, FORMAT_FLOPPY, FORMAT_HDD, DELETE_FILE

FUNCTION

FIND_CTR_RANGE, FCR
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}

AVAILABILITY

Only available with an option installed that includes Histograms.

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 PCMIA/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}

AVAILABILITY

Available only with the Memory Card option installed.

EXAMPLE (GPIB)

The following 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

ADDITIONAL INFORMATION

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

MASS STORAGE

FORMAT_FLOPPY, FFLP

Command/Query

DESCRIPTION

The FORMAT_FLOPPY command formats a floppy disk in the Double Density or High Density format.

The FORMAT_FLOPPY? query returns the status of the floppy disk.

COMMAND SYNTAX

Format_FLoPpy [<type>]

<type> : = {DD, HD}

If no argument is supplied, HD is used by default.

QUERY SYNTAX

Format_FLoPpy?

RESPONSE FORMAT

Format_FloPpy <floppy_status>[,<read/write>,<free_space>,<floppy_size>]

<floppy_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 floppy.

<floppy_size> : = A decimal number giving the total number of bytes on the floppy.

EXAMPLE (GPIB)

The following will first format a floppy in the Double Density (720 kB) format and then verify its status:

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

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

Response message:

```
FFLP OK,RW,728064,737280,
```

RELATED COMMANDS

DIRECTORY

ADDITIONAL INFORMATION

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

AVAILABILITY

Available only when the removable hard disk option is fitted.

EXAMPLE (GPIB)

The following 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

ADDITIONAL INFORMATION

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

DISPLAY

FULL_SCREEN, FSCR

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>

EXAMPLE (GPIB)

The following enables the Full Screen presentation format:

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

FUNCTION

FUNCTION_RESET, FRST
Command

DESCRIPTION

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

COMMAND SYNTAX

<function> : Function_ReSeT

EXAMPLE (GPIB)

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

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

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

RELATED COMMANDS

DEFINE, INR

ACQUISITION

GLOBAL_BWL, GBWL

Command/Query

DESCRIPTION

The GLOBAL_BWL command turns on or off the Global Bandwidth Limit. When on, the selected bandwidth limit will apply to all channels; when off, a bandwidth limit can be set individually for each channel (see BWL, page 69). The response to the GLOBAL_BWL? query indicates whether the Global Bandwidth Limit is on or off.

COMMAND SYNTAX

Global_BWL <mode>

<mode> : = {OFF, ON}

QUERY SYNTAX

Global_BWL?

RESPONSE FORMAT

Global_BWL <mode>

EXAMPLE

The following deactivates the Global Bandwidth Limit, allowing a Bandwidth Limit to be set individually for each channel (using the BWL command syntax for individual channels):

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

RELATED COMMANDS

BANDWIDTH_LIMIT

DISPLAY

GRID

Command/Query

DESCRIPTION

The GRID command defines the style of the grid used in the display.

The GRID? query returns the grid style currently in use.

COMMAND SYNTAX

GRID <grid>

In standard display:

<grid> : = {SINGLE, DUAL, QUAD, OCTAL}

In XY display:

<grid> : = {SINGLE, DUAL, XYONLY}

QUERY SYNTAX

GRID?

RESPONSE FORMAT

GRID <grid>

EXAMPLE (GPIB)

The following sets the screen display to dual grid mode:

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

HARD COPY

HARDCOPY_SETUP, HCSU

Command/Query

DESCRIPTION

The HARDCOPY_SETUP command configures the oscilloscope's hardcopy driver. It enables you to specify the device type and transmission mode of the hardcopy 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, EPSONCOL, HPDJ, HPDJBW, HPPJ, HPTJ, HPLJ, HP7470A, HP7550A, TIFF, TIFFCOL TIFFCOMP}

<port> : = {GPIB, RS, CENT, FLPY, CARD, HDD, PRT}

<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>

AVAILABILITY

<card> : CARD only with Memory Card option installed.

<port> : HDD only with removable Hard Disk option installed.
<port> : PRT only with internal graphics printer installed.
<cmdiv> only with internal graphics printer installed.
<auto> only with internal graphics printer installed.
<device> See table below.

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

The table below lists the printer and graphic formats you can use for producing hardcopies remotely using <device>.

NOTATION	PRINTER, PLOTTER OR PROTOCOL
BMP	BMP
BMPCOMP	BMP compressed
CANONCOL	Canon 200/600/800 Series color printers
EPSON	Epson b & w
EPSONCOL	Epson color
HPLJ	HP LaserJet
HPDJ	HP Desk Jet color
HPDJBW	HP Desk Jet b & w
HP7470A	HP 7470A plotter
HP7550A	HP 7550A plotter
HPGL	Vector screen file
TIFF	TIFF
TIFFCOL	TIFF color

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

HARDCOPY COMMAND NOTATION	
DEV	Device
PENS	Plotter: plot pens
PFEED	Page feed
PORT	Transmission mode
CARD	Memory Card
HDD	Hard Disk
CENT	Centronics port
FLPY	Floppy disk
GPIB	IEEE-488 port
PRT	Internal printer
RS	RS-232-C port
CMDIV	Internal printer: cm/division
PSIZE	Plotter: paper size
AUTO	Auto print
FORMAT	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 hardcopy unit. This allows you to control the hardcopy 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 hardcopy unit the escape sequence is converted to the ASCII character code.

EXAMPLE (GPIB)

The following 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 210) 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 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 210) 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>,
<exp_trace> : = {TA, TB, TC, TD}
<hor_position> : = 0 to 10 DIV
 : = 0 to max segments

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

The suffix DIV is optional.

QUERY SYNTAX <exp_trace> : Hor_POSition?

RESPONSE FORMAT <exp_trace> : Hor_POSition <hor_position>[,]

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

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

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

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 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

Command/Query

DESCRIPTION

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

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

COMMAND SYNTAX

INE <value>

<value> : = 0 to 65535

QUERY SYNTAX

INE?

RESPONSE FORMAT

INE <value>

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

INR?

STATUS

INR?
Query

DESCRIPTION

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

INTERNAL STATE REGISTER STRUCTURE (INR)

Bit	Bit Value	Description	
15		0	Reserved for future use.
14	16384	1	Probe was changed.
13	8192	1	Trigger is ready.
12	4096	1	Pass/Fail test detected desired outcome.
11	2048	1	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 in acquisition memory but not yet read out into the main memory.
3	8	1	A time-out has occurred in a data block transfer.
2	4	1	A return to the local state is detected.
1	2	1	A screen dump has terminated.
0	1	1	A new signal has been acquired in acquisition memory and read out into the main memory.

QUERY SYNTAX

INR?

RESPONSE FORMAT

INR <state>

<state> : = 0 to 65535

EXAMPLE (GPIB)

The following reads the contents of the INR register:

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

Response message:

```
INR 1026
```

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

RELATED COMMANDS

ALL_STATUS, *CLS, INE

WAVEFORM TRANSFER

INSPECT?, INSP?

Query

DESCRIPTION

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

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

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

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

QUERY SYNTAX

<trace> : INSPect? '<string>'[,<data_type>]

<trace> : = {TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, C3, C4}

<string> : =

<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.

RESPONSE FORMAT

<trace> : INSPect "<string>"

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

AVAILABILITY

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

EXAMPLES (GPIB)

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

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

Response message:

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

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

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

RELATED COMMANDS

TEMPLATE, WAVEFORM_SETUP

DISPLAY

INTENSITY, INTS

Command/Query

DESCRIPTION

The INTENSITY command sets the intensity level of the grid, or the trace or text.

The intensity level is expressed as a percentage (PCT). A level of 100 PCT corresponds to the maximum intensity while a level of 0 PCT sets the intensity to its minimum value.

The response to the INTENSITY? query indicates the grid and trace intensity levels.

COMMAND SYNTAX

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

<value> : = 0 to 100 [PCT]

NOTE: Parameters are grouped in pairs. The first of the pair names the variable to be modified, while the second gives the new value to be assigned. Pairs can be given in any order and be restricted to those variables to be changed.

The suffix PCT is optional.

QUERY SYNTAX

INTenSity?

RESPONSE FORMAT

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

EXAMPLE (GPIB)

The following enables remote control of the intensity, and changes the grid intensity level to 75 %:

```
CMD$="INTS GRID,75": CALL IBWRT(SCOPE%,CMD$)
```

ACQUISITION

INTERLEAVED, ILVD

Command/Query

DESCRIPTION

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

RIS is not available for sequence mode acquisitions.

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

COMMAND SYNTAX

InterLeaVeD <mode>

<mode> : = {ON, OFF}

QUERY SYNTAX

InterLeaVeD?

RESPONSE FORMAT

InterLeaVeD <mode>

EXAMPLE

The following instructs the oscilloscope to use RIS mode:

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

RELATED COMMANDS

TIME_DIV, TRIG_MODE, MEMORY_SIZE

STATUS

*IST?
Query

DESCRIPTION

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

QUERY SYNTAX

```
*IST?
```

RESPONSE FORMAT

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

EXAMPLE (GPIB)

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

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

Response message

```
*IST 0
```

RELATED COMMANDS

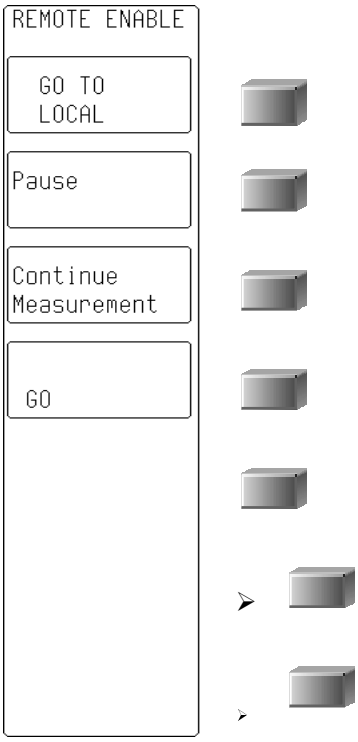
*PRE

DISPLAY

KEY

Command

DESCRIPTION



The KEY command allows control of a program from the Waverunner front panel. You can display one or two lines of 13 characters each in menus corresponding to the lower six menu buttons (soft keys). The top menu is reserved for GO TO LOCAL.

You can also assign an executable file, such as AUTOEXEC.DSO, to a menu button. AUTOEXEC.DSO is a text file containing remote commands. If the file is present, the scope will read the file from the floppy or memory card and execute the remote commands contained in it.

String text that you assign to these menus disappears when you switch to local, but reappears when you switch back to the remote state. Text is cleared at power-up, whenever you reset the oscilloscope, or when you assign an empty string to a location. For example: `KEY 2, ' '`.

Pressing any one of the menu buttons 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.

COMMAND SYNTAX

KEY <button>,'<string>','<string>'

➢ KEY <button>,'<string>','<string>','<file_name>'

<button> : = 1 to 5
<string> : = Up to two 13-character strings (any ASCII code)
<file_name> : = autoexec.dso

EXAMPLE (GPIB)

The menus illustrated this page were created by issuing the following:

```
CMD$="KEY 2, 'Pause'; KEY 3,
'Continue','Measurement';
KEY 4, ' ', 'GO'": CALL IBWRT(SCOPE%,CMD$)
```

RELATED COMMANDS

URR

DISPLAY

LOGO

Command/Query

DESCRIPTION

The LOGO command controls the placement 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>

EXAMPLE (GPIB)

The following turns on the logo:

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

MASK

Command/Query

DESCRIPTION

For PolyMask:

MASK COLOR allow 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 polygram 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 colors>,<error color>

See the table of color choices under 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

EXAMPLE (GPIB)

```
TA:MASK COLOR,BLUE,RED
```

DISPLAY

MEASURE_GATE, MGAT

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>

EXAMPLE (GPIB)

The following 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 the specifications in the Operator's Manual.

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

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

COMMAND SYNTAX

Memory_SIZe <size>

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

<size> : = {500, 1e+3, …, 2e+6, 4e+6, 8e+6}, for example, in standard numeric format.

Or, alternatively:

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

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

QUERY SYNTAX

Memory_SIZe? [NUM]

RESPONSE FORMAT

Memory_SIZe <size>

EXAMPLE

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

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

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

RELATED COMMANDS

TDIV

DISPLAY

MESSAGE, MSG

Command/Query

DESCRIPTION

The MESSAGE command displays a string of characters in the Message Field above the grid. The string can be up to 49 characters in length. The string is displayed as long as the oscilloscope 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 oscilloscope is reset, or if an empty string is sent (MSG " ").

The MESSAGE? query allows you to read the last message sent.

COMMAND SYNTAX

MeSsaGe '<string>'

<string> : = A string of a maximum of 49 characters

QUERY SYNTAX

MeSsaGe?

RESPONSE FORMAT

MeSsaGe "<string>"

EXAMPLE (GPIB)

The following causes the message Connect Probe 1 to appear in the message field:

```
CMD$="MSG '*Connect Probe 1*'": CALL
IBWRT(SCOPE%,CMD$)
```

DISPLAY

MULTI_ZOOM, MZOM

Command/Query

DESCRIPTION

With 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 Operator's Manual.

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

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

The suffix V is optional.

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

COMMAND SYNTAX

<channel> : OFfSeT <offset>

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

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

QUERY SYNTAX

<channel> : OFfSeT?

RESPONSE FORMAT

<channel> : OFfSeT <offset>

AVAILABILITY

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

EXAMPLE (GPIB)

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

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

STATUS

*OPC

Command/Query

DESCRIPTION

The *OPC (OPeration Complete) command sets to true the OPC bit (bit 0) in the standard Event Status Register (ESR). This command has no other effect on the operation of the oscilloscope because the oscilloscope starts parsing a command or query only after it has completely processed the previous command or query.

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.

COMMAND SYNTAX

```
*OPC
```

QUERY SYNTAX

```
*OPC?
```

RESPONSE FORMAT

```
*OPC 1
```

RELATED COMMANDS

```
*WAI
```

MISCELLANEOUS

*OPT?
Query

DESCRIPTION

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

QUERY SYNTAX

*OPT?

RESPONSE FORMAT

*OPT <option_1>,<option_2>,..,<option_N>

<option_n> : = A three- or four-character ASCII string

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

EXAMPLE (GPIB)

The following queries the installed options:

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

If, for example, the waveform processing options EMM, GP02, JTA, MC04 and WAVA and are installed, the response will be returned as:

```
*EMM, GP02, JTA, MC04, WAVA
```

Response message if no options are installed:

```
*OPT 0
```

ADDITIONAL INFORMATION

NOTATION	
GP02	Internal Printer/Centronics Option
HD02	Hard Disk Option
JTA	Jitter and Timing Analysis Option
MC02	PC Card Slot
EMM	Extended Math and Measurement Option
WAVA	WaveAnalyzer Option (includes histograms)

SAVE/RECALL SETUP

PANEL_SETUP, PNSU
Command/Query

DESCRIPTION

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

Only setup data read by the PNSU? query can be recalled into the oscilloscope. A panel setup error (see table on page 124) will be generated if the setup data block contains invalid data.

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

COMMAND SYNTAX

PaNel_SetUp <setup>

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

QUERY SYNTAX

PaNel_SetUp?

RESPONSE SYNTAX

PaNel_SetUp <setup>

EXAMPLE (GPIB)

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

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

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

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

RELATED COMMANDS

*RCL, *SAV

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

```
PArameter_CLear
```

RELATED COMMANDS

PARAMETER_DELETE, PARAMETER_VALUE, PASS_FAIL_CONDITION

CURSOR

PARAMETER_CUSTOM, PACU

Command/Query

DESCRIPTION

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

TIP: Use PAVA? to read the measured value of a parameter setup with PACU.

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
CUSTOMIZABLE PARAMETERS ON ALL MODELS		
DDLY	delta delay	<source1>,<source2>
PHASE	phase difference	<source1>,<edge1>,<level1>,<source2>,<edge2>,<level2>, <hysteresis>,<angular unit>
CUSTOMIZABLE PARAMETERS WITH EXTENDED MATH OPTION		
DTLEV	delta time at level	<source1>,<slope1>,<level1>,<source2>,<slope2>,<level2>, <hysteresis>
FLEV	fall at level	<source>,<high>,<low>
RLEV	rise at level	<source>,<low>,<high>
TLEV	time at level	<source>,<slope>,<level>,<hysteresis>
CUSTOMIZABLE PARAMETERS WITH WAVEANALYZER OPTION		
FWXX	full width at xx % of max	<source>,<threshold>
PCTL	percentile	<source>,<threshold>
XAPK	x position at peak	<source>,<rank>

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>,...]

AVAILABILITY

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

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 **DDLY**

Command Example:
```
PACU 2,DDLY,C1,C2
```

Query/Response Examples:
```
PACU? 2 returns:
PACU 2,DDLY,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		
1	line 1	of Custom or Pass/Fail display
2	line 2	of Custom or Pass/Fail display
3	line 3	of Custom or Pass/Fail display
4	line 4	of Custom or Pass/Fail display
5	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 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	
AVG	average
CUST	custom parameters
HIGH	highest value
HPAR	horizontal standard parameters
LOW	lowest value
PARAM	parameter definition for each line
SIGMA	sigma (standard deviation)
SWEEPS	number of sweeps accumulated for each line
VPAR	vertical standard parameters

QUERY SYNTAX

PArameter_STatistics? <mode>, <result>

<mode> : = {CUST, HPAR, VPAR}

<result> : = {AVG, LOW, HIGH, 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

CURSOR

PARAMETER_VALUE?, PAVA?

Query

DESCRIPTION

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

AVAILABLE ON ALL MODELS					
ALL	all parameters	DUTY	duty cycle	PER	period
AMPL	amplitude	FALL	falltime	PHASE	phase difference
AREA	area	FALL82	fall 80 to 20 %	PNTS	points
BASE	base	FREQ	frequency	RISE	risetime
CMEAN	mean for cyclic waveform	MAX	maximum	RISE28	rise 20 to 80 %
CRMS	root mean square for cyclic part of waveform	MEAN	mean	RMS	root mean square
CYCL	cycles	MIN	minimum	SDEV	standard deviation
DLY	delay	OVSN	negative overshoot	TOP	top
DUR	duration of acquisition	OVSP	positive overshoot	WID	width
CUSTOM PARAMETERS DEFINED USING PARAMETER_CUSTOM COMMAND*					
CUST1		CUST2		CUST3	CUST4 CUST5

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

AVAILABLE WITH EXTENDED MATH OR WAVEANALYZER OPTION					
AVG	average of distribution	HRMS	histogram rms value	PKS	number of peaks
CMEDI	median for cyclic waveform	HTOP	histogram top value	RANGE	range of distribution
CSDEV	standard deviation for cyclic part of waveform	LAST	last point	SIGMA	sigma of distribution
FRST	first point	MAXP	maximum population	TOTP	total population
HIGH	high of histogram	MEDI	median value		
HMEDI	median of a histogram	MODE	mode of distribution		

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

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

QUERY SYNTAX

<trace> : PArameter_VAlue? [<parameter>,...,<parameter>]

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

<parameter> : = See table of parameters.

Alternative forms of query for mask tests:

<trace> : PArameter_VAlue? <mask_test>, <mask>

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

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

RESPONSE FORMAT

<trace> : PArameter_VAlue <parameter>,<value>,

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

<value> : = A decimal numeric value

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

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

AVAILABILITY

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

EXAMPLE (GPIB)

The following query reads the 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: 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			
GT	greater than	LT	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, C3, C4}

<parameter> : = See tables of parameters on pages 168 and 173.

<rel_op> : = {GT, LT}

<ref_value> : = –1e15 to +1e15

NOTE: 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>.

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}

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

QUERY SYNTAX

PFCO? <line>

RESPONSE FORMAT

PFCO <line>,<trace>,<parameter>,<rel_op>,<ref_value>

Alternative form of response for mask tests:

PFCO <line>,<trace>,<mask_test>,<mask>

AVAILABILITY

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

EXAMPLE (GPIB)

The following sets the first test condition in the list to be "frequency on Channel 1 lower than 10 kHz":

```
CMD$="PFCO 1,C1,FREQ,LT,10000":
CALL IBWRT(SCOPE%,CMD$)
```

RELATED COMMANDS

CURSOR_MEASURE, CURSOR_SET, PASS_FAIL_COUNTER, PASS_FAIL_DO, PASS_FAIL_MASK, PARAMETER_VALUE

CURSOR

PASS_FAIL_COUNTER, PFCT

Command/Query

DESCRIPTION

The PASS_FAIL_COUNTER command resets the Passed/Failed acquisitions counters. The PASS_FAIL_COUNTER? query returns the current counts.

COMMAND SYNTAX

```
Pass_Fail_CounTer
```

QUERY SYNTAX

```
Pass_Fail_CounTer?
```

RESPONSE FORMAT

Pass_Fail_CounTer <pass/fail>,<value>,OF,<value>

<value> : = 0 to 999999

<pass/fail> : = {PASS, FAIL}

EXAMPLE (GPIB)

The following query reads the counters:

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

Response message:

```
PFCT PASS, 8, OF, 9
```

RELATED COMMANDS

CURSOR_MEASURE, CURSOR_SET, PASS_FAIL_DO, 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	
BEEP	emit a beep
PULS	emit a pulse on the CAL connector
SCDP	make a hard copy
STO	store in memory or on storage media
STOP	stop acquisition

COMMAND SYNTAX

Pass_Fail_DO [<outcome>[,<act>[,<act>...]]]

<outcome> : = {PASS,FAIL}

<act> : = {STOP, SCDP, STO}

NOTE:

BEEP is accepted only on models equipped with the CLBZ hardware option.

PULS is accepted only on models equipped with the CKIO software option.

PFDO without arguments deletes all actions.

STO performs the store operation as described in the Operator's Manual.

After every pass or fail detected, the oscilloscope sets the INR bit 12.

QUERY SYNTAX

Pass_Fail_DO?

RESPONSE FORMAT

Pass_Fail_DO [<pass_fail>[,<act>[,<act>...]]]

EXAMPLE (GPIB)

This following forces the oscilloscope to stop acquiring when the test passes:

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

RELATED COMMANDS

BUZZER, CURSOR_MEASURE, CURSOR_SET, INR, PARAMETER_VALUE, PASS_FAIL_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.

AVAILABILITY

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

EXAMPLE (GPIB)

The following 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

CURSOR

PASS_FAIL_STATUS?, PFST?

Query

DESCRIPTION

The PASS_FAIL_STATUS query returns the status of the Pass/Fail test for a given line number.

QUERY SYNTAX

Pass_Fail_STatus? <line>

<line> : = {1, 2, 3, 4, 5}

RESPONSE FORMAT

Pass_Fail_STatus <line>,<state>

<state> : = {TRUE, FALSE}

EXAMPLE (GPIB)

The following queries the state of the Pass/Fail test condition specified for line 3.

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

RELATED COMMANDS

PASS_FAIL_DO, PASS_FAIL_CONDITION, PARAMETER_VALUE

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 or cursors.

The vertical cursor positions are the same as those controlled by the CURSOR_SET command.

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

COMMAND SYNTAX

<trace> : PEr_Cursor_Set <cursor>,
<position>[,<cursor>,<position>,...,<cursor>,<position>

trace> : = {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: Parameters are grouped in pairs. The first of the pair names the variable to be modified, while the second gives the new value to be assigned. Pairs can be in any order and limited to those variables to be changed.

The suffix DIV is optional.

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.

QUERY SYNTAX

<trace> : PEr_Cursor_Set? <cursor>[,<cursor,...,<cursor>]

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

RESPONSE FORMAT

PEr_Cursor_Set <cursor>,<position>[,<cursor>,<position>,...,<cursor>,<position>

AVAILABILITY

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

EXAMPLE (GPIB)

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

```
CMD$="C2:PECS HREF,2.6 DIV,HDIF,7.4DIV"
```

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			
HABS	horizontal absolute	VABS	vertical absolute
HREL	horizontal relative	VREL	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>]

AVAILABILITY

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

EXAMPLE (GPIB)

The following 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

DISPLAY

PERSIST, PERS

Command/Query

DESCRIPTION

The PERSIST command enables or disables the persistence display mode.

COMMAND SYNTAX

PERSist <mode>

<mode> : = {ON, OFF}

QUERY SYNTAX

PERSist?

RESPONSE FORMAT

PERSist <mode>

EXAMPLE (GPIB)

The following turns the persistence display ON:

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

RELATED COMMANDS

PERSIST_COLOR, PERSIST_LAST, PERSIST_SAT, PERSIST_SETUP

DISPLAY

PERSIST_COLOR, PECL

Command/Query

DESCRIPTION

The PERSIST_COLOR command controls the color rendering method of persistence traces.

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

See the Operator's Manual.

COMMAND SYNTAX

PErsist_CoLor <state>

<state> : = {ANALOG, COLOR_GRADED}

QUERY SYNTAX

PErsist_CoLor?

RESPONSE FORMAT

PErsist_CoLor <state>

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

COLOR, COLOR_SCHEME, PERSIST, PERSIST_LAST, PERSIST_SAT, PERSIST_SETUP

DISPLAY

PERSIST_LAST, PELT

Command/Query

DESCRIPTION

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

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

COMMAND SYNTAX

PErsist_LasT <state>

<state> : = {ON, OFF}

QUERY SYNTAX

PErsist_LasT?

RESPONSE FORMAT

PErsist_LasT <state>

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

PERSIST, PERSIST_COLOR, PERSIST_SAT, PERSIST_SETUP

DISPLAY

PERSIST_SAT, PESA

Command/Query

DESCRIPTION

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

The response to the PERSIST_SAT? query indicates the saturation level of the persistence data maps.

COMMAND SYNTAX

PErsist_SAt <trace>,<value> [<trace>,<value>]

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

<value> : = 0 to 100 PCT

NOTE: The suffix PCT is optional.

QUERY SYNTAX

PErsist_SAt?

RESPONSE FORMAT

PErsist_SAt <trace>,<value>

AVAILABILITY

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

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

PERSIST, PERSIST_COLOR, PERSIST_PERS, PERSIST_SETUP

DISPLAY

PERSIST_SETUP, PESU

Command/Query

DESCRIPTION

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

The PERSIST_SETUP? query indicates the current status of the persistence.

COMMAND SYNTAX

PErsist_SetUp <time>,<mode>

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

<mode> : = {TOP2, ALL}

QUERY SYNTAX

PErsist_SetUp?

RESPONSE FORMAT

PErsist_SetUp <time>,<mode>

EXAMPLE (GPIB)

The following 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

STATUS

***PRE**
Command/Query

DESCRIPTION

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

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

COMMAND SYNTAX

PRE <value>

<value> : = 0 to 65 535

QUERY SYNTAX

*PRE?

RESPONSE FORMAT

*PRE <value>

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

*IST

PROBES

PROBE_CAL?, PRCA?

Query

DESCRIPTION

The PROBE_CAL? query performs a complete auto-calibration of a current probe connected to your Waverunner oscilloscope. At the end of this calibration, the response indicates how the calibration has terminated, and the oscilloscope then returns to the state it was in prior to the query.

QUERY SYNTAX

<channel> : PROBE_CAL?

RESPONSE FORMAT

PROBE_CAL <diagnostics>

<diagnostics> : = 0 or 1

0 = Calibration successful

EXAMPLE (GPIB)

The following forces a self-calibration:

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

Response message (if no failure): PROBE_CAL 0

RELATED COMMANDS

AUTO_CALIBRATE, *CAL?, PROBE_DEGAUSS?

PROBES

PROBE_DEGAUSS?, PRDG?

Query

DESCRIPTION

The PROBE_DEGAUSS? query performs the automatic degaussing of the current probe connected to your Waverunner oscilloscope. 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.

QUERY SYNTAX

<channel> : PROBE_DEGAUSS?

RESPONSE FORMAT

PROBE_DEGAUSS <diagnostics>

<diagnostics> : = 0 or 1

0 = Degaussing and calibration successful

(**Note:** If coupling is not DC, probe calibration will not be performed, and the <diagnostics> response will be 1.)

EXAMPLE (GPIB)

The following degausses and calibrates the connected probe:

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

Response message (if no failure):

```
PROBE_DEGAUSS 0
```

RELATED COMMANDS

PROBE_CAL?, PROBE_NAME?

PROBES

PROBE_NAME?, PRNA?

Query

DESCRIPTION

The PROBE_NAME? query returns the name of a probe connected to your Waverunner oscilloscope. Passive probes are identified by their attenuation factor.

QUERY SYNTAX

<channel> : PROBE_NAME?

RESPONSE FORMAT

PROBE_NAME <name>
<diagnostics> : = 0 or 1
0 = successful

EXAMPLE (GPIB)

The following obtains an identification of the connected probe:

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

RELATED COMMANDS

PROBE_CAL?, PROBE_DEGAUSS?

SAVE/RECALL SETUP

***RCL**
Command

DESCRIPTION

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

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

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

COMMAND SYNTAX

*RCL <panel_setup>

<panel_setup> : = 0 to 4

EXAMPLE (GPIB)

The following recalls your Waverunner oscilloscope setup previously stored in panel setup 3:

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

RELATED COMMANDS

PANEL_SETUP, *SAV, EXR

MISCELLANEOUS

REAR_OUTPUT, ROUT

Command/Query

DESCRIPTION

The REAR_OUTPUT command is used to set the type of signal put out at the Waverunner rear panel BNC connector. The REAR_OUTPUT? Query 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>]

EXAMPLE (GPIB)

The following turns off the BNC rear 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

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: 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.

AVAILABILITY

<device> : CARD only available with Memory Card option installed.

<device> : HDD only available with removable Hard Disk option installed.

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

<device> : = {CARD, FLPY, HDD}

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

AVAILABILITY

<device> : CARD only available with Memory Card option installed.

<device> : HDD only available with removable Hard Disk option installed.

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

SAVE/RECALL SETUP

***RST**
Command

DESCRIPTION

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

COMMAND SYNTAX

```
*RST
```

EXAMPLE (GPIB)

The following resets the oscilloscope:

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

RELATED COMMANDS

*CAL, *RCL

ACQUISITION

SAMPLE_CLOCK, SCLK

Command/Query

DESCRIPTION

The SAMPLE_CLOCK command allows you to control an external timebase. Set the number of data points that will be acquired when your Waverunner oscilloscope 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}

<recordlength> : = {10e+3, 10.0e+3, 11e+3…}, for example, in standard numeric format.

Or, alternatively:
= {50, 100, 200, 500, 1K, 2K, 5K, 10K, 20K, 50K, 100K, 200K, 500K, 1M, 2M}

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

<coupling> : = {D1M or D50}

TIP: You cannot have the record length larger than the maximum available memory of your oscilloscope. Waverunner will adapt to the closest valid <recordlength>. See the Operator's Manual for maximums.

QUERY SYNTAX

Sample_CLocK? [NUM]

RESPONSE FORMAT

Sample_CLocK <state>,<recordlength>

EXAMPLE

The following sets the oscilloscope 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 your Waverunner oscilloscope in non-volatile internal memory. The *SAV command stores the complete front panel setup of the oscilloscope at the time the command is issued.

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

COMMAND SYNTAX

*SAV <panel_setup>
<panel_setup> : = 1 to 4

EXAMPLE (GPIB)

The following saves the current Waverunner oscilloscope setup in panel setup 3:

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

RELATED COMMANDS

PANEL_SETUP, *RCL

HARD COPY

SCREEN_DUMP, SCDP

Command/Query

DESCRIPTION

The SCREEN_DUMP command causes the oscilloscope to dump the screen contents onto the hardcopy device. This command will halt your Waverunner oscilloscope's activities.

The time/date stamp which appears on the print-out corresponds to the time at which the command was executed.

COMMAND SYNTAX

```
SCreen_DumP
```

QUERY SYNTAX

```
SCreen_DumP?
```

RESPONSE FORMAT

SCreen_DumP <status>

<status> : = {OFF}

EXAMPLE (GPIB)

The following initiates a screen dump:

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

RELATED COMMANDS

INR, HARDCOPY_SETUP, HARDCOPY_TRANSMIT

DISPLAY

SCREEN_SAVE, SCSV

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>

EXAMPLE (GPIB)

The following enables the automatic screen saver:

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

DISPLAY

SELECT, SEL

Command/Query

DESCRIPTION

The SELECT command selects the specified trace for manual display control. An environment error (see table on page 124) is generated if the specified trace is not displayed.

The SELECT? query returns the selection status of the specified trace.

COMMAND SYNTAX

<trace> : SELect

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

QUERY SYNTAX

<trace> : SELect?

RESPONSE FORMAT

<trace> : SELect <mode>

<mode> : = {ON, OFF}

EXAMPLE (GPIB)

The following selects Trace B (TB):

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

RELATED COMMANDS

TRACE

ACQUISITION

SEQUENCE, SEQ

Command/Query

DESCRIPTION

The SEQUENCE command sets the conditions for the sequence mode acquisition. The response to the SEQUENCE? query gives the conditions for the sequence mode acquisition. The argument <max_size> can be expressed either as numeric fixed point, exponential or using standard suffixes. When the optional suffix NUM is used with the query, the response will be returned in standard numeric format.

COMMAND SYNTAX

SEQuence <mode>[,<segments>[,<max_size>]]

<mode> : = {OFF, ON}

<segments> : = the right-hand column in the table below:

MAX. MEMORY LENGTH PER CHANNEL	MAX. NUMBER OF SEGMENTS
10 000	50
25 000	50
50 000	200
100 000	500
200 000	500
250 000	500
500 000	2000
1 000 000	2000
2 000 000	2000

<max_size> : = {…10e+3, 10.0e+3,…11e+3,…}, for example, in standard numeric format.

Or, alternatively:

= {50, 100, 250, 500, 1000, 2500, 5K, 10K, 25K, 50K, 100K, 250K, 500K, 1M}

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

NOTE: The oscilloscope will adapt the requested <max_size> to the closest valid value.

QUERY SYNTAX

SEQuence? [NUM]

RESPONSE FORMAT

SEQuence <mode>,<segments>,<max_size>

<mode> : = {ON, OFF}

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

TRIG_MODE

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 is typically 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> : = time in seconds (0 to 1000.0)

EXAMPLE (GPIB)

The following command causes the scope to sleep for 3.2 seconds.

```
CMSD$="SLEEP 3.2";: CALL IBWRT(SCOPE%,CMD$)
```

RELATED COMMANDS

`*TRG,WAIT`

STATUS

*SRE

Command/Query

DESCRIPTION

The *SRE command sets the Service Request Enable register (SRE). This command allows you to specify which summary message bit or bits in the STB register will generate a service request. Refer to the table on page 210 for an overview of the available summary messages.

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

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

COMMAND SYNTAX

*SRE <value>

<value> : = 0 to 255

QUERY SYNTAX

*SRE?

RESPONSE FORMAT

*SRE <value>

EXAMPLE (GPIB)

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

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

STATUS

*STB?
Query

DESCRIPTION

The *STB? query reads the contents of the 488.1 defined 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 210 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 MSS = 1 RQS = 1	at least 1 bit in STB masked by SRE is 1 service is requested	1. 2.
5	32	ESB	1 an ESR enabled event has occurred	3.
4	16	MAV	1 output queue is not empty	4.
3	8	DIO3	0 reserved	
2	4	VAB	1 a command data value has been adapted	5.
1	2	DIO1	0 reserved	
0	1	INB	1 an enabled Internal state change has occurred	6.

NOTE: For the above table...

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

   Bit 6 = MSS if an *STB? query is received,

   = RQS if serial polling is conducted.

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

ACQUISITION

STOP
Command

DESCRIPTION

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

COMMAND SYNTAX

```
STOP
```

EXAMPLE

The following stops the acquisition process:

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

RELATED COMMANDS

ARM_ACQUISITION, TRIG_MODE, WAIT

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, C4, ALL_DISPLAYED}

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

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

AVAILABILITY

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

<dest> : CARD only available with Memory Card option installed.

<dest> : HDD only available with removable Hard Disk option installed.

EXAMPLE (GPIB)

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

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

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

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

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

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

RELATED COMMANDS

STORE_SETUP, RECALL

SAVE/RECALL SETUP

STORE_PANEL, STPN
Command

DESCRIPTION

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

NOTE: The communication parameters (the parameters modified by commands COMM_FORMAT, COMM_HEADER, COMM_HELP, COMM_ORDER and WAVEFORM_SETUP) and the enable registers associated with the status reporting system (commands *SRE, *PRE, *ESE, INE) are not saved by this command.

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

COMMAND SYNTAX

STore_PaNel DISK, <device>, FILE, '<filename>'

<device> : = {CARD, FLPY, HDD}

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

AVAILABILITY

<device> : CARD only available with Memory Card option installed.

<device> : HDD only available with removable Hard Disk option installed.

EXAMPLE (GPIB)

The following saves the current Waverunner oscilloscope 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 can be enabled for storage. This applies to both auto-storing and to the STORE command. Traces can 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: You can recall into the oscilloscope only waveforms stored in BINARY format.

COMMAND SYNTAX

STore_SeTup [<trace>, <dest>] [, AUTO, <mode>] [, FORMAT, <type>]

<trace> : = {TA, TB, TC, TD, C1, C2, C3, C4, ALL_DISPLAYED}
<dest> : = {M1, M2, M3, M4, CARD, FLPY, HDD}
<mode> : = {OFF, WRAP, FILL}
<type> : {BINARY, SPREADSHEET, MATHCAD, MATLAB }

QUERY SYNTAX

STore_SeTup?

RESPONSE FORMAT

STore_SeTup <trace>,<dest>,AUTO,<mode>

AVAILABILITY

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

<dest> : CARD only available with Memory Card option installed.

<dest> : HDD only available with removable Hard Disk option installed.

EXAMPLE (GPIB)

The following Channel 1 for storage. 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 your Waverunner oscilloscope'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.

For example, for revision 2.1, the file name generated will be LECROY21.TPL.

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

COMMAND SYNTAX

STore_TeMplate DISK,<device>

<device> : = {CARD, FLPY, HDD}

AVAILABILITY

<device> : CARD only available with the Memory Card option installed.

<device> : HDD only available with removable Hard Disk option installed.

EXAMPLE (GPIB)

The following 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

Command/Query

DESCRIPTION

The TDISP command controls the presence and format of the time and date display on the oscilloscope screen. The keyword CURRENT selects the current time and date for display. The keyword NONE removes both time and date. The keyword TRIGGER selects the trigger time of the uppermost waveform currently displayed.

COMMAND SYNTAX

`TDISP` <state>

<state> : = {`CURRENT,NONE,TRIGGER`}

QUERY SYNTAX

`TDISP?`

RESPONSE FORMAT

`TDISP` <state>

EXAMPLE (GPIB)

The following turns off the time and date display on the oscilloscope screen:

```
CMD$="TDISP NONE": CALL IBWRT(SCOPE%,CMD$)
```

WAVEFORM TRANSFER

TEMPLATE?, TMPL?

Query

DESCRIPTION

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

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

QUERY SYNTAX

TeMPLate?

RESPONSE FORMAT

TeMPLate "<template>"

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

RELATED COMMANDS

INSPECT

ACQUISITION

TIME_DIV, TDIV

Command/Query

DESCRIPTION

The TIME_DIV command modifies the timebase setting. The new timebase setting can be specified with suffixes: NS for nanoseconds, US for microseconds, MS for milliseconds, S for seconds, or KS for kiloseconds. An out-of-range value causes the VAB bit (bit 2) in the STB register (see table on page 210) to be set.

The TIME_DIV? query returns the current timebase setting.

COMMAND SYNTAX

Time_DIV <value>

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

The suffix S (seconds) is optional.

QUERY SYNTAX

Time_DIV?

RESPONSE FORMAT

Time_DIV <value>

EXAMPLE (GPIB)

The following sets the time base to 500 µsec/div:

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

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

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

RELATED COMMANDS

TRIG_DELAY, TRIG_MODE

DISPLAY

TRACE, TRA

Command/Query

DESCRIPTION

The TRACE command enables or disables the display of a trace. An environment error (see table on page 124) is set if an attempt is made to display more than four waveforms.

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

COMMAND SYNTAX

<trace> : TRAce <mode>

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

<mode> : = {ON, OFF}

QUERY SYNTAX

<trace> : TRAce?

RESPONSE FORMAT

<trace> : TRAce <mode>

AVAILABILITY

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

EXAMPLE (GPIB)

The following displays Trace A (TA):

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

DISPLAY

TRACE_OPACITY, TOPA

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>

EXAMPLE (GPIB)

The following allows traces to be distinguished even when they overlap:

```
CMD$="TOPA TRANSPARENT": CALL
BWRT(SCOPE%,CMD$)
```

ACQUISITION

***TRG**
Command

DESCRIPTION

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

COMMAND SYNTAX

```
*TRG
```

EXAMPLE (GPIB)

The following enables signal acquisition:

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

RELATED COMMANDS

`ARM_ACQUISITION, STOP, WAIT`

ACQUISITION

TRIG_COUPLING, TRCP

Command/Query

DESCRIPTION

The TRIG_COUPLING command sets the coupling mode of the specified trigger source.

The TRIG_COUPLING? query returns the trigger coupling of the selected source.

NOTE: The oscilloscope automatically determines trigger slope when in HFDIV coupling. The TRIG_SLOPE command is not used in HFDIV coupling mode.

HFDIV is indicated as HF in the trigger setup menus.

COMMAND SYNTAX

<trig_source> : TRig_CouPling <trig_coupling>

<trig_source> : = {C1, C2, C3, C4, EX, EX10}

<trig_coupling> : = {AC, DC, HFREJ, LFREJ}

QUERY SYNTAX

<trig_source> : TRig_CouPling?

RESPONSE FORMAT

<trig_source> : TRig_CouPling <trig_coupling>

AVAILABILITY

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

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

TRIG_COUPLING, TRIG_DELAY, TRIG_LEVEL, TRIG_MODE, TRIG_SELECT, TRIG_SLOPE, TRIG_WINDOW

ACQUISITION

TRIG_DELAY, TRDL

Command/Query

DESCRIPTION

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

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

If a value outside the range −10 000 div × time/div and 100 % is specified, the trigger time will be set to the nearest limit and the VAB bit (bit 2) will be set in the STB register.

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

COMMAND SYNTAX

TRig_DeLay <value>

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

−20 PS to −10 MAS (post-trigger)

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

QUERY SYNTAX TRig_DeLay?

RESPONSE FORMAT TRig_DeLay <value>

EXAMPLE (GPIB) The following sets the trigger delay to −20 s (post-trigger):

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

RELATED COMMANDS TIME_DIV, TRIG_COUPLING, TRIG_LEVEL, TRIG_MODE, TRIG_SELECT, TRIG_SLOPE, 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.

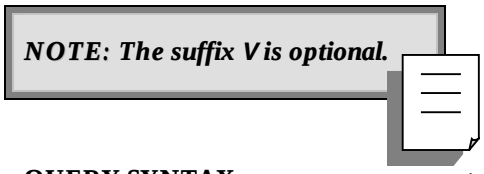
COMMAND SYNTAX

<trig_source> : TRig_LeVel <trig_level>

<trig_source> : = {C1, C2, C3, C4, EX, EX10}

<trig_level> : =

NOTE: The suffix V is optional.



QUERY SYNTAX

<trig_source> : TRig_LeVel?

RESPONSE FORMAT

<trig_source> : TRig_LeVel <trig_level>

AVAILABILITY

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

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

TRIG_COUPLING, TRIG_DELAY, TRIG_MODE, TRIG_SELECT, TRIG_SLOPE, TRIG_WINDOW

ACQUISITION

TRIG_MODE, TRMD

Command/Query

DESCRIPTION

The TRIG_MODE command specifies the trigger mode.

The TRIG_MODE? query returns the current trigger mode.

COMMAND SYNTAX

TRig_MoDe <mode>

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

QUERY SYNTAX

TRig_MoDe?

RESPONSE FORMAT

TRig_MoDe <mode>

EXAMPLE (GPIB)

The following selects the normal mode:

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

RELATED COMMANDS

ARM_ACQUISITION, STOP, TRIG_SELECT, SEQUENCE, TRIG_COUPLING, TRIG_LEVEL, TRIG_SLOPE, 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 can be given in any order and restricted to those variables to be changed.

The TRIG_SELECT? query returns the current trigger condition.

TRIGGER NOTATION			
DROP	Dropout	QL	Qualifier
EDGE	Edge	SNG	Single source
EV	Event	SQ	State-Qualified
GLIT	Glitch	SR	Source
HT	Hold type	STD	*Standard (Edge Trigger)
HV	Hold value	TEQ	Edge-Qualified
HV2	Second hold value	TI	Time
IL	Interval larger	TL	Time within
INTV	Interval	**TV**	**TV**
IS	Interval smaller	CHAR	Characteristics
I2	Interval-width window	FLD	Field
OFF	No hold-off on wait	FLDC	Field count
PL	Pulse larger	ILAC	Interlace
PS	Pulse smaller	LINE	Line
P2	Pulse-width window	LPIC	Lines per picture

COMMAND SYNTAX

For all but TV Trigger TRig_SElect <trig_type>,SR,<source>,QL,<source>,

HT,<hold_type>,HV,<hold_value>,HV2,<hold value>

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

<trig_type> : = {DROP, EDGE, GLIT, INTV, STD, SNG, SQ, TEQ}

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

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

<hold_value> : = See Operator's Manual for valid values

NOTE: The suffix S (seconds) is optional.

QUERY SYNTAX

TRig_SElect?

RESPONSE FORMAT

TRig_SElect
<trig_type>,SR,<source>,HT,<hold_type>,HV,
<hold_value>,<hold_value>

- HV2 only returned if <hold_type> is P2 or I2

AVAILABILITY

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

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
<trig_type>,SR,<source>,FLDC,<field_count>,FLD,<field>,
CHAR,<characteristics>,LPIC,<lpic>,ILAC,<ilace>,LINE,<line>

<trig_type> : = {TVP,TVN} TVP=TV Pos, TVN=TV Neg

<source> : = {C1, C2, C3, C4, NE, EX, EX10}

<field_count> : = {1, 2}

<field> : = 1 to field_count

<characteristics> : = {NTSC, PALSEC, CUST50, CUST60}

<lpic> : = 1 to 1500

<ilace> : = {1, 2}

<line> : = 1 to 1500 or 0 for any line

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
TVP,SR,<source>,FLDC,<field_count>,FLD,<field>,

CHAR,<characteristic>,LINE,<line>

AVAILABILITY

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

EXAMPLE (GPIB)

The following sets up the trigger system to trigger on the 3rd field, line 17, of the eight-field PAL/SECAM TV signal applied to the external input.

```
CMD$="TRSE TVN,SR,EX,FLDC,8,FLD,3,CHAR,PALSEC,
LINE,17": CALL IBWRT(SCOPE%,CMD$)
```

RELATED COMMANDS

TRIG_COUPLING, TRIG_DELAY, TRIG_LEVEL, 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 124) 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, C4, LINE, EX, EX10}

<trig_slope> : = {NEG, POS, WINDOW}

QUERY SYNTAX

<trig_source> : TRig_SLope?

RESPONSE FORMAT

<trig_source> : TRig_SLope <trig_slope>

AVAILABILITY

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

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

TRIG_COUPLING, TRIG_DELAY, TRIG_LEVEL, TRIG_MODE, TRIG_SELECT, TRIG_SLOPE, TRIG_WINDOW

ACQUISITION

TRIG_WINDOW, TRWI

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>

EXAMPLE

The following 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_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>

<status> : = 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 154). 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 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 enlarges the vertical amplitude of Trace A by a factor of 3.45 with respect to its original amplitude:

```
CMD$="TA:VMAG 3.45": CALL IBWRT(SCOPE%,CMD$)
```

RELATED COMMANDS

VERT_POSITION

DISPLAY

VERT_POSITION, VPOS

Command/Query

DESCRIPTION

The VERT_POSITION command adjusts the vertical position of the specified trace on the screen. It does not affect the original offset value obtained at acquisition time.

The VERT_POSITION? query returns the current vertical position of the specified trace.

COMMAND SYNTAX

<trace> : Vert_POSITION <display_offset>

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

<display_offset> : = –5900 to +5900 DIV

NOTE: The suffix DIV is optional. The limits depend on the current magnification factor, the number of grids on the display, and the initial position of the trace.

QUERY SYNTAX

<trace> : Vert_POSition?

RESPONSE FORMAT

<trace> : Vert_POSITION <display_offset>

EXAMPLE

The following shifts Trace A (TA) upwards by +3 divisions relative to the position at the time of acquisition:

```
CMD$="TA:VPOS 3DIV": CALL IBWRT(SCOPE%,CMD$)
```

RELATED COMMANDS

VERT_MAGNIFY

ACQUISITION

VOLT_DIV, VDIV

Command/Query

DESCRIPTION

The VOLT_DIV command sets the vertical sensitivity in Volts/div. The VAB bit (bit 2) in the STB register (see table on page 210) is set if an out-of-range value is entered.

The probe attenuation factor is not taken into account for adjusting vertical sensitivity.

The VOLT_DIV? query returns the vertical sensitivity of the specified channel.

COMMAND SYNTAX

<channel> : Volt_DIV <v_gain>

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

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

NOTE: The suffix V is optional.

QUERY SYNTAX

<channel> : Volt_DIV?

RESPONSE FORMAT

<channel> : Volt_DIV <v_gain>

AVAILABILITY

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

EXAMPLE

The following sets the vertical sensitivity of channel 1 to 50 mV/div:

```
CMD$="C1:VDIV 50MV": CALL IBWRT(SCOPE%,CMD$)
```

STATUS

***WAI**
Command

DESCRIPTION

The `*WAI` (WAIt to continue) command, required by the IEEE 488.2 standard, has no effect on the oscilloscope, as Waverunner only starts processing a command when the previous command has been entirely executed.

COMMAND SYNTAX

```
*WAI
```

RELATED COMMANDS

`*OPC`

ACQUISITION

WAIT
Command

DESCRIPTION

The WAIT command prevents your Waverunner oscilloscope from analyzing new commands until the oscilloscope has completed the current acquisition. The optional argument specifies the timeout (in seconds) after which the scope will stop waiting for new acquisitions. If <t> is not given, or if <t> = 0.0, the scope will wait indefinitely.

COMMAND SYNTAX

```
WAIT [<t>]
```

<t>:=timeout in seconds (0.0 to 1000.0, default is indefinite)

EXAMPLE (GPIB)

send: "TRMD SINGLE"
loop {send:"ARM;WAIT;C1:PAVA?MAX"
read response
process response
}

This example finds the maximum amplitudes of several signals acquired one after another. ARM starts a new data acquisition. The WAIT command ensures that the maximum is evaluated for the newly acquired waveform.

C1:PAVA?MAX instructs the oscilloscope to evaluate the maximum data value in the Channel 1 waveform.

RELATED COMMANDS

*TRG SLEEP

WAVEFORM TRANSFER

WAVEFORM, WF
Command/Query

DESCRIPTION

A WAVEFORM command transfers a waveform from the controller to the oscilloscope, whereas a WAVEFORM? query transfers a waveform from the oscilloscope to the controller.

WAVEFORM stores an external waveform back into the oscilloscope's internal memory. A waveform consists of several distinct entities:

the descriptor (DESC)

the user text (TEXT)

the time (TIME) descriptor

the data (DAT1) block, and, optionally

a second block of data (DAT2).

See Chapter 4 for further information on waveform structure.

NOTE: You can restore to the oscilloscope only complete waveforms queried with WAVEFORM? ALL.

The WAVEFORM? query instructs the oscilloscope to transmit a waveform to the controller. The entities can be queried independently. If the "ALL" parameter is specified, all four or five entities are transmitted in one block in the order enumerated above.

NOTE: The format of the waveform data depends on the current settings specified by the last WAVEFORM_SETUP, COMM_ORDER and COMM_FORMAT commands.

COMMAND SYNTAX

<memory> : WaveForm ALL <waveform_data_block>

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

<waveform_data_block> : = Arbitrary data block (see Chapter 5).

QUERY SYNTAX

<trace> : `WaveForm?` <block>
<trace> : = {`TA`, `TB`, `TC`,`TD`, `M1`, `M2`, `M3`, `M4`, `C1`, `C2`, `C3`, `C4`}

<block> : = {`DESC`, `TEXT`, `TIME`, `DAT1`, `DAT2`, `ALL`}

If you do not give a parameter, ALL will be assumed.

RESPONSE FORMAT

<trace> : `WaveForm` <block>,<waveform_data_block>

TIP: It may be convenient to disable the response header if the waveform is to be restored. See the COMM_HEADER command for further details.

AVAILABILITY

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

EXAMPLES (GPIB)

The following reads the block DAT1 from Memory 1 and saves it in the file "MEM1.DAT". The path header "M1:" is saved together with the data.

```
FILE$ = "MEM1.DAT"
CMD$ = "M1:WF? DAT1"
CALL IBWRT(SCOPE%,CMD$)
CALL IBRDF(SCOPE%,FILE$)
```

In the following example, the entire contents of Channel 1 are saved in the file "CHAN1.DAT". The path header "C1:" is skipped to ensure that the data can later be recalled into the oscilloscope.

```
FILE$="CHAN1.DAT":RD$=SPACE$(3)
CMD$="CHDR SHORT; C1:WF?"
CALL IBWRT(SCOPE%,CMD$)
CALL IBRD(SCOPE%,RD$) Skip first 3 characters "C1:"
CALL IBRDF(SCOPE%,FILE$) Save data in file "CHAN1.DAT"
```

The following illustrates how the waveform data saved in the preceding example can be recalled into Memory 1:

```
FILE$ = "CHAN1.DAT"
CMD$ ="M1:"
CALL IBEOT(SCOPE%,0)disable EOI
CALL IBWRT(SCOPE%,CMD$)
CALL IBEOT(SCOPE%,1)re-enable EOI
CALL IBWRTF(SCOPE%,FILE$)
```

The "M1:" command ensures that the active waveform is "M1". When the data file is sent to the oscilloscope, it first sees the header "WF" (the characters "C1:" having been skipped when reading the file) and assumes the default destination "M1".

RELATED COMMANDS INSPECT, COMM_FORMAT, COMM_ORDER, FUNCTION_STATE, TEMPLATE, WAVEFORM_SETUP, WAVEFORM_TEXT

WAVEFORM TRANSFER

WAVEFORM_SETUP, WFSU

Command/Query

DESCRIPTION

The WAVEFORM_SETUP command specifies the amount of data in a waveform to be transmitted to the controller. The command controls the settings of the parameters listed below.

NOTATION			
FP	first point	NP	number of points
SN	segment number	SP	sparsing

Sparsing (SP):

The sparsing parameter defines the interval between data points. For example:

`SP = 0` sends all data points

`SP = 1` sends all data points

`SP = 4` sends every 4th data point

Number of points (NP):

The number of points parameter indicates how many points should be transmitted. For example:

`NP = 0` sends all data points

`NP = 1` sends 1 data point

`NP = 50` sends a maximum of 50 data points

`NP = 1001` sends a maximum of 1001 data points

First point (FP):

The first point parameter specifies the address of the first data point to be sent. For waveforms acquired in sequence mode, this refers to the relative address in the given segment. For example:

`FP = 0` corresponds to the first data point

`FP = 1` corresponds to the second data point

`FP = 5000` corresponds to data point 5001

Segment number (SN): The segment number parameter indicates which segment should be sent if the waveform was acquired in sequence mode. This parameter is ignored for non-segmented waveforms. For example:

`SN = 0` all segments

`SN = 1` first segment

`SN = 23` segment 23

The `WAVEFORM_SETUP?` query returns the transfer parameters currently in use.

COMMAND SYNTAX

`WaveForm_SetUp`
`SP,`<sparsing>`,NP,`<number>`,FP,`<point>`,SN,`

NOTE: After power-on, all values are set to 0 (i.e. entire waveforms will be transmitted without sparsing).

Parameters are grouped in pairs. The first of the pair names the variable to be modified, while the second gives the new value to be assigned. Pairs can be given in any order and restricted to those variables to be changed.

QUERY SYNTAX

`WaveForm_SetUp?`

RESPONSE FORMAT

`WaveForm_SetUp`
`SP,`<sparsing>`,NP,`<number>`,FP,`<point>`,SN,`

EXAMPLE (GPIB) The following instructs every 3rd data point (SP=3) starting at address 200 to be transferred:

```
CMD$="WFSU SP,3,FP,200": CALL
IBWRT(SCOPE%,CMD$)
```

RELATED COMMANDS

`INSPECT, WAVEFORM, TEMPLATE`

WAVEFORM TRANSFER

WAVEFORM_TEXT, WFTX

Command/Query

DESCRIPTION

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.

The WAVEFORM_TEXT? query returns the text section of the specified trace.

COMMAND SYNTAX

<trace> : WaveForm_TeXt '<text>'

<trace> : = {TA, TB, TC, TD, M1, M2, M3, M4, C1, C2, C3, C4}

<text> : = An ASCII message (max. 160 characters long)

QUERY SYNTAX

<trace> : WaveForm_TeXt?

RESPONSE FORMAT

<trace> : WaveForm_TeXt "<text>"

AVAILABILITY

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

EXAMPLE (GPIB)

The following documents Trace A (TA):

```
MSG$= "'Averaged pressure signal. Experiment
carried out Jan.15, 98'"
CMD$= "TA:WFTX"+ MSG$: CALL IBWRT(SCOPE%,CMD$)
```

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 or 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}

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:

```
CMDS$="XYAS?": CALL IBWRT(SCOPE%,CMDS$)
```

Example of response message:

```
XYAS C1,C2
```

RELATED COMMANDS

TRACE

CURSOR

XY_CURSOR_ORIGIN, XYCO

Command/Query

DESCRIPTION

The XY_CURSOR_ORIGIN command sets the position of the origin for absolute cursor measurements on the XY display.

Absolute cursor values can be measured either with respect to the point (0,0) volts (OFF) or with respect to the center of the XY grid (ON).

The XY_CURSOR_ORIGIN? query returns the current assignment of the origin for absolute cursor measurements.

COMMAND SYNTAX

XY_Cursor_Origin <mode>

<mode> : = {ON, OFF}

QUERY SYNTAX

XY_Cursor_Origin?

RESPONSE FORMAT

XY_Cursor_Origin <mode>

EXAMPLE (GPIB)

The following sets the origin for absolute cursor measurements to the center of the XY grid.

```
CMDS$="XYCO ON": CALL IBWRT(SCOPE%,CMDS$)
```

RELATED COMMANDS

XY_CURSOR_VALUE

CURSOR

XY_CURSOR_SET, XYCS

Command/Query

DESCRIPTION

The XY_CURSOR_SET command allows you 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 or cursors.

The CURSOR_SET command is used to position the time cursors.

NOTATION	
XABS	vertical absolute on X axis
XREF	vertical reference on X axis
XDIF	vertical difference on X axis
YABS	vertical absolute on Y axis
YREF	vertical reference on Y axis
YDIF	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: The suffix DIV is optional.

Parameters are grouped in pairs. The first of the pair names the cursor to be modified, while the second indicates its new value. Pairs can be given in any order and restricted to those items to be changed.

QUERY SYNTAX

XY_Cursor_Set? [<cursor,…<cursor>]

<cursor> : = {XABS, XREF, XDIF, YABS, YREF, YDIF, ALL}

If <cursor> is not specified, ALL will be assumed.

RESPONSE FORMAT

XY_Cursor_Set
<cursor>,<position>[,<cursor>,<position>…,<cursor>,<position>]

EXAMPLE (GPIB)

The following 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	ΔY/ΔX
<cursor type>_PROD	ΔY*ΔX
<cursor type>_ANGLE	arc tan(ΔY/ΔX)
<cursor type>_RADIUS	sqrt(ΔX*ΔX + ΔY*ΔY)

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:

```
CMDS$="XYCV? HABS_RATIO,HREL_ANGLE,VABS_PROD:
CALL IBWRT(SCOPE%,CMDS$)
```

RELATED COMMANDS

`CURSOR_MEASURE, CURSOR_VALUE, XY_CURSOR_ORIGIN`

DISPLAY

XY_DISPLAY, XYDS

Command/Query

DESCRIPTION

The XY_DISPLAY command turns the XY display mode on or off. (When off, the scope will be in standard display mode.)

The XY_DISPLAY? query returns the current mode of the XY display.

COMMAND SYNTAX

XY_DiSplay <mode>

<mode> : = {ON, OFF}

QUERY SYNTAX

XY_DiSplay?

RESPONSE FORMAT

XY_DiSplay <mode>

EXAMPLE (GPIB)

The following turns on the XY display:

```
CMDS$="XYDS ON": CALL IBWRT(SCOPE%,CMDS$)
```

RELATED COMMANDS

GRID

DISPLAY

XY_RENDER, XYRD

Command/Query

DESCRIPTION

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.

COMMAND SYNTAX

XY_RENDER <state>

<state> : = {SHARP,SMOOTH}

QUERY SYNTAX

XY_RENDER?

RESPONSE FORMAT

XY_RENDER <state>

EXAMPLE (GPIB)

The following sets the rendering to sharp:

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

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 XY 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 XY persistence data map.

COMMAND SYNTAX

XY_SAturation <value>

<value> : = 0 to 100 PCT

NOTE: The suffix PCT is optional.

QUERY SYNTAX

XY_SAturation?

RESPONSE FORMAT

XY_SAturation <value>

EXAMPLE (GPIB)

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

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

RELATED COMMANDS

PERSIST_SAT