Programmer Manual

Tektronix

CSA 803C Communications Signal Analyzer & 11801C Digital Sampling Oscilloscope

070-9970-00

WARRANTY

Tektronix warrants that the products that it manufactures and sells will be free from defects in materials and workmanship for a period of one (1) year from the date of shipment. If a product proves defective during this warranty period, Tektronix, at its option, either will repair the defective product without charge for parts and labor, or will provide a replacement in exchange for the defective product.

In order to obtain service under this warranty, Customer must notify Tektronix of the defect before the expiration of the warranty period and make suitable arrangements for the performance of service. Customer shall be responsible for packaging and shipping the defective product to the service center designated by Tektronix, with shipping charges prepaid. Tektronix shall pay for the return of the product to Customer if the shipment is to a location within the country in which the Tektronix service center is located. Customer shall be responsible for paying all shipping charges, duties, taxes, and any other charges for products returned to any other locations.

This warranty shall not apply to any defect, failure or damage caused by improper use or improper or inadequate maintenance and care. Tektronix shall not be obligated to furnish service under this warranty a) to repair damage resulting from attempts by personnel other than Tektronix representatives to install, repair or service the product; b) to repair damage resulting from improper use or connection to incompatible equipment; c) to repair any damage or malfunction caused by the use of non-Tektronix supplies; or d) to service a product that has been modified or integrated with other products when the effect of such modification or integration increases the time or difficulty of servicing the product.
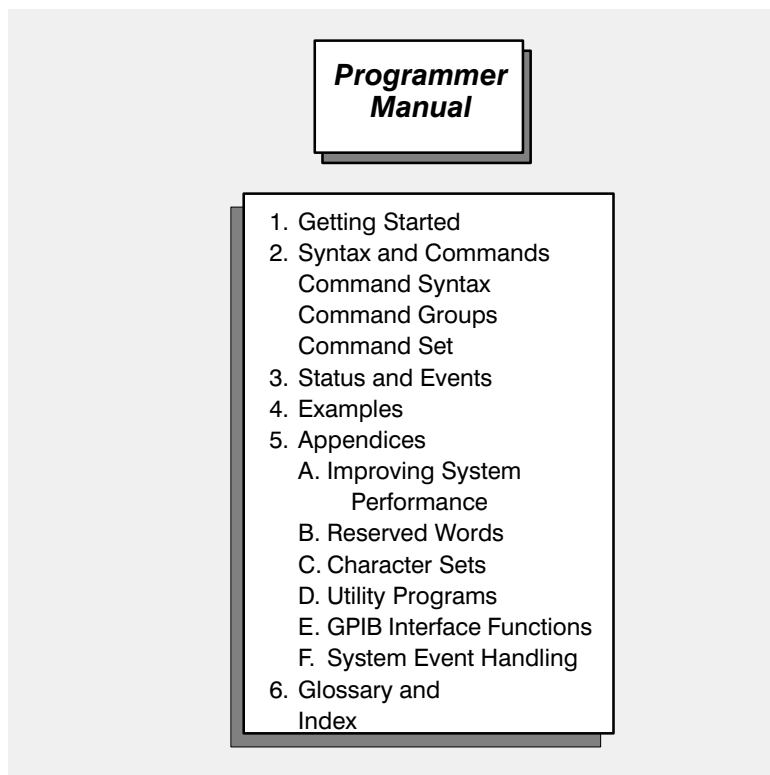
THIS WARRANTY IS GIVEN BY TEKTRONIX IN LIEU OF ANY OTHER WARRANTIES, EXPRESS OR IMPLIED. TEKTRONIX AND ITS VENDORS DISCLAIM ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. TEKTRONIX' RESPONSIBILITY TO REPAIR OR REPLACE DEFECTIVE PRODUCTS IS THE SOLE AND EXCLUSIVE REMEDY PROVIDED TO THE CUSTOMER FOR BREACH OF THIS WARRANTY. TEKTRONIX AND ITS VENDORS WILL NOT BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IRRESPECTIVE OF WHETHER TEKTRONIX OR THE VENDOR HAS ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.

About This Manual

This document provides comprehensive, easily accessible information to aid you in operating your instrument via the General Purpose Interface Bus (GPIB) or the RS-232-C interface.

How the Programmer Manual is Organized

The following illustration shows how this Programmer Manual is organized; a list of related documents is on the next page.

Programmer Manual

1. Getting Started
2. Syntax and Commands
   Command Syntax
   Command Groups
   Command Set
3. Status and Events
4. Examples
5. Appendices
   A. Improving System
        Performance
   B. Reserved Words
   C. Character Sets
   D. Utility Programs
   E. GPIB Interface Functions
   F. System Event Handling
6. Glossary and
   Index

CSA 803C & 11801C Programmer Manual

Related Manuals

You may want to refer to the other manuals that complete the documentation set for your CSA 803C or 11801C:

- The *CSA 803C User Manual* (Tektronix part number 070−9973−XX) or the *11801C User Manual* (Tektronix part number 070−9971−XX) covers all aspects of front-panel operation. Use this manual to quickly gain information about a specific topic, to get an overview of the menu system, or to get step-by-step instructions that demonstrate basic operation of the instrument.

- The *CSA 803C Service Manual* (Tektronix part number 070−9974−XX) provides information to maintain and service the CSA 803C, and provides a complete board-level description of CSA 803C operation. The *11801C Service Manual* (Tektronix part number 070−9972−XX) provides the equivalent information for the 11801C.

- The *User* and *Service Manuals* for the sampling heads you are using with your CSA 803C or 11801C.

Contacting Tektronix

Product Support	For application-oriented questions about a Tektronix measurement product, call toll free in North America: 1-800-TEK-WIDE (1-800-835-9433 ext. 2400) 6:00 a.m. − 5:00 p.m. Pacific time
	Or, contact us by e-mail: tm_app_supp@tek.com
	For product support outside of North America, contact your local Tektronix distributor or sales office.
Service Support	Contact your local Tektronix distributor or sales office. Or, visit our web site for a listing of worldwide service locations.
	http://www.tek.com
For other information	In North America: 1-800-TEK-WIDE (1-800-835-9433) An operator will direct your call.
To write us	Tektronix, Inc. P.O. Box 1000 Wilsonville, OR 97070-1000

Table of Contents

Status and Events

Examples

Appendices

Glossary and Index

List of Figures

List of Figures

List of Tables

List of Tables

Getting Started

You can write computer programs that remotely set the instrument's front panel controls or that take measurements and read those measurements for further analysis or storage.

This section covers the following:

- *This Manual* describes the major sections in this manual.

- *Setting Up Remote Communications* describes remote control. This includes connecting the instrument and setting the appropriate front panel controls.

This Manual

This manual includes the following sections.

Getting Started

The Getting Started section (Section 1) describes the structure and content of this manual and how to set up remote communications to your instrument either over the GPIB or RS-232-C interface.

Command Syntax

The *Command Syntax* section (Section 2) describes the structure and content of the messages your program sends to the instrument. Figure 1-1 shows the command parts as described in the *Command Syntax* section.

Command Parts   Header   Colon

```
LABel          DELete:TRA2
```

Mnemonic Space Link Argument
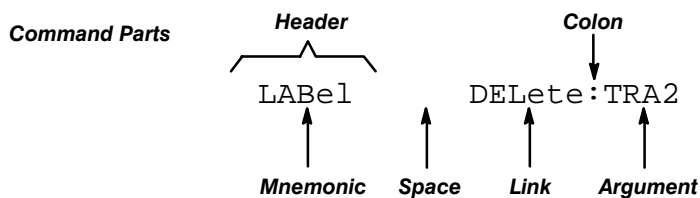
Figure 1-1: Common Message Elements

Commands

The *Commands* section (Section 3) describes each command's effect. The *Command Groups* subsection provides a list by functional area. The *Command Set* subsection arranges commands alphabetically (Figure 1-2).

Commands Grouped in 15 Functional Areas   *and*   **Commands Listed Alphabetically**
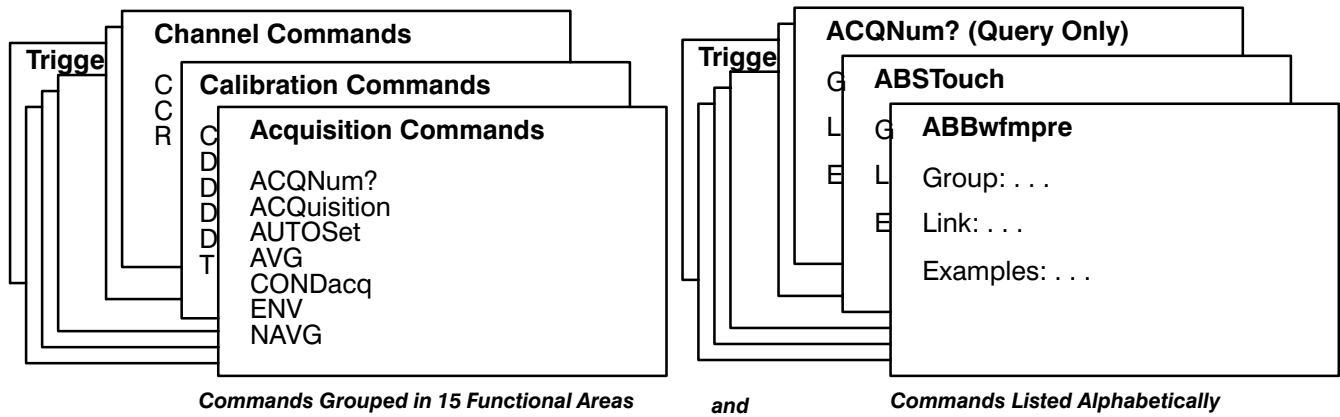
Figure 1-2: Functional Groupings and an Alphabetical List of Commands

Status and Events

The program requests information from the instrument. The instrument provides information in the form of status and error messages.

The *Status and Events* section (Section 4) describes how to use service requests (SRQ's) and various event messages in your programs.

Programming Examples

The *Programming Examples* section (Section 5) describes some example programs and how to compile them.

Setting Up Remote Communications

This section describes the implementation of each interface on the CSA 803C and 11801C, shows how to connect your instrument to other instruments that have either a GPIB or an RS-232-C interface, and explains how to set up the instrument's front panel for remote operation.

Connecting the Instrument to a GPIB Network

Before connecting devices to the GPIB, you should be aware of some rules concerning GPIB networks, cables, and connectors.

GPIB Interface Requirements — GPIB networks can be connected in any configuration, subject to the following rules:

- No more than 15 devices (including the controller) can be included on a single bus.

- In order to maintain bus electrical characteristics, one device load must be connected every two meters (six feet) of cable length. Generally, each instrument represents one device load on the bus.

- The total cumulative cable length must not exceed 20 meters.

- At least two-thirds of the device loads must be powered on when the network is in operation.

- There must be only one cable path from each device to each other device on the network; loop configurations are not allowed.

Cables — an IEEE STD 488 GPIB cable (available from Tektronix) is required to connect two GPIB devices.

GPIB Connector — a 24-pin GPIB connector is located on the rear panel of the instrument. See Figure 1-3. The connector has a D-type shell and conforms to IEEE STD 488. GPIB connectors can be stacked on each other. See Figure 1-4.
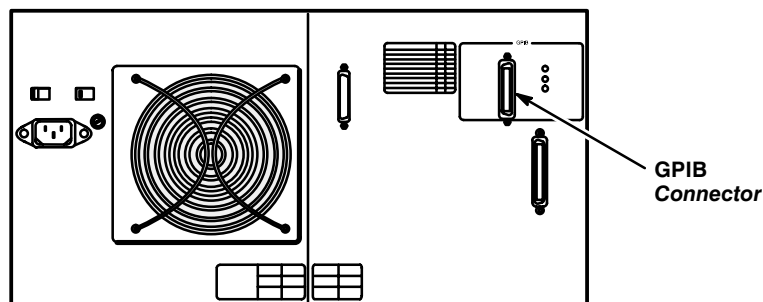
GPIB *Connector*

Figure 1-3: Location of GPIB Connector on Rear Panel
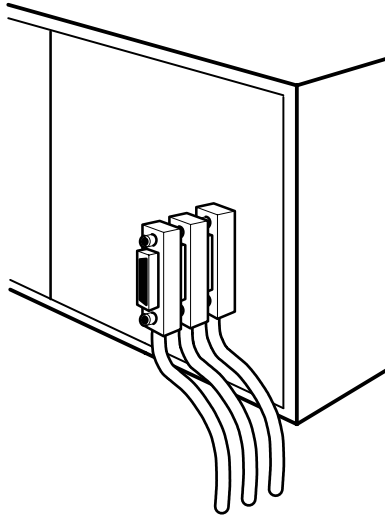
Figure 1-4: How GPIB Connectors can be Stacked Together

Setting Up GPIB Parameters — the following steps tell how to set up the GPIB parameters at the front panel.

☐ **Step 1:** Press the **UTILITY** major menu button to the right of the display area. The Utility major menu appears in the major menu area at the bottom of the display.

☐ **Step 2:** Touch the **GPIB**/**RS232C** selector in the major menu area. The **GPIB**/**RS232C Parameters** pop-up menu appears in the display area, with the following GPIB selectors:

   ■ **Mode** sets the instrument to be a talker/listener, to be a talker-only, or to be off the bus. Set the instrument to be a talker-only when the only device to be connected to it is a listen-only device, such as a printer or plotter. Otherwise, set it to be a talker/listener.

NOTE

The settings for the address and terminator parameters must match those of your controller. See the operating manual for your controller to select the appropriate parameters for its GPIB interface.

   ■ **Address** sets a unique primary communication address for the instrument. The address range is 0 to 30.

   ■ **Terminator** sets the method of indicating the end of device-dependent messages sent between the controller and the instrument. The choices are EOI (assert EOI line with transmission of last byte of message) or EOI/LF (send line feed character and assert EOI line with its transmission).

■ **Debug** specifies whether or not GPIB device-dependent messages (instrument commands) appear at the top of the instrument display.

NOTE

*When **Debug** is on, input/output processing is slowed.*

☐ **Step 3:** Repeatedly touch the selector for each parameter (except **Address**) until the value you want appears.

☐ **Step 4:** Touch the **Address** selector to assign the knobs to address selection. Rotate either knob to change the address.

```
                    GPIB/RS232C Parameters
   GPIB Parameters              RS232C Parameters

     Mode       Address      Baud       Echo       Stop
                             Rate                   Bits
  TalkListen      1        19200Bd      Off          1

  Terminator     GPIB       Parity    Flagging     Delay
                 Debug
     EOI          Off        None       Soft         0

                           EOL String Verbose     RS232C
       Tek C & F                                   Debug
         V81.1             CR/LF        Off         Off

                              Exit


  GPIB/      Identify    Color    Hardcopy    Main Size
  RS232C                                      100ps/div
                                    8 Pin     Main Pos
                                    HiRes      45.6ns
  InitializeInstrument  Labeling    Page     Remove/Clr
            Options                  to        Trace
            16:00:09     Off      Enhanced
            12-FEB-93             Accuracy
```

Figure 1-5: Typical GPIB Settings on the GPIB/RS232C Parameters Pop-Up Menu

After these parameters are set, the GPIB interface is ready to operate.

For more information, refer to the explanation of the **GPIB/RS232C Parameters** pop-up menu in the *User Manual* for your instrument.

Connecting the Instrument to an RS-232-C Device

The RS-232-C interface provides a point-to-point connection between two items of equipment, such as a computer or terminal and the instrument. The remainder of this section tells how to connect and set up the instrument for communication over the RS-232-C interface.

RS-232-C Interface Requirements — the RS-232-C standard defines two types of devices: Data Terminal Equipment (DTE) and Data Communications Equipment (DCE).

The instrument is configured as a DCE device. A 25-pin female D-type-shell RS-232-C connector is located on its rear panel. In industry-standard usage, a 25-pin male D-connector appears on DTE devices and a 25-pin female D-connector appears on DCE devices. A straight-through male-to-female cable (at least 9-wire) of less than 50 feet is typically used for local DTE-to-DCE connection.

Note, however, that some DTE devices may have female connectors. Also, the RS-232-C ports of many personal computers are configured as DCE devices, with either a 25-pin or a 9-pin connector. Refer to the documentation for your computer or terminal to determine if it is a DTE or a DCE device.

Table 1-1 shows how the pins map when connecting the instrument to another device in any of three common configurations:

- instrument to a 25-pin DTE device (most terminals)

- instrument to a 25-pin DCE device (for example, an IBM PC or compatible with a 25-pin COM port)

- instrument to a 9-pin DCE device (for example, an IBM PC or compatible with a 9-pin COM port)

In most cases, this pin mapping information will allow you to connect the devices in these configurations.

Table 1-1: RS-232-C Pin Mappings

CSA 803C/11801C	25-Pin DTE	25-Pin DCE	9-Pin DCE
1	1	1	NC
2	2	3	3
3	3	2	2
4	4	8	7
5	5	20	8
6	6	6	6
7	7	7	5
8	8	4	1
20	20	5	4

For more complicated cases (such as when working with non-standard devices or cables), the pin-out information in Table 1-2 should allow you to wire an appropriate connector. The following suggestions may help:

- Pay special attention to the input signal requirements of the external device (many devices require a constant high signal at one or more input pins)

- For DCE-to-DCE connections, do not connect an output line of one DCE to an output line of the other. *Disregarding this restriction can damage one or both devices*

- Ensure that the signal ground of the instrument is connected to the signal ground of the external device

- Ensure that the chassis ground of the instrument is connected to the chassis ground of the external device

Table 1-2: CSA 803C and 11801C RS-232-C Pin-out

Pin Number	Function	Mnemonic	Direction[1]
1	Chassis Ground	−none−	
2	Transmit Data	TxD	Input
3	Received Data	RxD	Output
4	Request to Send	RTS	Input
5	Clear to Send	CTS	Output
6	Data Set Ready	DSR	Output
7	Signal Ground	−none−	
8	Data Carrier Detect	DCE	Output
20	Data Terminal Ready	DTR	Input

[1] **Direction is from the perspective of the controller or terminal.**

Setting Up RS-232-C Parameters — you can set the parameters of the RS-232-C interface from the front panel (using the Utility major menu and the steps described here), or from within a program (using the `RS232` command). After these parameters are set, the RS-232-C interface is ready to operate.

Use the following steps to set up the RS-232-C parameters at the instrument front panel for remote operation.

☐ **Step 1:** Press the **UTILITY** major menu button to the right of the display area. The Utility major menu appears in the major menu area toward the bottom of the display.

☐ **Step 2:** Touch the **GPIB/RS232C** selector in the major menu area. The **GPIB/RS232C Parameters** pop-up menu is now displayed.

```
                    GPIB/RS232C Parameters

    GPIB Parameters              RS232C Parameters

     Mode      Address        Baud       Echo        Stop
                              Rate                    Bits
   TalkListen     1          19200Bd      Off          1

   Terminator    GPIB        Parity    Flagging     Delay
                 Debug
      EOI         Off         None       Soft          0

                            EOL String  Verbose     RS232C
        Tek C & F                                    Debug
          V81.1               CR/LF       Off         Off

                               Exit


    GPIB /    Identify    Color    Hardcopy    Main Size
    RS232C                                     100ps/div
                                    8 Pin      Main Pos
                                    HiRes        45.6ns
   InitializeInstrument  Labeling    Page     Remove/Clr
             Options                  to        Trace
             16:00:09      Off      Enhanced
             12-FEB-93              Accuracy
```

Figure 1-6: The GPIB/RS232C Parameters Pop-Up Menu

☐ **Step 3:** Repeatedly touch the selector for each parameter except **Baud Rate** and **Delay** until the value you want appears. Touching **Baud Rate** or **Delay** activates the knobs to control these parameters. The RS-232-C selectors are:

NOTE

The baud rate, stop bits, and parity settings must match those of the controller or terminal, or RS-232-C data communication will be impossible. Also, the controller or terminal's RS-232-C port must be set to use 8-bit characters.

■ **Baud Rate** sets the data transmission rate. The selections are 110, 150, 300, 600, 1200, 2400, 4800, 9600, or 19200 baud.

■ **Stop Bits** sets the number of stop bits sent after each character. The selections are 1, 1.5, or 2 bits.

■ **Parity** sets the error check bit for each character. The selections are none, even, or odd. When the parity setting is odd or even, the instrument generates the selected parity on output and checks incoming data against the selected parity on input. When the parity setting is none, no input parity error checking is performed, and no output parity is generated.

■ **Echo** allows characters sent to the instrument to be echoed. When echo is turned on, all characters sent to the RS-232-C port are echoed; when echo is turned off, input characters are not echoed.

Turn echo off when a computer program is transmitting data to the instrument (for example, when a BASIC program on a small computer is being used to control the instrument via the RS-232-C port). The computer program will not expect to see its commands echoed back, and the program will fail. The first command your program sends the instrument should be `ECHO OFF;VERBOSE OFF;INIT`.

Turn echo on when using a CRT or hardcopy terminal, or a computer with a terminal emulation program. Turning echo on in this case allows you to see what you have just typed on your computer or terminal screen.

■ **Flagging** sets the method of controlling the flow of data between devices. Flagging is a way for the device receiving data to tell the transmitting device when to stop or resume sending data. The selections are none, hard, or soft. When flagging is set to none, the instrument does not use or recognize any flagging.

When flagging is set to hard, the instrument uses the DTR (Data Terminal Ready) and CTS (Clear To Send) lines to control data transmission. On output, the instrument transmits data only when the DTR line is asserted. When the DTR line is not asserted, the instrument stops transmitting data. On input, the instrument unasserts the CTS line to stop transmission when its input buffer is three-quarters full, and asserts the CTS line to restart transmission when its input buffer is three-quarters empty.

When flagging is set to soft, on output the instrument stops transmitting data when it receives an XOFF (DC3) character, and begins transmitting again when it receives an XON (DC1) character. On input, the instrument sends an XOFF character to halt transmission when its input buffer is three-quarters full, and sends an XON character to resume transmission when its input buffer is three-quarters empty.

NOTE

Use hard flagging for binary data transfer. Soft flagging is usually not used with binary data transfer, since the data may contain XON and XOFF character equivalents.

■ **Delay** sets the minimum delay time for the instrument to respond to a query. The delay range is 0 to 60 seconds, in multiples of 20 ms.

■ **Verbose** displays status and event messages on the front panel as commands are executed. When verbose is turned on, each command sent to the instrument returns a response; for example, successfully executed commands return a response of OK, successfully executed queries return their query data, and events return a response of EVENT XXX, where XXX is an event code. When verbose is turned off, the controller must query the instrument to receive the message.

Turn verbose off when a computer program is transmitting data to the instrument (for example, when a BASIC program on a small computer is controlling the instrument with the RS-232-C interface). The first command your program sends the instrument should be ECHO OFF;VERBOSE OFF;INIT.

Turn verbose on when using a CRT or hardcopy terminal, or a computer with a terminal emulation program. Turning verbose on in this case gives you feedback on the execution of commands you have typed.

■ **EOL String** sets the end-of-line message terminator for the response to a query. The selections are CR (carriage return), LF (line feed), CR/LF (carriage-return-followed-by-line-feed), or LF/CR (line-feed-followed-by-carriage-return).

■ **Debug** controls whether or not RS-232-C commands appear at the top of the instrument display as they are executed.

NOTE

When debug is on, input/output processing is slowed.

Command Syntax

You can control the instrument through the GPIB and RS-232-C interfaces using commands and queries. This section describes the syntax these commands and queries use. It also describes the conventions the instrument uses to process them. The section entitled *Command Set* lists the commands and queries themselves.

A large set of commands can be used to control the operations and functions of the instrument from the external interfaces (GPIB and RS-232-C). The same command syntax is used for both interfaces.

The GPIB and RS-232-C command messages conform to the Tektronix Codes, Formats, Conventions, and Features Standard, or "Tek Codes and Formats" for short. It defines the format of program elements and statements for the command language.

NOTE

Contact your Tektronix field representative for information on the Tek Codes and Formats Standard.

You transmit commands to the instrument using an enhanced American Standard Code for Information Interchange (ASCII) character encoding. The CSA 803C and 11801C support both the standard ASCII character set and an additional "escape" character set that includes graphic elements. The character sets are described in Appendix C.

This manual describes commands and queries using the Backus-Naur Form (BNF) notation given in Table 2-1.

Table 2-1: BNF Symbols

Symbol	Meaning
< >	Defined element (e.g., *<arg>*)
::=	Is defined as (e.g., *<arg>* ::= argument)
|	Exclusive OR (e.g., PLUS | MINUS)
{ }	One of group is required (e.g., { ON | OFF }
[ ]	Optional item (e.g., [*<link>*:]*<arg>*)
...	Previous element(s) may be repeated

Command Structure

Command language messages are composed of set commands (also referred to as simply *commands*) and query commands (also referred to as *queries*). Set commands tell the instrument to take a specific action. Queries ask the instrument to return information about its state.

Commands are composed of four syntactic elements:

<header>	::=	The command name; if it ends with a question mark, the command is a query.
<delimiter>	::=	A space, colon (:), comma (,), or semi-colon (;) which breaks the message into segments for the instrument to process.
<link>	::=	A command sub-function. Not all commands have links.
<argument>	::=	A quantity, quality, restriction, or limit associated with the header or link.

Figure 2-1 shows the four syntactic elements.
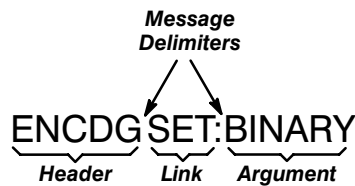
Figure 2-1: Example of Syntax Elements

Data Element Definitions

The data element types are: numeric, global, and quoted strings. Each is defined in Tables 2-2 through 2-4.

Table 2-2: Numeric Data Types

Element	Meaning
<ui>	Unsigned integer, range is 1 through 65,535; no leading space permitted (e.g., 9999).
<NR1>	Signed integer value (e.g., -5).
<NR2>	Floating point value, without an exponent (e.g., 3.7).
<NR3>	Floating point value, with an exponent (e.g., 2.2E-3).
<NRx>	$\{$ *<NR1>* $\mid$ *<NR2>* $\mid$ *<NR3>* $\}$. Range is: $-1\text{E} \pm 300$, 0, 1E ± 300, to 15 significant digits.

Table 2-3: Global Data Types

Element	Meaning
<asc curve>	ASCII-formatted trace data for one or more data points, in the form: *<NR1>* [$\{$,*<NR1>* $\}$ ...]
<bblock>	Binary block formatted trace or setting data, in the form: %*<byte count><data>* [$\{$ *<data>* $\}$ ...] *<checksum>* (Refer to the CURVe and SET? commands for a full explanation.)

Table 2-4: Quoted String Data Type

Element	Meaning
<qstring>	Quoted string data. This element can be any character(s) defined in the ASCII or expanded character sets, enclosed by apostrophes or quotation marks, and following these rules:

- You must use the same delimiter type to open and close the string; you cannot open with an apostrophe and close with a quotation mark or vice versa.

- You can use an apostrophe or quotation mark within the string if you follow the above rule and you enter the enclosing delimiter twice (i.e., ″double ″″ quote″).

- You can use a maximum string length of 127 characters, unless otherwise noted.

- You cannot use strings that include an embedded ASCII NULL character (0). However, carriage returns and line feeds can be included as text in a string.

Set and Query Commands

You can use most commands to set or query. Set commands modify instrument functions; queries return the current value(s) of functions. Most commands can be both set and queried. However, some commands and/or links can only be used to set, and other commands and/or links can only be used to query. Attempting to make a set-only command or link into a query always results in a syntax error.

Query-only commands contain a question mark appended to the header. The words **Query Only** appear in bold at the beginning of the text for these commands in the *Command Set* subsection of Section 3.

Query-only links contain a question mark preceding the link in the link description and are indicated in the *Command Set* description with the words **Query Only** in bold at the beginning of the text. As with any query, append the question mark to the header when querying the link.

A few commands and links can only be set. These commands are indicated in the *Command Set* description with the words **Set Only** in bold at the beginning of the description.

Set Commands

Set commands cause the instrument to perform a function or change a setting or mode. The four basic types of set commands are:

- Set commands with no arguments

- Set commands with arguments that have no links

- Set commands with link arguments

- Set commands with a mix of link and non-link arguments

Set Commands with No Arguments — are formed with a header only.

The INIt (Initialize) command is an example of a set command that has no arguments. It is also a set-only command.

    INIT

Set Commands with Arguments That Have No Links — are formed by specifying a header, following it with a delimiter (space or comma), and then adding its argument.

The AVG command is an example of a set command that has only an argument, in this case, ON or OFF:

    AVG ON

NOTE

The rules governing quoted strings begin on page 2-13.

Some commands take quoted string arguments (indicated by the notation *<qstring>* in the *Command Set*). This means that the argument must be enclosed in single or double quotes. The DATE command is an example:

```
DATE '14-FEB-93'
```

Other commands take numeric arguments. Numeric arguments are not enclosed in quotes, but simply follow the header and delimiter, which is usually a space. The NAVg command is an example:

```
NAVG 64
```

Set Commands with Link Arguments — are formed by specifying a header, following it with a delimiter, and then adding a link, a colon and an argument. Multiple sets of link arguments can be specified by placing a comma between each link argument. Links can follow a command header in any order.

The ENCdg command is an example of a set command that takes link arguments. ENCdg has two links, WAVfrm and SET:

```
ENCDG WAVFRM:BINARY
```

```
ENCDG SET:ASCII
```

```
ENCDG WAVFRM:BINARY,SET:ASCII
```

Set Commands with a Mix of Link and Non-link Arguments — are formed by specifying a header, following it with a delimiter, and then adding comma-delimited link arguments and/or non-link arguments (usually in any order).

For example, the following COPy command sets the PRInter link to HPGl and specifies the STARt argument for the COPy header:

```
COPY PRINTER:HPGl,START
```

Queries

Queries cause the instrument to return a measurement, trace data, or a status condition (for example, a current setting or mode). The instrument puts the response message in its output buffer.

Query commands have three basic structures:

■ Set command headers that are made into queries.

These are formed by placing a question mark (?) after the header and omitting the set command argument.

NOTE

*In query examples, the query command is shown in **bold** type and the instrument response is in regular type.*

AVG?
AVG ON

■ Set command headers with links that are made into queries.

These are formed by placing a question mark (?) after the header, omitting the colon and argument for each link, and separating each link being queried by a comma. Query links may follow a query header in any order.

ENCDG? WAVFRM
ENCDG WAVFRM:BINARY

When query links are omitted, all links and their current arguments are returned.

ENCDG?
ENCDG SET:ASCII,WAVFRM:ASCII

■ Queries that have no set-command equivalent (these are called "query-only.")

An example of a query-only command is NVRam?, which returns the number of bytes of nonvolatile RAM available for storing front panel settings:

NVRAM?
NVRAM 104376

Multiple Query Forms

Most set/query commands have only two basic forms of query: specific (`<header> ? <link> [ { , <link> } ... ] `) and general (`<header> ?`). However, some commands have additional query forms because of their ability to specify a particular trace, channel, or color. These commands are:

```
ADJTRACE [ <ui> ] ?

CH [ <alpha><ui> ] ?

COLOR<ui>?

TRACE [ <ui> ] ?
```

NOTE

`<ui>` is an unsigned integer value which in these examples represents a trace number (1 to 8) or a channel (1 to 4).

The multiple query forms of these commands return information ranging from most specific to most general.

For example, the `CH [ <alpha> [ <ui> ] ]` command can produce queries in four basic forms:

- **`CH<alpha><ui> ? <link> [ { , <link> } ... ]`**

  This form returns the specified links and arguments for the specified channel (`<ui>`) only.

- **`CH<alpha><ui> ?`**

  This form returns all links and arguments for the specified channel.

- **`CH?`**

  This form returns all links and arguments for all channels.

The `ADJtrace<ui>`, `COLor<ui>`, and `TRAce<ui>` commands are similar, in that you can specify the trace number (`<ui>`) or query all traces by omitting `<ui>`.

Using Query Responses as Commands

Any response to a query that has a corresponding set command can always be returned to the instrument as a valid command. This makes it easy to save a response from the instrument in a string variable, then send the contents of the string variable back to the instrument as a command when you want to return the instrument to its previous state.

For example, the response to the following query could be used as a set command:

```
ENCDG? WAVFRM,SET
ENCDG WAVFRM:BINARY,SET:ASCII
```

The response to a query that has no corresponding set command (a query-only command) may not be returned to the instrument as a set command string. Any such attempt always results in a syntax error.

For example, if you sent the next example query to the instrument and then tried to send the response (NVRAM 104376) as a set command, you would get a syntax error.

```
NVRAM?
NVRAM 104376
```

However, a query response that includes a mixture of set, set-only, and query-only links can be returned to the instrument as a set command without generating a syntax error. In such cases, the instrument simply ignores the attempted modification of the query-only link(s).

For example, the response from the CHM1? query (or from any form of the CH command query where you don't specify a link), includes a response from the UNITS query-only link. However, the instrument lets you return the query response from such a command as a set command without causing a syntax error.

Concatenating Commands

Any combination of set and/or query commands may be joined with a semicolon. Thus:

```
RQS ON
```

```
ENCDG?
```

```
UPTIME?
```

may be combined as:

```
RQS ON;ENCDG?;UPTIME?
```

The response to a command message containing more than one query consists of two or more messages, separated by semicolons.

Thus, for the query command:

```
RQS?;EVENT?
```

the query response might be:

```
RQS ON;EVENT 0,"RQS is ON,...SRQ pending"
```

Defining New Command Strings

The instrument provides a command (DEFine) that enables you to create new command names. That is, you can rename an existing command function, or you can concatenate several existing commands under a single, new command name.

For example, to create a command that gives you the date and time, you could give the following command:

```
DEFINE "DATIME?","DATE?;TIME?"
```

Also, there is an UNDEF command to remove any or all new commands that you have created. So, to remove the previous example, give this command:

```
UNDEF "DATIME?"
```

The DEF? query command will return a list of all defined strings and their associated command definitions. To query for a list of defined command strings, give this command:

```
DEF?

DEFINE "DATIME?","DATE?;TIME?";DEFINE "E","rs232
echo:on";DEFINE "V","rs232 verbose:on"
```

Measurement (*<meas>*) Commands

The symbol *<meas>* represents one or more of the instrument measurements. For example, *<meas>*? represents a measurement query, such as RMS?. Each *<meas>*? measurement has its own entry in the command set. Refer to the *<meas>*? entry for a list of all the measurements.

The STAT? query returns measurement statistics (mean and standard deviation) for the measurement selected with the STATIstics MODE command. See the *<meas>*? entry for a list of measurements.

Waveforms and Traces

The terms waveform and trace both pertain to signals acquired or stored by the instrument; but are not interchangeable in command syntax. Some headers begin with WFM*xxx* (e.g., WFMPre, WFMScaling); other headers, links, or arguments use TRAce*<ui>* (e.g., TRACE3) form.

Command Processing Conventions

Command processing conventions are rules that specify how the instrument interprets commands you send to it, or how it handles input/output to or from a GPIB or an RS-232-C interface port. Command processing conventions relate to such things as: abbreviating commands being sent to the instrument, getting "long-form" or "short-form" responses from the instrument, using upper or lower case characters in commands, using quoted command strings, and special considerations for each type of interface.

Abbreviating Commands

Each command reserved word (header, link, or argument) that is transmitted to the instrument has an abbreviation. Abbreviations are used in examples in the *Command Set* section; the abbreviated spelling is shown with uppercase letters in the header, link, and argument syntax blocks. The complete list of reserved words and their abbreviations is in Appendix B, *Reserved Words*.

For example, the reserved word:

    TBMain

may be abbreviated to:

    TBM

Getting Long-form or Short-form Responses

The `LONgform` command determines whether the instrument responds to queries in long form or short form. In long form, queries return fully spelled reserved words, and an event query returns both the numeric event code and its associated message string. In short form, queries return abbreviations of reserved words and event queries return only the numeric event code.

NOTE

Long form is easier to read; short form is more efficient during data transfers.

The following two examples are with `LONgform` set to `ON`:

ENCDG? WAVFRM
ENCDG WAVFRM:ASCII

EVENT?
EVENT 250, "NO TRACES DEFINED"

NOTE

Note that the `LONgform` *command only affects instrument responses; set commands and queries can always be sent to the instrument in either long or short form. Examples in this book use abbreviated command spellings; responses are in long form.*

Here are the same examples with `LONgform` set to `OFF`:

ENCDG? WAVFRM
ENC WAV:ASC

EVENT?
EVENT 250

Using Upper and Lower Case

The CSA 803C and 11801C accept both upper and lower case alphabetic input data. Thus, the following two commands are recognized as identical:

RQS ON

rqs on

The instrument returns the same case of alphabetic data to the GPIB or RS-232-C ports as you enter from within a quoted string.

Using Quoted Strings

Some commands accept or return quoted string (`<qstring>`) data; for example, the `<qstring>` argument to the `DATE` command contains the day, month, and year. The rules for quoted string usage are described below.

■ The maximum length of any quoted string is 127 characters, excluding delimiters (unless noted otherwise in the *Command Set* section).

■ An apostrophe (′) or a quotation mark (″) is a valid string delimiter. However, quoted string data returned to the controller (query data) is delimited solely with quotation marks.

■ The same type of delimiter that opens a quoted string must close that string. Examples:

"this is a quoted string″ and ′so is this′

′But this is not a quoted string″

■ You can mix quotation marks and apostrophes within a string if the previous rule is followed. For example:

"this is an ′acceptable′ string″ and ′so ″is″ this′

■ A delimiter may be included within a string by simply repeating the delimiter. Examples:

"double ″″ quote″ and ′single ′′ quote′

■ For GPIB transfers, a quoted string may not be terminated with an EOI interface signal prior to the closing delimiter.

For example, a missing end-quote after the command `TESt` in the command below produces an invalid string.

"test*<EOI>*

■ A carriage return or line feed embedded within a quoted string does not terminate that string but is considered to be just another character in the string.

■ A quoted string may not include an embedded ASCII NULL character (0).

Terminating Messages

Message terminators are transmitted by a sending device to let receiving devices know that message transmission is complete. The CSA 803C and 11801C allow you to select a message terminator that is compatible with the controller or terminal you are using.

Terminators for the RS-232-C interface — are selected through the front panel using the **GPIB/RS232C** pop-up menu from the Utility major menu or through the interface using the `RS232` command. RS-232-C terminators are:

- CR (carriage return)

- LF (line feed)

- CR/LF (carriage return followed by line feed)

- LF/CR (line feed followed by carriage return)

Line feeds and carriage returns embedded within binary block (`<bblock>`) data are treated as data bytes, not as message terminators. Once the instrument begins reading a binary block, line feeds and carriage returns are not processed as terminators until the byte count of the block is satisfied.

Terminators for the GPIB interface — are selected through the front panel only, using the **GPIB/RS232C** pop-up menu from the Utility major menu. GPIB terminators are:

- EOI (assert EOI management line with the last byte of transmission)

- EOI/LF (transmit line feed as last character *and* assert EOI line)

Using White Space

All command elements (headers, links, arguments, or punctuation) may be preceded or followed by white space, defined as:

- For RS-232-C — as blanks and carriage returns or line feeds that precede a header

- For GPIB with the EOI terminator — as any combination of blanks, carriage returns, or line feeds

- For GPIB with the EOI and line feed terminator — as any combination of blanks or carriage returns (blank characters)

Thus, the following example:

```
ENCDG  WAVFRM  :  ASCII  ,  SET  :  BINARY
```

is equivalent to:

```
ENCDG WAVFRM:ASCII,SET:BINARY
```

Null Commands

Commands consisting solely of any combination of blank characters, carriage returns, and line feeds are called null commands. Null commands are ignored and do not produce an error.

I/O Buffers

The following information pertains to both GPIB and RS-232-C input/output buffers, except as noted otherwise.

I/O Buffer Sizes — are 256 bytes for the GPIB Input buffer, 1024 bytes for the RS-232-C Input buffer, and 1024 bytes for the GPIB and RS-232-C Output buffer.

Data that exceeds the sizes of the GPIB and RS-232-C input/output buffers (256/1024 bytes) can be accepted. The instrument parses input data as soon as it is received at either port, thereby continuously emptying the input buffers while processing commands.

If an external controller fills an input buffer before the instrument has an opportunity to process the contents, the instrument holds off the external controller (with GPIB interface signals or RS-232-C flagging) until the buffer has been processed, leaving room for more input data.

Likewise, if a query response fills an output buffer, the instrument stops sending data to the buffer until some of the data are read by the external controller or terminal.

When a New Message is Received at the GPIB Port — the instrument unconditionally clears its GPIB output buffer (no error is reported). This means that the GPIB output buffer of the instrument must be read by the controller after each message containing a query, or the response will be lost (overwritten).

When GPIB Input and Output Message Buffers are Full — the instrument unconditionally clears the GPIB output buffer. An execution error is also reported (event code 203, "I/O buffers full").

If the GPIB Buffers Are Empty — and the instrument is talk-addressed and not currently processing a GPIB command, it returns a Talked-With-Nothing-To-Say (TWNTS) message to the controller. This message is one byte with all eight bits set, ended by a message terminator (FF*<EOI>*). It is then up to the controller program to take appropriate action.

If the RS-232-C Output Buffer is Empty — and an external device attempts to read data from the RS-232-C port, the external device will "hang" the interface (no further input/output operations will be possible). This condition cannot occur when using a computer or terminal to send commands interactively to the instrument over the RS-232-C interface.

NOTE

> *If a "hang" condition occurs, consult your controller or terminal operator manual for restart instructions.*

This condition may occur when executing a program that expects input from the instrument's RS-232-C port. In such cases, it is up to the program to recognize a "timeout" condition for expected input and take appropriate action.

GPIB Specific Conventions

When the instrument receives a Device Clear (DCL) or Selected Device Clear (SDC) interface message from the GPIB, it does the following:

1. Clears any service requests and all pending events except power on.

2. Clears the GPIB input and output buffers.

3. Restarts GPIB message processing in the instrument.

DCL and SDC interface messages do not change instrument settings or stored data, and do not interrupt front panel control or non-programmable functions.

RS-232-C Specific Conventions

You should be aware of the processing conventions that are specific to the RS-232-C interface. These conventions pertain to:

- Transferring binary block data

- Echoing character input

- Using Verbose mode

- Processing "break" characters

- RS-232-C I/O errors

When Transferring Binary Block Data — using the RS-232-C port, note the following points:

- Do not transmit binary block data to the instrument when ECHO is set to ON. Attempting to do so causes the input block to be discarded and generates event code 164.

- Do not use binary data transfers with soft flagging unless you can ensure that the data does not contain XON or XOFF characters. Using DTR/CTS (hard) flagging guarantees correct data transfer.

■ All eight bits of a binary block data byte contain meaningful information. To ensure that all eight bits are received or transmitted, an RS-232-C device must be configured to receive and transmit eight-bit characters (set the RS-232-C word length to eight bits).

For example, an IBM-compatible controller uses this MS-DOS command to set up its RS-232-C interface:

```
MODE COM1:9600,n,8,1
```

Echoing Character Input — means that all characters received at the RS-232-C port are echoed back to the command source when ECHO is set to ON.

You can turn echo on or off from the front panel by selecting **GPIB/RS232C** from the Utility major menu and touching the **Echo** selector, or you can send the commands: `RS232 ECHO:ON` or `RS232 ECHO:OFF`.

When you are using a computer program to transmit commands to the instrument (for example, when a BASIC program is being used to control the instrument via the RS-232-C port), ECHO should be set to OFF.

When you are using a CRT or hardcopy terminal, or a computer running terminal emulation software to send commands interactively to the instrument via the RS-232-C port, ECHO should be set to ON.

When ECHO is set to ON, it has the following effects on command input:

■ The instrument solicits command input with a ">" prompt. When this prompt appears on an RS-232-C device, enter a valid command and terminator.

■ All command input is buffered. Therefore, commands will not be analyzed or executed until a terminator is received at the RS-232-C port. (As you may recall, RS-232-C I/O is normally not buffered, which means that each input character is processed as soon as it is received at the RS-232-C port.)

■ Until the command is terminated, it may be edited with any of the following special characters:

CONTROL-R retypes the current input command and places the cursor to the right of the last character of the command.

CONTROL-U deletes the current command and returns the cursor to the start of the line.

BACKSPACE erases the character to the left of the cursor (the effect of the backspace character is compatible with CRT terminals, but not with hardcopy terminals). If a character has been erased with the backspace key, the newly edited command can be seen by using the CONTROL-R character (applies to both CRT and hardcopy terminals).

DEL or RUBOUT same function as BACKSPACE.

BACKSLASH (\) use the backslash to place special editing characters (CR, LF, BACKSPACE, DEL, CONTROL-R, or CONTROL-U) in a quoted string. To place a backslash character in a quoted string, enter two consecutive backslashes ("\\" is interpreted as "\").

■   Command input is discarded if it exceeds 256 bytes (the input buffer size) before a terminator is entered. If this happens, a command error (event code 163) is posted to the RS-232-C port and the input buffer is emptied.

■   Non-printable ASCII characters are echoed with the visual representations shown in Table 2-5.

Table 2-5: Non-printable ASCII Character Representations

ASCII Character	Echoed Character	ASCII Character	Echoed Character
NUL (0)	^@	DC2 (18)	^R[1]
SOH (1)	^A	DC3 (19)	^S[2]
STC (2)	^B	. .	.
. .	.	. .	.
. .	.	NAK (21)	^U[1]
BS (8)	^H[1]	. .	.
HT (9)	^I	. .	.
LF (10)	^J[1]	SUB (26)	^Z
. .	.	ESC (27)	^[
. .	.	FS (28)	^\
CR (13)	^M[1]	GS (29)	^]
. .	.	RS (30)	^^
. .	.	US (31)	^_
DC1 (17)	^Q[2]	DEL (127)	^?[1]

[1] **Only echoed when preceded with a backslash.**
[2] **Only echoed when soft flagging is disabled.**

Using Verbose Mode — causes the instrument to return a response for each command sent. When VERBose is set to OFF, only valid queries return a response from the instrument.

You can turn verbose on or off from the front panel by selecting **GPIB**/**RS232C** from the Utility major menu and touching the **RS232C Parameters Verbose** selector, or you can send the commands RS232 VERBOSE:ON or RS232 VERBOSE:OFF.

When RS-232-C VERBOSE is set to ON, each semicolon or terminated input command causes the instrument to return one of these responses:

`OK`	Returned for a successfully executed set command
`<query response>`	Returned for a successfully executed query
`EVENT <NR1>[,<qstring>]`	Returned when the instrument detects an error while parsing or executing a query/set command, where the `<NR1>` value represents an event code, and the optional `<qstring>` is an event code description string that describes the numerical event code. The event code description string `<qstring>` is only returned when `LONgform` is set to `ON`.
	If more than one error is detected while parsing a query or set command, only one `EVENT` response is returned to the RS-232-C port. All other errors are stacked and may be polled with the `STBYTE?` or `EVENT?` commands.

Table 2-6 demonstrates typical response behavior with `VERBose` mode set to `ON`.

Table 2-6: Examples of Responses with VERBOSE ON

Input Command(s)	CSA 803C/11801C Response
`LONGFORM OFF`	`OK`
`INPUT STO1;RS232? BAUD`	`OK;RS232 BAUD:9600`
`JUNK;INIT;INPUT?`	`EVENT 156,OK;INPUT STO1`
`JUNK;INIT`	`EVENT 156;OK`

When RS-232-C `VERBose` is set to `OFF`, only valid queries cause the instrument to return responses to the RS-232-C. Valid set commands, invalid set commands, and invalid queries elicit no response event messages from the instrument. Errors associated with invalid commands are not discarded; they are stacked and may be polled at any time by using the `STBYTE?` or `EVENT?` commands.

Table 2-7 demonstrates typical response behavior with `VERBose` set to `OFF`.

Table 2-7: Examples of Responses with VERBOSE OFF

Input Command(s)	CSA 803C/11801C Response
`INPUT STO1;RS232?BAUD`	`RS232 BAUD:9600`
`JUNK;INIT;INPUT?`	`INPUT STO1`
`JUNK;INIT`	*(none)*

Verbose mode affects event communication at power-on. When the instrument is turned on and completes its power-on cycle, the instrument communicates events differently depending on the state of the verbose function.

NOTE

The factory default state for verbose mode is off.

■ When VERBose is set to ON at power-on, an asynchronous message is written to the RS-232-C port. This message reports either that the instrument is operating satisfactorily (Event 401, Power on), or that diagnostics have discovered a fault (Event 394, Test completed and failed).

■ When VERBose is set to OFF at power-on, no asynchronous messages are written to the RS-232-C port. Instead, power-on events are stacked in the usual manner.

When the instrument Senses a BREAK Signal — at the RS-232-C port, it returns a special message that acknowledges this transmission. The form of the acknowledgement message depends on whether ECHO is set to ON or OFF.

■ When ECHO is set to ON, the instrument signals that it has processed the BREAK signal by echoing a new prompt symbol (for example, >) for command input.

■ When ECHO is set to OFF, the instrument signals that it has processed the BREAK signal by sending the following character string to the RS-232-C device:

DCL<*terminator*>

Reception of the BREAK signal clears the RS-232-C input and output buffers and restarts the instrument's RS-232-C message processing. BREAK signals do not change instrument settings or stored data, and do not interrupt front panel operation or non-programmable functions.

RS-232-C I/O Errors — are reported when there is a problem with parity, framing, or input buffer overruns.

To report RS-232-C errors, the instrument prints an error message on the display and posts an event code to both the GPIB and the RS-232-C port. See Table 2-8.

Table 2-8: RS-232-C I/O Errors

I/O Error	Event Code	Information
Parity	653	Check to identify transmission errors (PARITY ON)
Framing	654	A stop bit was not detected when data was received at RS-232-C port (indicates baud-rate mismatch)
Input Buffer Overrun	655	Software or hardware input buffer overflowed with data (caused by improper or nonexistent flagging)

To recover from I/O errors, the CSA 803C/11801C RS-232-C interface takes the following actions:

NOTE

During these I/O error recovery steps (when ECHO is set to OFF), the instrument may process incomplete commands, causing spurious syntax or semantic errors to be reported.

■ When ECHO is set to OFF, all unparsed input buffer data are discarded until a semicolon or *<terminator>* character is encountered. Command processing resumes or resynchronizes from the point at which the semicolon or *<terminator>* is found.

■ When ECHO is set to ON, all buffered but unparsed input data are discarded and you are prompted again for input.

Command Groups

This section presents the command set in two ways. First, the commands are presented by functional group. Second, they are listed alphabetically. The functional groups are shown in Table 2-9. The alphabetical list provides more detail on each command.

Functional Groups

Table 2-9 lists the groups and their function(s). Table 2-10 on the following two pages show all the commands grouped by function.

Table 2-9: Functional Groups in the Command Set

Group	Functions Controlled
Acquisition	Acquisition (sampling) of traces
Calibration	Self-calibration functions
Channel	Sampling head vertical parameters
Cursor	Trace cursor selection and positioning
Data Transfer	Transfer of trace data and front panel settings to and from the instrument
Diagnostics	Self-tests and extended diagnostics
Display and Color	Display, Histogram, Mask testing, and color parameters
External I/O	Printer parameters, debug functions, and RS-232-C parameters
Label and Text	Placement of user-defined labels and text
Measurement	Measurement functions
Miscellaneous	System and front panel functions
Status and Event	Instrument event reporting, hardware identification, and configuration information
Time Base	Main and window record length and position
Trace and Settings	Creation and removal of traces and associated front panel settings
Triggering	Triggering parameters

Table 2-10: Command Groups

Acquisition	Calibration	Channel	Cursor
ACQNum?	CALibrate	CH	CURSor
ACQuisition	DAFiltering	CH?	DOT1Abs
AUTOSet	DAMeasref	RHOPos	DOT2Abs
AVG	DCOmp		DOT1Rel
CONDacq	DIV2		DOT2Rel
ENV	TBCalmode		H1Bar
FFT			H2Bar
FMOde			RHOZero
NAVg			V1Bar
NENv			V2Bar
NGRAded			
NHISt.pt			
NWAVfrm			

Data Transfer	Diagnostics	Display and Color	External I/O
ABBwfmpre	DIAg?	COLor	ALTinkjet
BYT.or	TESt	COLor DEFault	BITMap
CURVe		DISPlay	COPy
ENCdg		DSYS	DEBug
INPut		GRAticule	DSKJETC
OUTput		HISTOgram	FEOi
SET?		MASK	HP560C
VPCurve?		MASKDefine	HPGl
WAVfrm?		MASKStat	LSRJET
WFMPre		STATHist	PIN8
		TEXt	PIN24
		USEREye	RS232

Label and Text	Measurement		Miscellaneous/ System
LABAbs	AMPlitude?	PDElay?	ABSTouch
LABel	COMpare	PERiod?	BCOrrection
LABRel	CROss?	PHAse?	CHIdent
TEXt	DUTy?	PP?	DATE
	EXTInction?	REFSET	DEFine
	EYEPattern	RISetime?	DSYmenu?
	FALltime?	RMS?	FPAnel
	FREq?	SFRequency?	FPUpdate
	JITter?	SMAgnitude?	INIt
	MAX?	SNRatio?	LONgform
	MEAN?	STAT?	POWeron?
	<meas>?	STATIstics	SPEaker
	MEAS?	THD?	TIMe
	MID?	TOPBase	UNDEF
	MIN?	UNDershoot?	UPTime?
	MPAram	WIDth?	
	MSLIst	YTEnergy?	
	MSNum?	YTMns_area?	
	MSYs	YTPls_area?	
	NOIse?		
	OVErshoot?		

Table 2-10: Command Groups (Cont.)

Status and Event	Time Base / Horizontal	Trace and Settings	Triggering
EVENT?	MAINPos	ADJtrace	TRIgger
ID?	TBMain	CLEar	
NMAsk	TBWin	DELete	
RQS		DISPNum?	
SAMid?		FPSList?	
SRQMask		FPSNum?	
STByte?		HREfpt	
UID?		MAXTranum?	
		NVRam?	
		RECall	
		REMove	
		SELect	
		SETseq	
		STD? *(11801C)*	
		STOList?	
		STONum?	
		STORe	
		TRAce	
		TRAList?	
		TRANum?	
		WFMScaling	
		WIN	
		WINList?	
		WINNum?	

Acquisition Commands

The Acquisition commands control trace acquisition. The commands are presented in three groups: commands that control signal acquisition, such as ACQuisition in Table 2-11, commands that control trace functions, such as AVG in Table 2-12, and commands that affect acquisition parameters, such as NAVg in Table 2-13.

Table 2-11: Signal Acquisition Commands

Command	Meaning
ACQuisition	Starts and stops unconditional trace acquisition.
AUTOSet	Adjusts the trace signal for optimal display.
CONDacq	Controls the condition(s) on which the acquisition of traces stops.

Table 2-12: Trace Function Commands

Command	Meaning
AVG	Turns trace averaging on or off. (Averaging can also be defined in the trace description; use the TRAce command in the Trace and Settings group.)
AVG.ENV	Turns both trace averaging and enveloping on or off. (Can also be defined in the trace description; use the TRAce command in the Trace and Settings group.)
ENV	Turns trace enveloping on or off. (Enveloping can also be defined in the trace description using the TRAce command.)

Table 2-13: Acquisition Parameter Commands

Command	Meaning
ACQNum?	Returns the number of acquisition systems.
NAVg	Sets the number of acquisitions to be used in trace averaging.
NENv	Sets the number of acquisitions to be used in trace enveloping to halt conditional acquisition.
NGRAded	Sets the number of bins that must overflow on a color graded display to halt conditional acquisition.
NHISt.pt	Sets the number of data points that must be included in a histogram to halt conditional acquisition.
NMAsk	Sets the number of mask hits that must be acquired to stop conditional acquisition.
NWAVfrm	Sets the number of trace records that must be processed into histogram, color graded, or masks to halt conditional acquisition.

Calibration Commands

The Calibration commands initiate instrument self-calibration features and report on their condition. Calibration refers to the state of accuracy of the instrument. The Calibration and Status commands are shown in Table 2-14.

Table 2-14: System Calibration and Status Commands

Command	Meaning
CALibrate	Performs manual or automatic calibration of a specified sampling head channel.
DAFiltering	Delay Adjust Filtering sets the hardware filtering constant used by the delay adjustment command, CALibrate DADj.
DAMeasref	Sets the inter-head delay to mid-range for the specified sampling head channel.
DCOmp	Sets continuous strobe delay calibration or or off.
DIV2	Halves the internal calibrator frequency.
TBCalmode	Sets the time base calibration mode.

Channel/Vertical Commands

The Channel/Vertical commands set and query the vertical parameters of an input channel. The CH<*ui*> command has a large number of links, some of which are specific to a sampling head type. The Channel/Vertical commands are shown in Table 2-15.

Table 2-15: Channel/Vertical Commands

Command	Meaning
CH	Sets the vertical parameters for the specified channel.
CH?	Returns the vertical parameters for all channels.
RHOPos	Sets the rho calconstant on the specified trace.

Cursor Commands

Cursor commands control the creation and placement of cursors on traces. Cursors are positioned with the dot or bar cursor commands. Use the CURSOR command to display the cursors and control the readout in the Cursor menu. (The readouts show the difference (Δ) between the cursors.) The four cursor types are split dot, paired dot, horizontal bar, and vertical bar cursors. The Cursor commands are shown in Table 2-16.

Table 2-16: Cursor Commands

Command	Meaning
CURSor	Selects cursor operating characteristics.
DOT1Abs DOT2Abs	Positions the first and second split or paired cursors to specified absolute locations.
DOT1Rel DOT2Rel	Positions the first and second split or paired cursors relative to the DOT1Abs and DOT2Abs locations, respectively.
H1Bar H2Bar	Positions the first and second horizontal bar cursors to specified absolute locations.
RHOZero	Calibrates the rho scale for the selected trace.
V1Bar V2Bar	Positions the first and second horizontal bar cursors to specified absolute locations.

Cursors and the Selected Trace

All cursor commands and queries apply to the selected trace. When no traces are defined, there is no selected trace. Cursor parameters cannot be set or queried without a selected trace.

By default, the most recently defined trace is the selected trace. Use the SELect command to select a different trace when more than one trace appears on the display.

Each displayed trace has a unique set of cursor parameters. Therefore, cursor queries and set commands access cursor data that apply only to the selected trace. Altering the cursor parameters of the selected trace has no effect on the cursor data of any other trace.

Cursor Positioning Methods

Dot cursors may be horizontally positioned by graticule divisions, percentage of the trace record length, or horizontal units of the selected trace. Dot cursors can also be positioned relative to their current position.

Bar cursors may be horizontally or vertically positioned by graticule divisions or the units of the selected trace.

When cursors are positioned or queried by graticule divisions, use the illustration shown in the DOT1Abs command entry in the *Command Set* subsection to interpret the dimensions of graticules.

Data Transfer Commands

Data transfer commands transfer trace information and front panel settings to and from the instrument through the external interfaces. The Data Transfer commands are presented in two groups: data transfer execution commands in Table 2-17 and data transfer parameter commands in Table 2-18.

Table 2-17: Data Transfer Execution Commands

Command	Meaning
CURVe	Transfers unscaled trace data. Scaling information is included in the trace preamble. (See WFMPre command.)
SET?	Returns the front panel settings to the controller.
VPCurve?	Transfers all trace records used in a variable persistence trace.
WAVfrm?	Returns the trace preamble and trace data points for the trace specified by OUTPut. WAVfrm? is equivalent to entering WFMPRE?;CURVE?
WFMPre	Contains the links of the trace preamble.

Table 2-18: Data Transfer Parameter Commands

Command	Meaning
ABBwfmpre	Controls whether a WFMPre? query returns all links or an abbreviated set of links.
BYT.or	Sets the byte order for binary data transfer.
ENCdg	Selects ASCII or binary format for data transfers.
INPut	Selects the memory location in which to store a trace transferred to the instrument.
OUTput	Selects the stored or displayed traces to be transferred from the instrument.

Retrieving Trace Data From the Instrument

In general, the controller retrieves trace data from the instrument in the following sequence:

1. Use the `ENCdg` command to select trace encoding (ASCII or binary).

2. If binary encoding is in effect, use the `BYT.or` command to select the binary data transmission order (`LSB` or `MSB`) appropriate for the target controller.

3. Use the `OUTput` command to select the stored or displayed traces that are to be retrieved.

4. Finally, use `WFMPre?` and `CURVe?` (or `WAVfrm?`) to query the instrument for the preamble and unscaled trace data of the trace selected by `OUT-put`.

Notice that once you establish trace encoding and the binary data transmission order, steps 1 and 2 can be omitted for all subsequent retrievals.

Sending Trace Data to the Instrument

In general, trace data are sent back to the instrument as a stored trace in the following sequence:

NOTE

The controller normally sends trace data that came from a previous query.

1. If the trace data are binary encoded, use `BYT.or` to specify the transmission order of the returned data.

2. Use the `INPut` command to select the stored trace location where the data will be written.

3. Use the `WFMPre` command to return the trace preamble to the instrument.

4. Finally, use the `CURVe` command to return unscaled trace data to the instrument.

Diagnostics Commands

The Diagnostics commands invoke self-tests diagnostics or extended diagnostics. The Diagnostics commands are shown in Table 2-19.

Table 2-19: Diagnostics Commands

Command	Meaning
DIAg?	Returns the result of the diagnostic tests.
TESt	Performs self-test or extended-test diagnostics.

TESt deletes all user-defined expansion strings created with the DEFine command, resets the TEXt X:, Y: coordinates to 0,0, and removes user-entered text from the display.

Display and Color Commands

Display commands control how traces appear on the display, Histogram analysis, and Mask testing. Color commands determine the colors used on the display. The Display and Color commands are shown in Table 2-20.

Table 2-20: Display and Color Commands

Command	Meaning
COLor	Determines the colors used for traces and other display features.
DISPlay	Controls display intensity, number of graticules, dot/vector mode, and transfers of color graded pixel data.
DSYS	Selects the Display Modes major menu to enable Histograms and Mask testing.
GRAticule	Sets X and Y axis units and scaling.
HISTOgram	Controls vertical and horizontal histogram displays for active traces.
MASK	Creates mask polygons for trace testing.
MASKStat	Starts Mask test of traces and returns Mask statistics.
STATHist	Returns statistical information created by the HISTOgram function.
TEXt	Writes user defined character(s) to a selected area of the screen.

External I/O Commands

The External I/O commands produce hardcopy printout of the display, set the parameters for the RS232 interface, and control the debug feature for the GPIB and RS-232-C interfaces. The External I/O commands are presented in two groups: interface commands in Table 2-21 and printer commands in Table 2-22.

Table 2-21: Interface Commands

Command	Meaning
DEBug	Displays the ASCII commands on the front panel as they are executed via the specified interface.
FEOi	Forces a message terminator in a command string.
RS232	Sets the parameters of the RS-232-C interface.

Table 2-22: Printer Commands

Command	Meaning
ALTinkjet	Controls HP Thinkjet and HP LaserJet printers.
BITMap	Controls screen capture by an external computer.
COPy	Produces a printout of the display.
DSKJETC	Controls color DeskJet printers (Same as HP560).
HP560C	Controls HP 560C DeskJet printers.
HPGl	Controls Tektronix HC100 plotters and other devices conforming to the HP-GL format.
LSRJET	Controls black and white, HP Thinkjet, and HP LaserJet printers (Same as ALTinkjet).
PIN8	Controls standard Epson 8-pin bit image graphics printers, such as the Tektronix 4644.
PIN24	Controls extended Epson 24-pin dot graphics printers.

Label and Text Commands

Label and Text commands control how user text is placed on the display. Labels are text you create to identify traces and front panel settings. The Label and Text commands are shown in Table 2-23.

Table 2-23: Label and Text Commands

Command	Meaning
LABAbs	Positions the label associated with the selected trace to an absolute location.
LABel	Defines and deletes labels, and controls whether they are displayed.
LABRel	Positions the label associated with the selected trace to a location relative to its former absolute location.
TEXt	Defines and positions a text string on the display.

Measurement Commands

The Measurement commands take trace measurements and control the parameters with which these measurements are taken. In this discussion, the measurement system features are briefly presented, then the measurement commands are defined, followed by measurement parameter commands, and finally measurement execution commands. For a detailed discussion of the measurement system, refer to the *User Manual* for your instrument.

Selecting Measurements

Measurements initiated from the GPIB/RS-232-C interface operate differently than front panel measurements. From the ASCII interface, measurements can be specified for any trace or for a range of traces.

Each displayed trace has a set of user selected measurements and a set of user controlled measurement parameters that define how the measurements will be taken. Select measurements for a trace with the MSLIst command. The selected measurements are taken continuously while the Measure major menu is selected (MSYs command). Use the MEAS*<ui>* command to return the measurement results for all trace measurements. Or, on the selected trace only, you can query individual measurements with *<meas>?* command (e.g., RISE?, CROss?). Individual measurement queries are allowed regardless of the major menu selected and they need not be specified with MSLIst.

Measurement Modes: Hardware, Software, and Software Statistics

The CSA 803C and 11801C support software, hardware, and software statistical modes for taking measurements. The `MPARAM MMODE` command selects the mode.

The software mode (`SW`) calculates measurements from the trace record data. Most measurements can be performed in the software mode. This is the default measurement mode.

The hardware mode (`HW`) uses special hardware timer circuits to perform timing measurements on samples before they are digitized and assembled into a trace record. The `HW` mode provides measurement results much quicker than the `SW` mode. Only timing measurements may be taken in the hardware mode.

The measurements `CROss`, `FALltime`, `FREq`, `PDElay`, `PERiod`, `RISetime`, and `WIDth` can be taken in the hardware mode. The other measurements may be taken in the software mode only.

The Software Statistical Mode (`STAT`) calculates trace measurement parameters using histograms from the color graded data base. All measurements, except the three Frequency Domain measurements, can be performed in software statistics mode.

Measurement Statistics

The CSA 803C and 11801C can collect statistics on a specified measurement taken on the selected trace. Select a measurement and the sample size (number of measurement samples to include) with the `STATIstics` command. Use the `STAT?` command to return the statistical mean, standard deviation, and the number of samples used. Measurement statistics are available for software and hardware measurements only, because Statistical measurement mode automatically produces values based on many sets of waveform data.

Comparing Measurement With References

The `COMpare` command lets you compare a measurement to a reference value. The `REFSET` command sets the reference value to either the current value for each active measurement or to some absolute value for a single measurement *<meas>*.

Measurement (*<meas>*) Commands

Four types of measurements can be taken: timing, amplitude, area/energy, and frequency domain. The measurements are defined in Tables 2-24 through 2-27.

NOTE

No measurements can be taken on XY traces.

The symbol `<meas>` represents one or more measurements. A `<meas>` measurement can be individually queried or `<meas>` can represent one or more measurements in a `MSLIst` command.

Table 2-24: Timing Measurements

<meas>	Meaning
CROss	The time from the trigger point to a specified reference level crossing. HW, SW or STAT.
DUTy	Percentage of a period that a trace spends above the MESial level. SW or STAT.
FALltime	The transition time of a falling pulse edge, from the distal to proximal levels. HW, SW, or STAT.
FREq	Frequency (reciprocal of the period measurement). HW, SW, or STAT.
JITter	The jitter of the trace, calculated at the location (mesial or eye cross) that you specify with the MPARAM JITTLO command. STAT only.
PDElay	Propagation delay between mesial crossings on two traces (used with REFTRace). HW, SW, or STAT.
PERiod	The time between the first and next mesial crossing of the same slope. HW, SW, or STAT.
PHAse	The phase relationship of the reference trace to the selected trace, expressed as a value from −360 to +360 degrees. SW or STAT.
RISetime	The transition time of a rising pulse edge, from the proximal to distal levels. HW, SW, or STAT.
WIDth	The time between the first and next mesial crossing of the opposite slope. HW, SW, or STAT.

Table 2-25: Amplitude Measurements

<meas>	Meaning
AMPlitude	The amplitude (topline − baseline) of the selected trace. SW or STAT.
EXTInction	The extinction ratio of the selected trace. Three different formulas can be chosen via the MPAram ERMethod command. SW or STAT.
MAX	Maximum amplitude (most positive peak voltage). SW or STAT.
MEAN	Average amplitude (arithmetic mean voltage). SW or STAT.
MID	Amplitude midpoint, halfway between the maximum amplitude and the minimum amplitude. SW or STAT.
MIN	Minimum amplitude (most negative peak voltage). SW or STAT.
NOIse	The noise of the trace, calculated halfway between the left and right crossings, or halfway between the left and right measurement zones. STAT only.
OVErshoot	Difference between the maximum amplitude and the topline value, given as a percentage of the difference between the topline and baseline values. SW or STAT.
PP	Peak-to-peak value; the voltage difference between the maximum amplitude and minimum amplitude. SW or STAT.
RMS	True root-mean-square voltage. SW or STAT.
SNRatio	AMPlitude/NOIse. STAT only.
UNDershoot	Difference between the baseline value and the minimum amplitude, given as a percentage of the difference between the topline and baseline values. SW or STAT.

Table 2-26: Area/Energy Measurements

<meas>	Meaning
YTEnergy?	The energy represented under the curve of a YT trace (this integral of the squared voltages can be divided by the resistance of the circuit to yield power measurements). SW or STAT.
YTMns_area?	The difference between the area under a Yt curve above a specified reference level, and the area under the curve below that level. SW or STAT.
YTPls_area?	The total, absolute value of all areas between a YT trace and a user-specified reference level. SW or STAT.

Table 2-27: Frequency Domain Measurements

<meas>	Meaning
SFRequency?	The spectral frequency (harmonic or peak) of the selected trace. SW only.
SMAgnitude?	The spectral magnitude (harmonic or peak) of the selected trace. SW only.
THD?	The total harmonic distortion of the selected trace, that is, the magnitude of the fundamental divided by the sum of the magnitudes of all other harmonics. SW only.

Measurement Parameter Group

Measurement parameters are set with the `MPAram` command and the appropriate link for a specified trace. Some measurement parameters apply just to software measurements, some to statistical measurements, and some apply to hardware measurements while others apply to all three measurement modes. The `MPAram` command links shown in Table 2-28 set the measurement parameters.

Table 2-28: Measurement Parameter (MPARAM) Links

Command	Meaning
BASeline	Sets the absolute value of the baseline when measurement tracking (MTRack) is turned off or to TOPline.
DAInt	Sets the data interval (one waveform period or the entire measurement zone) for taking measurements. DAInt has no effect on any timing measurements in any mode.
DISPErsion	Sets whether the results of the queries JITter? and NOIse? are displayed as peak-to-peak or RMS deviation (Standard Deviation) values.
DISTal	Sets the distal (farthest from origin) reference level, either in % of the baseline-to-topline values (typically 90%), or in absolute voltage, or in a delta voltage from the baseline or topline.
ERMethod	Sets the extinction ratio method.
HNUmber	Selects the harmonic number on which the SMAgnitude and SFRequency measurements are made when SMOde is set to HARmonic.
JITT.histpt?	Valid only in MMOde STAT. Returns the number of histogram points used to calculate the jitter on the trace. Updates only if JITter is in the trace's measurement list and the trace is selected or a MEAS? query is sent.
JITTLEvel?	Valid only in MMOde STAT. Returns the jitter measurement level. Updates only if JITter is in the trace's measurement list and the trace is selected or a MEAS? query is sent.
JITTLOcation	Sets jitter measurement location on the trace. On an eye diagram JITTLOcation can be measured at the Eye cross (CROss) or at the mesial (MESial).
LMZone	Sets the left limit of the measurement zone.

Table 2-28: Measurement Parameter (MPARAM) Links (Cont.)

Command	Meaning
MESial	Sets the mesial (middle) reference level, either in % of the baseline-to-topline values (typically 50%), or in absolute voltage, or in a delta voltage from the baseline or topline.
MFIltering	Sets the filtering constant for hardware measurements.
MLEvelmode	Determines whether certain measurement parameters are absolute voltage values, or percentages of the baseline-to-topline value, or delta voltages from baseline (or topline).
MMOde	Selects the hardware, software, or statistical software measurement mode.
MSLOpe	Sets the crossing slope for measurements.
MTRack	Controls measurement tracking (whether you or the instrument set the baseline and topline values).
MTRANs	Sets the transition number on which hardware measurements begin.
NOIS.histpt?	Valid only in MMODE STAT. Returns the number of histogram points used to calculate the noise on the trace. Updates only if NOIse is in the trace's measurement list and the trace is selected or a MEAS? qeuery is sent.
NOISLocation	Sets whether the noise is measured around the trace's topline or baseline.
PINdex	Selects the peak index on which the SMAgnitude and SFRequency measurements are made when SMOde is set to PEAK.
PROXimal	Sets the proximal (nearest to origin) reference level, either in % of the baseline-to-topline values (typically 10%), or in absolute voltage, or in a delta voltage from the baseline or topline.
REFBaseline	Sets the absolute value of the baseline on the reference trace when measurement tracking (MTRack) is off or topline.
REFFiltering	Sets the filtering constant on the reference trace for hardware measurements.
REFLEvel	Sets the reference level used in CROss and other measurements, either in % of the baseline-to-topline values (typically 50%), or in absolute voltage, or in a delta voltage from the baseline or topline.

Table 2-28: Measurement Parameter (MPARAM) Links (Cont.)

Command	Meaning
REFLMzone	Sets the left limit of the measurement zone on the reference trace.
REFMesial	Sets the mesial (middle) reference level on the reference trace, either in % of the baseline-to-topline values (typically 50%), or in absolute voltage, or in a delta voltage from the baseline or topline.
REFRmzone	Sets the right limit of the measurement zone on the reference trace.
REFSLope	Sets the crossing slope on the reference trace.
REFSNratio	Sets the amplitude of a noise rejection centered on the MESial level of the reference trace.
REFTOpline	Sets the absolute value of topline on the reference trace when Tracking (MTRack) is off or baseline.
REFTRace	Selects the reference trace which provides the endpoint for some measurements.
REFXsition	Sets the transition number on the reference trace that will end hardware measurements.
RMZone	Sets the right limit of the measurement zone.
SMOde	Selects whether the SMAgnitude or SFRequency measurements are made on a selected harmonic or peak.
SNRatio	Sets the amplitude of a noise rejection band centered on the MESial level.
TOPline	Sets the absolute value of the topline when measurement tracking (MTRack) is turned off or set to BASeline.

Measurement Parameter Interactions — in Table 2-29 show how various parameters affect the measurement system.

Table 2-29: Measurement Parameter Interactions

Parameter(s)	Effects
BASeline TOPline	Used to calculate range values for PROXimal, MESial, DISTal, REFlevel, and REFMesial when MTRack is not set to BOTH and MLEvelmode is not set to ABSOLute.
DAInt	Limits the measurement zone for MEAN, RMS, YTEnergy, YTMns_area, and YTPls_area.
DISPErsion	PP or RMSDEV set whether the results of JITter? and NOIse? queries are displayed as peak-to-peak or RMS deviation (Standard Deviation) values.
DISTal PROXimal	Sets the measurement end points (voltage levels) for RISetime and FALltime *<meas>*.
JITTLOcation	CROss or MESial set jitter measurement location on the trace. On an eye diagram JITTLOCation can be measured at the Eye cross or at the mesial.
LMZone RMZone	Establishes a measurement zone to isolate part of a trace.
MESial	Sets measurement endpoints for DUTy, FREq, PDElay, PERiod, PHAse, and WIDth *<meas>*; and when DAInt is set to SINgle, for MEAN, RMS, YTEnergy, YTMns_area and YTPls_area.
MLEvelmode	Determines whether range values for PROXimal, MESial, DISTal, REFMesial, and REFLEvel are interpreted as percentages, or as absolute numbers, or as deltas from baseline or topline.
MMOde	Selects hardware, software, or software statistical measurement modes. Only timing measurements can be taken in the hardware mode.
MTRack	Determines whether BASeline and TOPline are set dynamically by the instrument using measurement tracking or are set by you. Neither, one, or both can be set dynamically. If either baseline and/or topline is not set dynamically by the instrument, then it can be set by you.
MTRANs, REFXsition	Selects the signal transition number on which to begin and end the PDElay measurement. The transitions are defined by slope, mesial level, and filtering constant.
NOISLocation	TOPline or BASeline set whether the noise is measured around the trace's topline or baseline.

Table 2-29: Measurement Parameter Interactions (Cont.)

Parameter(s)	Effects
REFLEvel	Sets the crossing level for CROss and a comparison level for YTMns_area *<meas>*. MMOde and MLEvelmode affect the range. Also used in SFRequency and SMAgnitude when SMOde is PEAK.
REFTRace	Selects the reference trace used with PDElay, and PHAse *<meas>*. The reference trace has a set of parameters that control the endpoint of some timing measurements (e.g., REFBaseline, REFFiltering, REFLEvel, REFSLope, etc.).
SNRatio	Noise filter when MMOde set to SW for DUTy, FREq, PDElay, PERiod, PHAse, and WIDth ; and when DAInt is set to SINgle, for MEAN, RMS, YTEnergy, YTMns_area and YTPls_area.

Measurement Execution Commands

The commands shown in Table 2-30 control the taking of measurements and the measurement execution modes.

Table 2-30: Measurement Execution Commands

Command	Meaning
COMpare	Controls comparison mode. When comparison mode is on, reference values set with the REFSET parameter are subtracted from measurement values; the difference is returned.
EYEPattern	Specifies that the selected trace is an eye pattern and automatically sets the mesial at the crossing level.
MEAS?	Executes and returns the values of the measurements in the measurement list (MSLIst).
<meas>?	Executes and returns the value of the specified measurement (<meas>) taken on the selected trace.
MSLIst	Selects the measurements (<meas>) for a trace which are executed continuously in the Measure major menu, and are executed once at a MEAS? query.
MSNum?	Returns the number of measurements in the measurement list (MSLIst).
MSYs	Controls display of the Measure major menu which must be selected to collect measurement statistics.
REFSET	Sets reference values on the specified trace for comparison measurements with the COMpare command.
STAT?	Returns measurement statistics (mean and standard deviation) for the <meas> specified with STATIstics.
STATIstics	Selects measurements for statistics calculations.
TOPBase	Selects the method of determining topline and baseline. (No effect when MMOde is STAT.)

Miscellaneous/ System Commands

The Miscellaneous/System commands include front panel commands, system status queries, and other useful functions. The Miscellaneous/System commands are presented in two groups: front panel commands in Table 2-31 and system commands in Table 2-32.

Table 2-31: Front Panel Commands

Command	Meaning
ABSTouch	Mimics a touch to the front panel display area or the major menu buttons.
CHIdent	Controls the function of the channel indicator light on the sampling head front panel.
DSYmenu?	Returns which major menu is currently displayed.
FPAnel	Controls front panel lockout.
FPUpdate	Controls when front panel readouts are updated.
SPEaker	Controls whether the instrument beeps when the display is touched.

Table 2-32: System Commands

Command	Meaning
BCOrrection	Sets baseline correction ON or OFF for all acquired traces.
DATE	Sets the date on the system calendar.
DEFine	Defines logical names for command strings.
INIt	Initializes the system.
LONgform	Controls whether the instrument returns full or abbreviated query responses and event information.
POWeron?	Returns the number of times the instrument has been powered on.
TIMe	Sets the time on the system clock.
UNDEF	Deletes logical names previously defined with DEF.
UPTime?	Returns the number of hours the instrument has been powered on.

Status Queries and Event Commands

Status Queries and Event commands report identifying information and operating status of the instrument to an external controller or device. Status queries are presented in Table 2-33 and Event commands in Table 2-34.

Table 2-33: Status Queries

Command	Meaning
ID?	Returns version numbers of system firmware.
UID?	Returns serial numbers of the instrument and its sampling heads.

Table 2-34: Event Commands

Command	Meaning
EVENT?	Returns event code information.
RQS	Sets whether the instrument asserts the SRQ line after an event occurs (GPIB only).
SAMid?	Returns the model number of sampling heads.
SRQMask	Controls (masks) reporting of certain classes of event.
STByte?	Returns status byte information (RS-232-C only).

Time Base/ Horizontal Commands

The Time Base/Horizontal commands control horizontal position, establish acquisition scaling, and select a time base. The Time Base/Horizontal group contains the commands shown in Table 2-35.

Table 2-35: Time Base/Horizontal Commands

Command	Meaning
MAINPos	Sets the horizontal position of the main trace record with respect to the trigger.
TBMain	Sets the main horizontal (time base) parameters.
TBWin	Sets the window horizontal (time base) parameters.

Trace and Settings Commands

Trace and Setting commands select, store, remove, and specify trace and front panel setting (FPS) characteristics. Trace and Settings commands are presented in two groups: Trace commands in Table 2-36 and front panel settings commands in Table 2-37.

Table 2-36: Trace Commands

Command	Meaning
ADJtrace<*ui*>	Controls pan/zoom mode, vertical size and position, and window trace separation.
CLEar	Discards acquired data for displayed traces.
DELete	Deletes stored traces or front panel settings.
DISPNum?	Returns the number of traces displayed.
HREfpt	Sets the horizontal pivot point about which a trace is expanded.
MAXTranum?	Returns the highest trace number possible with the current configuration. (With the CSA 803C, the returned number is always 8.)
REMove	Discards displayed traces and their descriptions.
SELect	Designates the selected trace.
STD [<*ui*>] ?	*11801C only.* Returns the saved trace description specified by <*ui*>. When <*ui*> is omitted, returns all saved trace descriptions.
STOList?	Returns a list of all stored traces.
STONum?	Returns the number of stored traces.
STORe	Copies displayed traces to memory.
TRAce<*ui*>	Defines a trace and its characteristics.
TRAList?	Returns a list of all displayed traces.
TRANUm?	Returns the number of displayed traces.
WFMScaling	Sets a scaling flag to create new traces in fast (integer) or high-precision (floating-point) mode.
WIN	Sets a variety of window trace parameters including window position and autowindow placement.
WINList?	Returns the numbers of displayed window traces.
WINNum?	Returns the total number of defined window traces.

Table 2-37: Front Panel Settings Commands

Command	Meaning
DELete	Deletes stored traces or front panel settings.
FPSList?	Returns a list of stored front panel settings.
FPSNum?	Returns the number of stored front panel settings.
NVRam?	Returns the amount of available nonvolatile RAM.
RECall	Recalls stored front panel settings from memory.
SETseq	Controls sequencing of front panel settings.
STORe	Stores front panel settings in nonvolatile RAM.

Triggering Commands

The Trigger command selects and defines the operation of the triggering system. The Trigger command is shown in Table 2-38.

Table 2-38: Triggering Commands

Command	Meaning
TRIgger	Sets trigger parameters such as source, level, mode, and slope and returns its current status.

Alphabetic Command Summary

The following key for the Alphabetic Command Summary provides a quick definition of symbols and data elements used in Table 2-39.

`< >`	::= Defined item
`{ }`	::= One item from group required
`[ ]`	::= Optional item(s)
`( )`	::= Grouped items
`|`	::= Exclusive OR
`FPS`	::= Front Panel Setting
`<NR1>`	::= Signed integer
`<NR2>`	::= Floating point, no exponent
`<NR3>`	::= Floating point with exponent
`<NRx>`	::= { `<NR1>` | `<NR2>` | `<NR3>` }
`<ui>`	::= Unsigned integer
`<curve data>`	::= Tek Codes and Formats binary block data (`<bblock>`) or ASCII data points (`<NR1>`[{,`<NR1>` }...})
`<meas>`	::= One or more of the measurement commands that apply to the selected waveform unless a specific waveform is designated
`<qstring>`	::= Quoted string
`?`	::= Query-only header or link

`HEAder`	Header, link, or argument; minimum spelling in `CAPs`
`RESponse`	Query response; minimum spelling in `CAPs`

Commands are set/query unless otherwise noted. Query-only headers are followed by a `?`. Query-only links are indicated with a leading `?`; the argument(s) in parentheses after the colon show the response form. (Do not enter the colon when querying a link.)

Table 2-39: Alphabetical Command Summary

Command	Link	Set/Query Only
`ABBwfmpre { ON	OFF }`		
`ABSTouch <NRx>,<NRx>`	`<NRx>,<NRx>`	
`ACQNum?`		Query Only
`ACQuisition { RUN	STOP }`		
`ADJtrace<ui> <link>:<arg>`	`COLor: <arg>`	
	`GRLocation: { UPPer	LOWer }`	
	`HMAg: <NRx>`	
	`HPOsition: <NRx>`	
	`PANzoom: { ON	OFF }`	
	`TRSep: <NRx>`	
	`VPOsition: <NRx>`	
	`VSIze: <NRx>`	
`ALTinkjet <link>:<arg>`	`? COLor`	Query Only
	`? COLORn`	Query Only
	`COLORn:<ui>`	
	`DIRection: { HORiz	VERt }`	
	`FORMat: { DIThered	DRAft	HIRes	REDuced	SCReen }`	
	`PORt: { CENTRonics	GPIb	RS232 }`	
`AMPlitude?`		Query Only
`AUTOSet [ <link>: ] <arg>`	`HORiz: { ON	OFF }`	
	`MASK: { AUTO	MANual }`	
	`MODe: { EDGe	PERiod }`	
	`SMOFfadj: { ON	OFF }`	
	`STARt`	Set Only
	`TRIgger: { ON	OFF }`	
	`UNDO`	Set Only
	`VERt: { ON	OFF }`	

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
AVG { ON | OFF }		
BCOrrection { ON | OFF }		
BITMap *<link>*:*<arg>*	DATACompress: { ON | OFF }	
	DATAFormat: { BINary | BINHex | TIFf }	
	DIRection: { HORiz | VERt }	
	FORMat: { DIThered | DRAft | HIRes | REDuced | SCReen }	
	PORt: { CENTRonics | GPIb | RS232 }	
BYT.or { LSB | MSB }		
CALibrate *<alpha><ui>* *<link>*: *<arg>*	AUTO: { BLOwby | DADj | LOOpgain | OFFSet | ONUll | TDRAmplitude }	Set Only
	BLOwby: *<NRx>*	
	CSTore: { ALL | BLOwby | LOOpgain | OFFSet | ONUll | TDRAmplitude }	Set Only
	DADj: *<NRx>*	
	DRECall: { ALL | BLOwby | LOOpgain | OFFSet | ONUll | TDRAmplitude }	Set Only
	LOOPgain: *<NRx>*	
	ONUll: *<NRx>*	
	TDRAmplitude: *<NRx>*	
	URECall: { ALL | BLOwby | LOOpgain | OFFSet | ONULl | TDRAmplitude }	Set Only
CH *<alpha><ui><link>*:*<arg>*	? ATTenuation: { X1 | X2 | X5 | X10 }	Query Only
	CDELay: *<NRx>*	
	DATAType: { REPetitive | RANdom }	
	EXTAttenuation: *<NRx>*	
	OFFSet: *<NRx>*	
	SENSitivity: *<NRx>*	
	SMOothing: { ON | OFF }	
	TDRDelay: *<NRx>*	

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
	TDRPolarity: { PLUs | MINUs }	
	TDRState: { ON | OFF }	
	TOFfset: *<NRx>*	
CHIdent { FLAsh | QUIet }		
CLEar { ALLTrace | *<qstring>* | TRAce*<ui>* }		Set Only
COLor *<ui>* *<link>*:*<arg>*	DEFAult	
	HUE: *<NRx>*	
	LIGhtness: *<NRx>*	
	SATuration: *<NRx>*	
COLor DEFAult		
COMpare { ON | OFF }		
CONDacq *<link>*:*<arg>*	? REMAining *<NR1>*	Query Only
	TYPe: { AVG | AVG.env | BOTh | CONTInuous | ENV | GRADed | HIST.pt | MASK | MASK*<ui>* | RECOrd | WAVfrm }	Set Only
	WAIT	Set Only
COPy [*<link>*:] *<arg>*	ABOrt	Set Only
	FORMat: { DIThered | DRAft | HIRes | REDuced | SCReen }	Set Only
	KILl	Set Only
	PRINter: { ALTinkjet | BITMap | DSKJETC | HP560C | HPGl | LSRJET | PIN8 | PIN24 | }	
	STARt	Set Only
	? STATUs: { IDLe | ABORTING | PRINting | SPOoling }	Query Only
CROss?		Query Only
CURSor *<link>*:*<arg>*	? MODE: { ABSolute | RELative }	Query Only
	REAdout: { ON | OFF }	
	REFErence: TRAce*<ui>*	

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
	SETZero: { CUR1 | CLEar }	Set Only
	TYPe: { PAIred | SPLit | VBArs | HBArs }	
	? XUNit { DIVS | FEEt | INChes | METers | SEConds | VOLts }	Query Only
	? YUNit: { DIVS | RHO | VOLts }	Query Only
	? ZEROPoint: *<NRx>*	Query Only
CURVe [CRVId:*<arg>*,] *<curve data>*	[CRVid: { STO | TRAce } *<ui>*] | <curvedata>	
DAFiltering *<alpha><ui>*: *<NR1>*	*<alpha><ui>*: *<NR1>*	
DAMeasref *<alpha><ui>*		Set Only
DATE *<qstring>*	<qstring> "*<dd>-<mon>-<yy>*"	
DCOmp { ON | OFF }		
DEBug *<link>*:*<arg>*	GPIb: { ON | OFF }	
	RS232: { ON | OFF }	
DEFine *<qstring>,<qstring>* [? [*<qstring>*]	*<qstring>,<qstring>*	
DELete [*<link>*:] *<arg>*	{ ALLFps | ALLSTD (11801C only) | ALLSTO }	Set Only
	FPS*<ui>*	Set Only
	<qstring>	Set Only
	STD*<ui>* (11801C only)	Set Only
	STO*<ui>*	Set Only
DIAg?		Query Only
DISPlay *<link>*:*<arg>*	CLEar	Set Only
	C.WINBottom: *<NRx>*	
	C.WINLeft: *<NRx>*	
	C.WINRight: *<NRx>*	
	C.WINTop: *<NRx>*	
	? DATA	Query Only

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
	D.WINBottom: *<NRx>*	
	D.WINLeft *<NRx>*	
	D.WINRight *<NRx>*	
	D.WINTop: *<NRx>*	
	GRAticule: { DUAl | SINgle }	
	GRADFirst: { ON | OFF }	
	? GRADScale:	Query Only
	INTENsity: *<NRx>*	
	MODe: { DOTs | VECtors }	
	? NR.PT *<NRx>*	Query Only
	PERSistence: *<NRx>*	
	REFREsh: { 0 | *<NRx>* }	
	STATIstics: { HISTogram | MASK }	
	TYPe: { INFinite | NORmal | VARiable | GRADed }	
	? XSize: *<NRx>*	Query Only
	? YSize: *<NRx>*	Query Only
DISPNum?		Query Only
DIV2 { ON | OFF }		
DOT1Abs *<link>*:*<arg>*; DOT2Abs *<link>*:*<arg>*	? OHMS: *<NR3>*	Query Only
	? OQUal: { EQ | LT | GT | UN | ER }	Query Only
	PCTg: *<NRx>*	
	XCOord: *<NRx>*	
	XDIv: *<NRx>*	
	? XQUal: { EQ | LT | GT | UN }	Query Only
	? YCOord: *<NR3>*	Query Only
	? YDIv: *<NR3>*	Query Only
	? YQUal: { EQ | LT | GT | UN }	Query Only

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
DOT1Rel *<link>*:*<arg>* ; DOT2Rel *<link>*:*<arg>*	PCTg: *<NRx>*	Set Only
	XCOord: *<NRx>*	Set Only
	XDIv: *<NRx>*	Set Only
DSKJETC *<link>*:*<arg>*	? COLor	Query Only
	? COLORn	Query Only
	COLORn:*<ui>*	
	DIRection: { HORiz | VERt }	
	FORMat: { DIThered | DRAft | REDuced | SCReen }	
	PORt: { CENTRonics | GPIb | RS232 }	
DSYmenu?		Query Only
DSYS	{ ON | OFF }	
DUTy?		Query Only
ENCdg *<link>*:*<arg>*	DISPlay: { ASCii | BINary }	
	HISTogram: { ASCii | BINary }	
	SET: { ASCii | BINary }	
	WAVfrm: { ASCii | BINary }	
ENV { ON | OFF }		
EVENT?		Query Only
EXTInction?		Query Only
EYEPattern { ON | OFF }		
FALltime?		Query Only
FEOi		Set Only
FFT { *<link>*:*<arg>* | ? *<link>*: *<arg>* }	FORMat: { DBM | LINear | VPEak }	
	WINDow: { BLACKman | BLHarris | HAMming | HANning | RECTangular | TRIAngular }	
FMOde { CENter | SHIft }		
FPAnel { ON | OFF }		

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
FPSList?		Query Only
FPSNum?		Query Only
FPUpdate { ON	OFF }		
FREq?		Query Only
GRAticule *<link>*:*<arg>*	PROPvelocity: *<NRx>*	
	REFAmplitude: *<NRx>*	
	XUNit: { FEEt	INChes	METers	SEConds }	
	YUNit: { VOLts	RHO }	
H1Bar *<link>*:*<arg>*;	? OHMS: *<NR3>*	Query Only
H2Bar *<link>*:*<arg>*	? OQUal: { EQ	LT	GT	UN	ER }	Query Only
	YCOord: *<NRx>*	
	YDIv: *<NRx>*	
HISTogram *<link>*:*<arg>*	CLEar	
	C.WINBottom: *<NRx>*	
	C.WINLeft: *<NRx>*	
	C.WINRight: *<NRx>*	
	C.WINTop: *<NRx>*	
	? DATA	Query Only
	D.WINBottom: *<NRx>*	
	D.WINLeft: *<NRx>*	
	D.WINRight: *<NRx>*	
	D.WINTop: *<NRx>*	
	HISTScaling: { LINear	LOG10 }	
	? NR.PT *<NRx>*	Query Only
	TYPe: { HORiz	VERt	NONe }	

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
HP560C *<link>*:*<arg>*	? COLor	Query Only
	? COLORn	Query Only
	COLORn:*<ui>*	
	DIRection: { HORiz | VERt }	
	FORMat: { DIThered | DRAft | REDuced | SCReen }	
	PORt: { CENTRonics | GPIb | RS232 }	
HPGl *<link>*:*<arg>*	COLor*<ui>*: *<NRx>*	
	COLor: DEFAult	Set Only
	FORMat: { DRAft | HIRes | SCReen }	
	PORt: { CENTRonics | GPIb | RS232 }	
HREfpt { LEFt | CENter | RIGht }		
ID?		Query Only
INIt		Set Only
INPut { STO*<ui>* | *<qstring>* }	{ STO*<ui>* | *<qstring>* }	
JITter?		Query Only
LABAbs *<link>*:*<arg>*	PCTg: *<NRx>*	
	XCOord: *<NRx>*	
	YDIv: *<NRx>*	
LABel *<link>*:*<arg>*	DELete: { ALL | FPS [*<ui>*] | *<qstring>* | STO [*<ui>*] | TRAce [*<ui>*] }	Set Only
	DISPlay: { ON | OFF }	
	FPS*<ui>*: *<qstring>*	
	MODe: { AUTO | MANual }	
	STO*<ui>*: *<qstring>*	
	TRAce*<ui>*: *<qstring>*	

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
LABRel *<link>*:*<arg>*	PCTg: *<NRx>*	Set Only
	XCOord: *<NRx>*	Set Only
	YDIv: *<NRx>*	Set Only
LONgform { ON | OFF }		
LSRJET *<link>*:*<arg>*	? COLor	Query Only
	? COLORn	Query Only
	COLORn:*<ui>*	
	DIRection: { HORiz | VERt }	
	FORMat: { DIThered | DRAft | HIRes | REDuced | SCReen }	
	PORt: { CENTRonics | GPIb | RS232 }	
MAINPos *<NRx>*	<NRx>	
MASK*<ui>* *<link>*:*<arg>*	C.Point: *<mpoint>*	
	DELete	
	D.Point: *<mpoint>*	
	? NCOunt *<NRx>*	Query Only
	? NR.PT *<NRx>*	Query Only

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
MASKDefine { OC1 | OC3 | OC9 | OC12 | OC18 | OC24 | OC36 | OC48 | CC.SI.64kb | CC.DO.64kb | CC.DA.64kb | CC.TI.64kb | CC.1.5mb | CC.SY.2mb | CC.CO.2mb | CC.SY.6mb | CC.CO.6mb | CC.8mb | CC.32mb | CC.34mb | CC.44mb | CC.97mb | CC.ZEro.139mb | CC.ONe.139mb | DS1 | DS1.Old | DS1C | DS2 | DS3 | DS4NA | DS4XNA | FC133 | FC133E | FC266 | FC266E | FC531 | FC531E | FC1063 | FC1063E | GBETHERNET | STS1 | STS3 | STSX3 | STM1 | FDDI | CEPT | USERMask | USEREye | NONe }		
MASKStat *<link>*:*<arg>*	CLEar	
	COUnt: { ON | OFF }	
	? NWFm *<NRx>*	Query Only
	? TOTal *<NRx>*	Query Only
MAX?		Query Only
MAXTranum?		Query Only
MEAN?		Query Only
MEAS [*<ui>* [TO*<ui>*]] ?		Query Only
<meas>?		Query Only
MID?		Query Only
MIN?		Query Only
MPAram*<ui>* *<link>*:*<arg>*	BASeline: *<NRx>*	
	DAInt: { WHOle | SINgle }	
	DISPErsion: { PP | RMSDev }	

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
	DISTal: *<NRx>*	
	ERMethod: { NORmal	FDDI	SONet }	
	FROMMLevelmode: { ABSolute	RELative	BASEDelta	TOPDelta }	
	FROMMTrack: { ON	OFF	BASeline	TOPline	BOTh }	
	HNUmber: *<ui>*	
	JITTLOcation: { CROss	MESial }	
	? JITTLEvel: *<NRx>*	Query Only
	? JITT.histpt: *<NRx>*	Query Only
	LMZone: *<NRx>*	
	MESial: *<NRx>*	
	MFIltering: *<NRx>*	
	MLEvelmode: { ABSolute	RELative	BASEDelta	TOPDelta }	
	MMode: { HW	SW	STAT }	
	MSLOpe: { PLUs	MINUs }	
	MTRack: { ON	OFF	BASeline	TOPline	BOTh }	
	MTRANs: *<NRx>*	
	NOISLocation: { TOPline	BASeline }	
	? NOIS.histpt: *<NRx>*	Query Only
	PINdex: *<ui>*	
	PROXimal: *<NRx>*	
	REFBaseline: *<NRx>*	
	REFFiltering: *<NRx>*	
	REFLEvel: *<NRx>*	
	REFMLevelmode: { ABSolute	RELative	BASEDelta	TOPDelta }	
	REFLMzone: *<NRx>*	

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
	REFMesial: *<NRx>*	
	REFMTrack: { ON | OFF | BASeline | TOPline | BOTh }	
	REFRzone: *<NRx>*	
	REFSLope: { PLUs | MINUs }	
	REFSNratio: *<NRx>*	
	REFTOpline: *<NRx>*	
	REFTRace: TRAce*<ui>*	
	REFXsition: *<NRx>*	
	RMZone: *<NRx>*	
	SMOde: { PEAK | HARMonic }	
	SNRatio: *<NRx>*	
	TOPline: *<NRx>*	
MSLISt*<ui>* *<meas>* [, *<meas>* .]		
MSNum [*<ui>*] ?		Query Only
MSYs { ON | OFF }		
NAVg *<NRx>*	*<NRx>*	
NENV *<NRx>*	*<NRx>*	
NGRAded *<NRx>*	*<NRx>*	
NHISt.pt *<NRx>*	*<NRx>*	
NMAsk *<NRx>*	*<NRx>*	
NOIse?		Query Only
NVRam?		Query Only
NWAVfrm *<NRx>*	*<NRx>*	
OUTput *<arg>*	{ ALLSTO | ALLTrace }	
	STO*<ui>*	
	STO*<ui>*TOSTO*<ui>*	

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
	TRAce*<ui>*	
	TRAce*<ui>*TOTRAce*<ui>*	
	<qstring>	
OVErshoot?		Query Only
PDElay?		Query Only
PERiod?		Query Only
PHAse?		Query Only
PIN8 *<link>*:*<arg>*	FORMat: { DRAft | HIRes | REDuced }	
	PORt: { CENTRonics | GPIb | RS232 }	
PIN24 *<link>*:*<arg>*	FORMat: { DRAft | HIRes | REDuced }	
	PORt: { CENTRonics | GPIb | RS232 }	
POWeron?		Query Only
PP?		Query Only
RECall { FPNext | FPS*<ui>* | *<qstring>* | STD *<sui>*:TRA *<dui>* }	FPNext	Set Only
	FPS*<ui>*	Set Only
	<qstring>	Set Only
	STD*<sui>*: TRAce*<dui>* (11801C only)	Set Only
REFSET*<ui>* *<link>*:*<arg>*	CURRent: *<meas>*	
	<meas>: *<NRx>*	
REMove { ALLTrace | TRAce*<ui>* | *<qstring>* }	ALLTrace	Set Only
	TRAce*<ui>*	Set Only
	<qstring>	Set Only
RHOPos*<ui>* *<NRx>*		
RHOZero		Set Only
RISetime?		Query Only
RMS?		Query Only
RQS { ON | OFF }		

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
RS232 *<link>*:*<arg>*	BAUd: *<NRx>*	
	DELAy: *<NRx>*	
	ECHo: { ON | OFF }	
	EOL: { CR | CRLf | LF | LRCr }	
	FLAgging: { SOFt | HARd | OFF }	
	PARity: { ODD | EVEN | NONe }	
	STOPBits: *<NRx>*	
	VERBose: { ON | OFF }	
SAMid? [*<alpha>* [*<ui>*]]		Query Only
SELect { TRAce*<ui>* | *<qstring>* }	TRAce*<ui>*	
	<qstring>	
SET?		Query Only
SETSeq { ON | OFF }		
SFRequency?		Query Only
SMAgnitude?		Query Only
SNRatio?		Query Only
SPEaker { ON | OFF }		
SRQMask *<link>*: { ON | OFF }		
STAT? [MEAN | STDDev | NCUrrent]		Query Only
STATHist *<link>*	? HIST.pt	Query Only
	? NWFm	Query Only
	? MEAN	Query Only
	? RMSDev	Query Only
	? PP	Query Only

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
	? SIGMA1	Query Only
	? SIGMA2	Query Only
	? SIGMA3	Query Only
STATIstics *<link>:<arg>*	MEAS: { *<meas>* | NONe }	
	N: *<NRx>*	
	RESET	
STBYte?		Query Only
STD [*<sui>*] ? (11801C only)		Query Only
STOList?		Query Only
STONum?		Query Only
STORe [*<link>*:] *<arg>*	{ FPS*<ui>* | *<qstring>* }	Set Only
	TRAce*<ui>*: { STO*<ui>* | *<qstring>* | STD*<ui>* (11801C only) }	Set Only
	<qstring>: STO*<ui>*	Set Only
TBCalmode { FASt | HIPrec | OFF | ONCE }		
TBMain *<link>:<arg>*; TBWin *<link>:<arg>*	LENgth: *<NRx>*	
	TIMe: *<NRx>*	
	? XINcr *<NR3>*	Query Only
TESt [XTNd]		Set Only
TEXt [*<link>*:] *<arg>*	CLEar	Set Only
	STRing: *<qstring>*	Set Only
	X: *<NRx>*	Set Only
	Y: *<NRx>*	Set Only
THD?		Query Only
TIMe *<qstring>*	*<qstring>*	
TOPBase *<arg>*	{ IEEE | ALTErnate }	

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
TRAce*<ui>* *<link>*:*<arg>*	DEScription: *<qstring>*	
	? WFMCalc { FASt | HIPrec }	Query Only
TRAList?		Query Only
TRANUm?		Query Only
TRIgger *<link>*:*<arg>*	ATTenuation { X1 | X10 } **11801C only**	
	ENHANced: { ON | OFF }	
	EXTCoupling { AC | DC } **11801C only**	
	HIFreq: { ON | OFF }	
	HOLdoff: *<NRx>*	
	HMOde: { AUTO | MANual }	
	LEVel: *<NRx>*	
	MODe: { AUTO | NORmal }	
	SLOpe: { PLUs | MINUs }	
	SOUrce: { EXTernal | INTernal | PREscaler | *<alpha><ui>* }	
	? STATUs { TRG | NOTrg }	Query Only
UID { *<link>*:*<arg>* }	? *<alpha><ui>* *<qstring>*	Query Only
	? MAIn: *<qstring>*	
UNDEF { *<qstring>* | ALL }		Set Only
UNDershoot?		Query Only
UPTime?		Query Only
USEREye *<link>*:*<arg>*	CREate	Set Only
	{ ? } RATe:*<NRx>*	
V1Bar *<link>*:*<arg>*; V2Bar *<link>*:*<arg>*	XCOord: *<NRx>*	
	XDIv: *<NRx>*	
VPCurve? { NWFm }	? NWFm *<NR1>*	Query Only
WAVfrm?		Query Only

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
WFMPre <*link*>:<*arg*>	? BIT/nr 16	Query Only
	? BN.fmt RI	Query Only
	? BYT/nr 2	Query Only
	? BYT.or { LSB | MSB }	Query Only
	? CRVChk { CHKsm0 | NONe | NULl }	Query Only
	DATE: <*qstring*>	
	? ENCdg { ASCii | BINary }	Query Only
	LABel: <*qstring*>	
	NR.pt: { 512 | 1024 | 2048 | 4096 | 5120 }	
	? PT.fmt: { ENV | Y | XY }	Query Only
	RHOFactor: <*NRx*>	
	RHOPos: <*NRx*>	
	TIMe: <*qstring*>	
	? WFId { STO<*ui*> | TRAce<*ui*> }	Query Only
	XINcr: <*NRx*>	
	XMUlt: <*NRx*>	
	? XUNit { DIVS | FEEt | INChes | METers | SEConds | VOLts }	Query Only
	XZEro: <*NRx*>	
	YMUlt: <*NRx*>	
	YUNit: { DIVS | RHO | VOLts }	
	YZEro: <*NRx*>	
WFMScaling { FORce | OPTional }		
WIDth?		Query Only

Table 2-39: Alphabetical Command Summary (Cont.)

Command	Link	Set/Query Only
WIN*<ui>* *<link>*:*<arg>*	? ABSLevel *<NRx>*,*<qual>*	Query Only
	? BASeline *<NRx>*	Query Only
	FILtering: *<NRx>*	
	LEVel: *<NRx>*	
	LMOde: { ABSolute | RELative }	
	LOCate	Set Only
	MODe: { AUTO | MANual }	
	POS: *<NRx>*	
	SLOpe: { PLUs | MINUs }	
	? TOPline *<NR3>*	Query Only
	TRACK: { ON | OFF }	
	TRANS: *<NRx>*	
WINList?		Query Only
WINNum?		Query Only
YTEnergy?		Query Only
YTMns_area?		Query Only
YTPls_area?		Query Only

Command Set

You can use commands either 1) to set instrument features or 2) to query instrument values. You can use some commands to do both, some to only set and some to only query. This section of the manual marks "set only" commands with the words "Set Only" included with the command description. Query only commands are marked with a question mark appended to the header and with the words "Query Only" in the command description. This section presents an alphabetical listing of all instrument commands.

The headers, mnemonics, and arguments for the commands are presented with the minimal spelling shown in upper case. For example, to use the abbreviated form of the `ABBwfmpre` command just type `ABB`.

ABBwfmpre { ON | OFF }

Group: Data Transfer Commands

`ABBwfmpre` determines whether the response to a `WFMPre?` query is abbreviated or includes all links. When `ABBwfmpre` is set to `ON` (i.e., abbreviated), the `WFMPre?` response is:

```
WFMPRE NR.PT:<NR1>, PT.FMT:<arg>, RHOFACTOR:<NR3>,
    RHOPOS: <NR3>, XINCR:<NR3>, XMULT:<NR3>,
    XZERO:<NR3>, YMULT:<NR3>, YZERO:<NR3>
```

When `ABBwfmpre` is set to `OFF`, the `WFMPre?` response includes all 21 links of the `WFMPre` command. The power-on default setting is `ABBwfmpre OFF`.

Examples: `ABB ON`

ABSTouch *<NRx>,<NRx>*

Group: Miscellaneous/System Commands

`ABSTouch` activates a location on the front panel by giving its X,Y coordinates. `ABSTouch` always works, regardless of the state of the front panel (`FPAnel ON/OFF`) or touch panel button. Touch coordinates, whether from `ABSTouch` or from the front panel, are stored in a first in, first out (FIFO) buffer. You can access the FIFO with the `ABSTouch?` query.

Link: *<NRx>,<NRx>*

Arguments: The touch coordinates are specified as X,Y.

X = 0 to 11; Y = 0 to 21

X,Y touch panel screen coordinates range from 0,0 (upper left) to 10,21 (lower right) as shown in Figure 2-2.
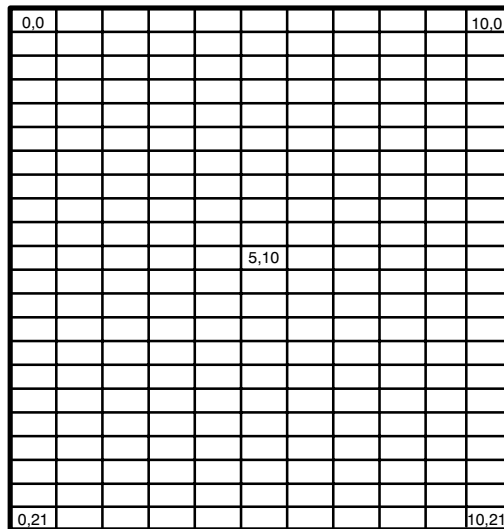
Figure 2-2: X, Y Touch Panel Screen Coordinates

Coordinates of the front panel buttons are listed in Table 2-40.

Table 2-40: Front Panel Button X, Y Coordinates

Button	X,Y	Button	X,Y
Waveform	11,0	Utility	11,5
Trigger	11,1	Touch Panel	11,6
Measure	11,2	Acquisition Run/Stop	11,7
Display Modes	11,3	Autoset	11,8
Store/Recall	11,4	Hardcopy	11,9
		Sequence Setting	11,10

NOTE

`ABSTouch` *cannot be used to touch a sampling head channel button.*

Examples: ABST 11,0

Query Note: Every front panel touch, whether from ABSTouch or the front panel, is stored in a 20-deep FIFO buffer. ABSTouch? returns the oldest touch coordinates and removes them from the buffer. If no touches are in the buffer, ABSTouch? returns:

ABSTOUCH −1, −1

Examples: **ABST?**
ABSTOUCH 11,6

ACQNum?

Group: Acquisition Commands

Query Only. ACQNum? returns the total number of acquisition systems. An acquisition system contains up to four channels.

Examples: **ACQN?**
ACQNUM 1

ACQuisition { RUN | STOP }

Group: Acquisition Commands

ACQuisition starts and stops unconditional trace acquisition.

Examples: ACQ RUN

ADJtrace*<ui> <link>:<arg>*

Group: Trace and Settings Commands

Adjusts the displayed position of the specified trace without modifying the time base or sampling unit parameters. Range of *<ui>* is 1 to 8.

Link: COLor: *<arg>*

COLor sets the front panel color for the specified trace to one of the predefined trace colors as seen in the front panel Color pop-up menu. The specified color can be any of the predefined trace colors listed in Table 2-41.

The trace colors are used to display new traces in their order of creation. The Window color is used for all created window traces. These trace colors can be set to any absolute color with the COLor command described on page 2-90.

Table 2-41: Predefined Trace Colors

Color Number	Color Function
1	Trace 1 Color
2	Trace 2 Color
3	Trace 3 Color
4	Trace 4 Color
5	Window Color

Examples: ADJ3 COL:2

Link: GRLocation: { UPPer | LOWer }

Positions the selected trace to the upper or lower graticule pair.

Examples: ADJ2 GRL:LOW

Link: HMAg: *<NRx>*

Arguments: The allowed values for *<NRx>* are 1, 2, 2.5, 4, 5, and 10.

HMAg sets the trace horizontal magnification factor when PANzoom is ON. The HMAg value depends on the record LENgth of TBMain or TBWin as shown in Table 2-42.

Table 2-42: Valid HMAg Values

Record LENgth	Valid HMAg Value(s)
512	1
1024	1, 2
2048	1, 2, 4
4096	1, 2.5, 5, 10
5120	1, 2.5, 5, 10

Examples: ADJ2 HMA:2

    Link: HPOsition: *<NRx>*

Arguments: *<NRx>* can range from 0 to 4608. The limit is less than 4608 if the record length is less than 5120 or if HMAg is less than ten.

    HPOsition sets the trace horizontal position when PANzoom is ON. HPOsition range is in trace points.

Examples: ADJ2 HPO:300

    Link: PANzoom: { ON | OFF }

    Sets pan/zoom mode to ON or OFF. PANzoom is always ON for stored or scalar traces, but cannot be ON for XY traces.

Examples: ADJ2 PAN:ON

    Link: TRSep: *<NRx>*

Arguments: *<NRx>* can range between −5.0 to +5.0.

    TRSep sets the window trace separation in graticule divisions only if the trace is not XY and was created on the Window time base in integer mode (TRACE WFMCALC:FAST).

Examples: ADJ3 TRS:−2.2

    Link: VPOsition: *<NRx>*

Arguments: *<NRx>* can range between −1E + 15 to 1E + 15.

    VPOsition sets the trace vertical graphical position only if the trace was created in floating-point mode (TRACE WFMCALC:HIPREC).

Examples: ADJ4 VPO:−8.9E−6

Link: `VSIze: <NRx>`

Arguments: `<NRx>` can range between 1E −15 to 1E + 15.

`VSIze` sets the trace vertical graphical size only if the trace was created in floating-point mode.

Examples: `ADJ4 VSI:4.5E-2`

Query Notes: `ADJ<dui>?` returns its links and arguments in the following order:

```
ADJtrace<ui> PANzoom:<arg>, HMAg:<NR3>,
    HPOsition:<NR1>, VPOsition:<NR3>, VSIze:<NR3>
    TRSEP:<NR3>, GRLocation:<arg>, COLor:<NR1>
```

`ADJtrace?` returns the same information as `ADJtrace<ui>?` for all defined traces in low-to-high trace order.

ADJ? Predefined Responses — Several `ADJtrace` links can only be set under restricted conditions, but can be queried at any time. These links return the predefined values listed in Table 2-43 if queried while they cannot be set.

Table 2-43: ADJ? Predefined Responses

ADJ Link	Returned Value(s)
HMAg	−1.0E+0
HPOsition	1.0E+16
TRSep	1.0E+16
VPOsition	1.0E+16
VSIze	−1.0E+0

ALTinkjet *<link>:<arg>*

Group:	External I/O Commands

ALTinkjet specifies printing parameters for HP Thinkjet and LaserJet printers operating in HP graphics mode. This command is identical to the LSRJET command.

NOTE

ALTinkjet *does not support Thinkjet and LaserJet printers operating in Epson emulation mode.*

Link: DIRection: { HORiz | VERt }

DIRection selects the printing orientation. HORiz prints rows left-to-right and top-to-bottom. VERt prints columns bottom-to-top and left-to-right.

Examples: ALT DIR:HOR

Link: FORMat: { DIThered | DRAft | HIRes | REDuced | SCReen }

FORMat selects the printing format. DIThered modifies print contrast for the printer. DRAft prints selected fields in reverse video. REDuced is a quarter the size of DRAft but does not show intensified regions. SCReen is a one-to-one mapping of 3-bit pixel information.

NOTE

Due to graphics imaging constraints on standard LaserJet printers, HIRes *format may not generate a suitable copy on a standard LaserJet printer*

Examples: ALT FORM:DRA

Link: PORt: { CENTRonics | GPIb | RS232 }

PORt specifies the output port for the printer.

Examples: ALT POR:RS232

Link: ? COLor

Query Only. ? COLor returns the mapping of printer colors and the instrument color index.

Examples: `ALT? COL`
`ALTINKJET COLOR0:4095, COLOR1:0; COLOR2:3203, COLOR3:341`
`COLOR4:1046, COLOR5:637, COLOR6:1710, COLOR7:2323`

Link: `? COLORn`

Query Only. `? COLORn` returns the mapping of printer color n and the instrument color index.

Examples: `ALT? COLOR`
`ALTINKJET COLOR0:4095, COLOR1:0; COLOR2:3203, COLOR3:341;`
`COLOR4:1046, COLOR5:637, COLOR6:1710, COLOR7:2323`

Link: `COLor<ui>: <NRx>`

Arguments: `<ui>` can range between 0 and 7. `<NRx>` can range between 0 and 4095.

Set Only. `COLor<ui>` assigns colors to the instrument color index as shown in Table 2-44.

Table 2-44: Default ALTinkjet Color Assignments

Color Index	ALTinkjet Color	Color Index	ALTinkjet Color
0	4095 (0xFFF)	4	1046 (0x416)
1	0 (0x000)	5	637 (0x27D)
2	3203 (0xC83)	6	1710 (0x6AE)
3	341 (0x155)	7	2323 (0x913)

AMPlitude?

Group: Measurement Commands

Query Only. `AMPlitude?` returns the trace amplitude of the selected trace, expressed as the difference of Topline-Baseline. `AMPlitude?` is available only when `MMOde` is set to either `SW` or `STAT` on the selected trace.

Examples: **AMP?**
AMPLITUDE 3.2E-3, EQ

AUTOSet [*<link>***:**] *<arg>*

Group: Acquisition Commands

`AUTOSet` controls vertical, horizontal, and trigger automatic ranging and positioning of input signals on the selected trace for both acquired and stored traces. For acquired signals, the vertical size is set and the time base is adjusted. For stored traces, the display is scaled.

Link: HORiz: { ON | OFF }

`HORIZ` sets `AUTOSet` to `ON` or `OFF` for the horizontal parameters. See the `MODe` link.

Examples: AUTOS HOR:ON

Link: MASK: { AUTO | MANual }

`MASK:AUTO` causes an `AUTOSet STARt` to occur automatically when a `MASKDefine` command is executed (assuming that `DSYS ON` was previously selected). Also, at the end of the autoset, the following two additional commands are automatically performed:

MASKStat COUnt: ON

CONDacq TYPe: WAVfrm

These two commands are also performed if the `AUTOSet STARt` command is received (and if `MASK:AUTO` and `DSYS ON` were previously performed).

`MASK:MANual` prevents the `AUTOSet STARt` and the other two commands from occurring.

Examples: AUTOS MASK:MAN

Link: MODe: { EDGe | PERiod }

MODE selects mode for Main traces. EDGe centers the first transition and spreads the 20% to 80% transition region over 2 to 5 divisions. PERiod attempts to place 2 to 5 periods of the trace on the display.

Examples: AUTOS MOD:EDG

Link: SMOFfadj: { ON | OFF }

SMOFfadj when set ON, sets permission to override the standard amplitude for some of the standard masks, allowing for vertical offset adjustment.

Examples: AUTOS SMOF:ON

Link: STARt

Set Only. STARt begins autosetting of the selected trace.

Examples: AUTOS STAR

Link: TRIgger: { ON | OFF }

TRIgger sets autoranging to ON or OFF for the trigger signal.

Examples: AUTOS TRI:ON

Link: UNDO

Set Only. UNDO cancels a previous autoset and returns to the settings in effect before the last AUTOSET START.

Examples: AUTOS UNDO

Link: VERt: { ON | OFF }

VERT sets AUTOSet to ON or OFF for the vertical parameters.

Examples: AUTOS VER:ON

Query Note: AUTOSet? returns its links and arguments in the following order:

AUTOS HOR:*<arg>*, VER:*<arg>*, TRI:*<arg>*, MOD:*<arg>*

AVG { ON | OFF }

Group: Acquisition Commands

AVG sets averaging ON or OFF for the vertical expression component
(`<y exp>`) of the trace description of the selected trace.

- When `<y exp>` is *not* enclosed with ENV and AVG is set to ON, `<y exp>` is enclosed with AVG().

- When `<y exp>` is enclosed with ENV and AVG is set to ON, AVG() replaces ENV().

- When `<y exp>` is enclosed with AVG() and AVG is set to OFF, the enclosing AVG() is removed.

NOTE

You cannot set AVG OFF when `<y exp>` is not enclosed with AVG(). You cannot set AVG to ON if the selected trace is XY or has only stored and/or scalar components.

Table 2-45: Examples Using AVG

<y exp> Before	Command	*<y exp>* After
M2	AVG ON	AVG(M2)
M1	AVG OFF	−error−
ENV(M1−M2)	AVG ON	AVG(M1−M2)
AVG(M3)	AVG OFF	M3
AVG(M4)	AVG ON	AVG(AVG(M4))

Query Note: AVG? returns the state of averaging for the entire `<y exp>`. AVG ON means the entire `<y exp>` is enclosed by AVG. AVG OFF means the entire `<y exp>` is not enclosed, although an AVG function may be embedded within the description.

BCOrrection { ON | OFF }

Group: Miscellaneous/System Commands

BCOrrection sets baseline correction to ON or OFF for all acquired traces. When ON the mainframe attempts to keep the selected trace at the same vertical screen position even though the input signal changes. This is useful in TDR applications. ON happens automatically when GRAT YUNIT:RHO is done; OFF happens automatically when GRAT YUNIT:VOLT is done.

Examples: BCO ON

BITMap *<link>:<arg>*

Group: External I/O Commands

BITMap specifies printing parameters for screen captures, in which data from the front panel display is processed by an external computer. Screen capture data include a title block and a pixel block.

BITMAP Title Block — The title block contains three ASCII strings terminated by new line characters. The first string includes the mainframe name, time and date, and the serial number. The second string contains the number of pixels per raster line. The third string gives the number of raster lines.

When BITMap DATAFormat is set to BINary, the title block is terminated with an ASCII NULL character following the third new line character. When BITMap DATAFormat is set to BINHex, the title block is terminated with the third new line character.

BITMAP Pixel Block — The pixel block is a stream of data bytes. The DATACompress and DATAFormat links determine the format (data compression scheme).

Link: DATACompress: { ON | OFF }

DATACompress specifies the pixel block data compression mode. When OFF, each byte contains one 3-bit pixel value in the three least-significant bits. When ON, each byte contains two 3-bit pixel values, with the first pixel in the least-significant three bits (see Figure 2-3). Also, when DATACompress is set to ON, the two most-significant bits in the byte encode the data repetition pattern, which is discussed below.

Examples: BITM DATAC:ON

Pixel Block Data Byte — The Figure 2-3 shows the bits in a pixel block data byte.

7	6	5	4	3	2	1	0

Repetition encoding if DATAC:ON   *Pixel 2 data if DATAC:ON*   *Pixel 1 data DATAC:{ON|OFF}*
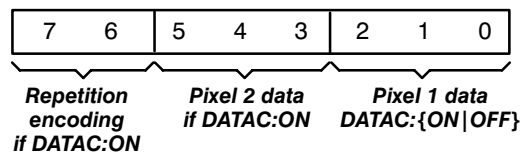
Figure 2-3: Bits in a Pixel Block Data Byte

Repetition Encoding — Table 2-46 lists the binary repetition encoding in bits 7 and 6 of the pixel data byte.

Table 2-46: Data Repetition Encoding

Bit 7	Bit 6	Meaning
0	0	Following byte(s) contain repetition count
0	1	Data pattern repeats once
1	0	Data pattern repeats twice
1	1	Data pattern repeats three times

When bits 7 and 6 encode the values 1 (01), 2 (10), or 3 (11), the pixel data is repeated one, two, or three times, respectively.

When bits 7 and 6 have the value 0 (00), the next one or two data bytes contain the repetition count. If the next byte has the decimal value 4 to 255, that is the pattern repetition count. If the next byte has the decimal value 1 to 3, these are the high-order bits of a 10-bit repetition count and the following byte contains the lower eight bits.

Link: `DATAFormat: { BINary | BINHex | TIFf }`

`DATAFormat` specifies the pixel block data format. `BINary` data are output in a stream without delimiters. `BINHex` data are output as ASCII hexidecimal bytes and each raster line is terminated with a new line character.

`TIFf` specifies that the bitmap format is standard color TIFF graphics format. The `TIFf` format specifies version 6.0 of the TIFF specification. `SCReen` and `DIThered` formats generate Palette Color (Class P) prints; `HIRes`, `DRAft`, and `REDuced` formats create Bilevel (Class B) prints. TIFF Compacted generates a TIFF file using compression mode 32773 (Packbits).

NOTE

Due to changes between version 5.0 and version 6.0 of the TIFF Specification, some application programs might not accept compacted Class P images.

Examples: `BITM DATAF:BIN`

Link: `DIRection: { HORiz | VERt }`

`DIRection` selects the printing orientation. `HORiz` prints rows left to right and from top to bottom. `VERt` prints columns bottom to top and from left to right.

Examples: BITM DIR:HOR

Link: FORMat: { DIThered | DRAft | HIRes | REDuced | SCReen }

FORMat selects print formatting. DIThered reduces saturation for icon and text backgrounds to improve print contrast. HIRes dithers icon and text backgrounds and increases foreground saturation to improve contrast for monochrome printers with limited gray-scale capability. DRAft prints black-on-white background except for selected icons or text which are printed white-on-black background.

REDuced prints black-on-white background only. SCReen is a one-to-one mapping of 3-bit pixel information.

Examples: BITM FORM:DIT

Link: PORt: { CENTRonics | GPIb | RS232 }

PORt specifies the output port for the printer.

Examples: BITM POR:GPI

BYT.or { LSB | MSB }

Group: Data Transfer Commands

BYT.or selects whether the least significant byte (LSB) or most significant byte (MSB) of binary trace data (<*bblock*>) is transmitted first during a data transfer. BYT.or sets the byte order for CURVe, HISTOgram? DATA, DISPlay? DATA, and VPCurve data transfers. BYT.or affects the same data transfer commands affected by the data encoding command ENCdG. Power-on default is MSB; LSB has a faster data transfer rate. Correct byte order depends on the controller.

Examples: BYT. LSB

CALibrate *<alpha><ui> <link>*: *<arg>*

Group: Calibration Commands

NOTE

Possible alpha values for the 11801C are A through D, for SM-11 Multi-channel Units, and M for the mainframe. The Alpha value for the CSA 803C is always M, for mainframe.

Performs manual or automatic calibration of the specified sampling head channel.

Link: `AUTO:` { `BLOwby` | `DADj` | `LOOpgain` | `OFFSet` | `ONUll` | `TDRAmplitude` }

Set Only. `AUTO` performs an automated calibration for the specified parameter on the specified channel. Each of these arguments is discussed as a link for the `CALibrate` command.

`AUTO:BLOwby` is not currently used by any sampling head.

`AUTO:OFFSet` is available only if the installed sampling head has probe-tip offset capability (for example, the SD-14 sampling head has probe tip offset capability).

Examples: `CALM4 AUTO:DAD`

Link: `BLOwby:` *<NRx>*

Arguments: The range for *<NRx>* is specified by the sampling head.

`BLOwby` sets blowby compensation.

Examples: `CALM4 BLO:1.43E-3`

Link: CSTore: { ALL | BLOwby | LOOpgain | OFFSet | ONUll | TDRAmplitude }

Set Only. CSTore stores the specified parameter (or ALL) as a user setting in sampling head EEPROM. TDRAmplitude values are stored for both positive and negative polarity. ONUll (offset null) values are stored for both states of smoothing (on and off). The DADj value is not stored.

CSTore:BLOwby is not currently used by any sampling head.

OFFSet stores the user offset calibration values in the sampling head EEPROM. CSTORE:OFFSET is available only if the installed sampling head has probe-tip offset capability (for example, the SD-14 sampling head has probe tip offset capability).

ONull produces two values for offset null on the specified channel (one for SMOoth ON and one for SMOoth OFF. The best value for ONull depends upon the sampling rate (up to 200K for external trigger source; up to 100K for internal clock). Therefore, perform this CALibrate command with your external trigger source attached, if that is the type of trigger you will be using.

TDRAmplitude produces two values for the specified channel (one for + (positive) polarity and one for − (negative) polarity).

Examples: CALM4 CST:LOO

Link: DADj: *<NRx>*

Arguments: *<NRx>* can range from 0 to 100.

DADj sets the target measurement for delay adjustment. If you adjust either channel in a two-channel head, both channels are set.

Examples: CALM2 DAD:50

Link: DREcall: { ALL | BLOwby | LOOpgain | OFFSet | ONUll | TDRAmplitude }

Set Only. DREcall recalls from the sampling head EEPROM the factory default setting for the specified parameter.

DREcall:BLOwby is not currently used by any sampling head.

OFFSet recalls the factory default offset calibration values from the sampling head. DRECALL:OFFSET is available only if the installed sampling head has probe-tip offset capability (for example, the SD-14 sampling head has probe tip offset capability).

Examples: CALM1 DRE:ALL

Link: `LOOPgain: <NRx>`

Arguments: The range for `<NRx>` is specified by the sampling head.

`LOOpgain` sets the calibration loop gain.

Examples: `CALM3 LOO:1.2`

Link: `ONUll: <NRx>`

Arguments: The range for `<NRx>` is specified by the sampling head.

`ONUll` sets offset nulling for the current state of smoothing for a channel. The best value for `ONull` depends upon the sampling rate (up to 200K for external trigger source; up to 100K for internal clock). Therefore, perform this `CALibrate` command with your external trigger source attached, if that is the type of trigger you will be using.

Examples: `CALM7 ONU:O.23`

Link: `TDRAmplitude: <NRx>`

Arguments: The range for `<NRx>` is specified by the sampling head.

`TDRAmplitude` sets the TDR amplitude for the current state of polarity of the specified channel.

Examples: `CALM7 TDRA:3.5E-4`

Link: `URecall: { ALL | BLOwby | LOOpgain | OFFSet | ONULl | TDRAmplitude }`

Set Only. `URecall` recalls user settings from the sampling head EEPROM that were saved with `CSTore` link.

`URecall:BLOwby` is not currently used by any sampling head.

`OFFSet` recalls the user offset calibration values from the sampling head. `URECALL:OFFSET` is available only if the installed sampling head has probe-tip offset capability (for example, the SD-14 sampling head has probe tip offset capability).

Examples: `CALM32 URE:ONU`

Query Note: `CAL<alpha><ui>?` returns its links and arguments in the following order:

```
CALIBRATE<alpha><ui> BLOWBY:<NR3>, LOOPGAIN:<NR3>,
    ONULL:<NR3>, TDRAMPLITUDE:<NR3>, DADJ:<NR1>
```

CH *<alpha><ui><link>***:***<arg>*

Group: Channel Commands

NOTE

Possible alpha values for the 11801C are A through D, for SM-11 Multi-channel Units, and M for mainframe. The Alpha value for the CSA 803C is always M, for mainframe.

Sets and queries the vertical parameters of channels. The *<alpha>* component will always be M and the *<ui>* component will be a channel number.

Link: ? ATTenuation: { X1 | X2 | X5 | X10 }

Query Only. ? ATTenuation returns the combined probe and head attenuation factor.

Examples: **CHM6? ATT**
CHM6 ATTENUATION:X1

Link: CDElay: *<NRx>*

Arguments: The range for *<NRx>* is specified by the sampling head.

CDElay sets the delay between channels for dual-channel sampling heads with channel delay but without TDR capability (for example, the SD-14 sampling head).

Examples: CHM6 CDE:10

Link: DATAType: { REPetitive | RANdom }

In the sampling head DATAType sets the sampling loop to either RANdom, for random data, or REPetitive, for high-precision device characterization. If DATAType is set to RANdom for either channel in the sampling head, the instrument turns smoothing off and displays a warning. However, if DATAType is RANdom and smoothing is turned on, the instrument displays the following error message: Error 2003, Smoothing is not permitted when either channel in a head is in Random Data mode.

DATAType is currently not supported by any sampling head.

Examples: CHM6 DATAT:REP

Link: `EXTAttenuation:` *`<NRx>`*

Arguments: The range for *`<NRx>`* is 1E−6 to 1E+6. A value of 1 equals no attenuation and no amplification; a value less than 1 denotes amplification; a value greater than 1 denotes attenuation.

`EXTAttenuation` sets channel external attenuation to a user-specified value (for example, `100` sets channel external attenuation to x100 external attenuation).

Examples: `CHM6 EXTA:100`

Link: `OFFSet:` *`<NRx>`*

Arguments: The range for *`<NRx>`* is usually −2 V to +2 V, but can be larger for some sampling heads.

`OFFSet` sets the input vertical offset for channel with acquisition capability.

Examples: `CHM5 OFFS:-0.9`

Link: `SENSitivity:` *`<NRx>`*

Arguments: The range for *`<NRx>`* is usually 2.0E−3 to 2.55E−1 volts, but can be larger for some sampling heads.

`SENSitivity` sets the input vertical sensitivity (gain) for a channel with acquisition capability.

Examples: `CHM3 SENS:1.5E-3`

Link: `SMOothing:` `{ ON | OFF }`

`SMOothing` sets smoothing to `ON` or `OFF` for a head with smoothing capability.

Examples: `CHM4 SMO:ON`

Link: `TDRDelay:` *`<NRx>`*

Arguments: The range for *`<NRx>`* is specified by the sampling head.

`TDRDelay` sets the relative delay of the output pulse between the channels of a dual-channel head with TDR capability.

Examples: `CHM4 TDRD:3.1E-6`

Link: `TDRPolarity: { PLUs | MINUs }`

         `TDRPolarity` selects positive- or negative-going TDR output pulses for channels with switchable-polarity TDR.

Examples: `CHM7 TDRP:PLU`

Link: `TDRState: { ON | OFF }`

         `TDRState` sets the TDR generator to `ON` or `OFF` for channels with TDR capability.

Examples: `CHB12 TDRS:ON`

Link: `TOFfset: <NRx>`

Arguments: The range for `<NRx>` is specified by the sampling head.

         `TOFfset` sets the termination offset voltage for a channel with termination offset capability.

Examples: `CHM4 TOF:1.0E-3`

         Query Notes: `CH<alpha><ui>?` returns the links and arguments for the specified channel in the following order:

```
CH<alpha><ui> EXTATTENUATION:<NRx>, OFFSET:<NRx>,
    SENSITIVITY:<NRx>, SMOOTHING:<arg>, CDELAY:<NRx>,
    TDRDELAY:<NRx>, TDRPOLARITY:<arg>, TDRSTATE:<arg>,
    ATTENUATION:<arg>
```

         Some links and arguments may not be returned depending on which sampling head is used. For example, an SD-26 will not return the `TDRDELAY` link.

         `CH<alpha>?` returns the same information as `CH<alpha><ui>?`, for all installed channels of the specified unit, in low-to-high numeric order. Channels without heads installed are not included in the response.

         `CH?` returns the same as `CH<alpha><ui>?`, for all installed channels. The response is in low-to-high numeric order for M1 to M8, followed by units A through D, if installed. Channels or multi-channel units that are not installed are not included in the `CH?` response.

CHIdent { FLAsh | QUIet }

Group: Miscellaneous/System Commands

CHIdent controls the function of the channel indicator light on the front panel of the sampling heads. If CHIdent is set to FLAsh, the channel indicator light flashes to identify the selected channel. If CHIdent is set to QUIet, the channel indicator light for the selected channel does not flash.

Examples: CHI FLA

CLEar { ALLTrace | *<qstring>* | TRAce*<ui>* }

Group: Trace and Settings Commands

Set Only. CLEar discards acquired data for all displayed traces, the specified labeled trace, or for the specified trace. (See the REMove command.)

Link: { ALLTrace | *<qstring>* | TRAce*<ui>* }

Arguments: *<qstring>* can be a valid *<qstring>*. *<ui>* can range between 1 and 8.

No error is reported for sending CLEAR ALLTRACE when no traces are defined. Wildcard characters are valid with *<qstring>*. (Refer to Label Wildcard Characters on page 2-149 for wildcard definitions.)

Examples: CLE TRA5

COLor *<ui>* *<link>***:***<arg>*

Group: Display and Color Commands

COLor*<ui>* controls the front panel colors. The *<ui>* range is 0 to 7 and specifies the color index as listed in Table 2-47.

Table 2-47: Color Indexes

<ui>	**Color Specified**	*<ui>*	**Color Specified**
0	Background	4	Trace Color 4
1	Trace Color 1	5	Window traces
2	Trace Color 2	6	Graticule and Selectors
3	Trace Color 3	7	Cursors and Measurement Annotation

NOTE

Refer to the Tektronix Color Standard HLS coordinate system for the definitions of hue, saturation, and lightness.

Link: DEFAult

DEFAult sets the factory default hue, lightness, and saturation for the specified color.

Examples: COL1 DEFA

Link: HUE: *<NRx>*

Arguments: The range for *<NRx>* is 0 to 360 degrees.

HUE sets the hue of the specified color.

Examples: COL4 HUE:120

Link: LIGhtness: *<NRx>*

Arguments: The range for *<NRx>* is 0 to 100 percent.

LIGhtness sets the lightness of the specified color.

Examples: COL4 LIG:30

Link:	SATuration: *<NRx>*

Arguments:	The range for *<NRx>* is 0 to 100 percent.
	SATuration selects the saturation of the specified color.

Examples:	COL4 SAT:80

COLor DEFAult

Group:	Display and Color Commands
	COLOR DEFAULT sets all colors in the display to their factory-default values.

Examples:	COL DEFA

COMpare { ON | OFF }

Group:	Measurement Commands
	COMpare controls the measurement comparison mode. When COMpare is set to OFF, a measurement query returns the value of the measurement followed by an accuracy qualifier. COMpare OFF is the normal measurement mode. When COMpare is set to ON, a measurement query compares the measurement value with a reference value set with the REFSET command, and then returns the difference with an accuracy qualifier. If the reference measurement is undefined or the measurement qualifier is UN (uncertain), the returned comparison qualifier is also UN.

NOTE

For the list of measurement accuracy qualifiers and their definitions, refer to page 2-168.

Examples:	COM ON

CONDacq *<link>:<arg>*

Group:	Acquisition Commands

CONDacq sets the following conditions for trace acquisition: completion of a specified condition, continuous acquisition, or acquisition on a complete trace record.

Completion of any conditional acquisition TYPe (i.e., all types except CONTInuous) is signaled by event code 450, Conditional acquire complete.

Link: ? REMAining *<NR1>*

Query Only. ? REMAining returns a value indicating how much of the selected acquisition TYPe must still be acquired to complete acquisition. Refer to Table 2-48.

Table 2-48: REMAINING Means for CONDacq TYPE

TYPe	Meaning
AVG	Number of averages remaining
AVG.env	Number of averages and envelopes remaining
CONTInuous	Not meaningful; always returns 0
ENV	Number of envelopes remaining
GRADed	Number of points remaining
HIST.pt	Number of points remaining
RECOrd	Not meaningful; always returns 0
WAVfrm	Number of complete trace records remaining

NOTE

When conditional acquisition is complete and acquisition has stopped, the ? REMAining *query always returns* 0 *(zero).*

Examples: **COND? REMA**
CONDACQ REMAINING:22

Link: TYPe: { AVG | AVG.env | BOTh | CONTInuous | ENV |
GRADed | HIST.pt | MASK | MASK*<ui>* | RECOrd | WAVfrm }

TYPe selects the acquisition type, as shown in Table 2-49.

Table 2-49: Acquisition Types

TYPE	Meaning
AVG	Acquires NAVg number of averages for all traces that include AVG in their description.
AVG.env or BOTh	Acquires NAVg number of averages or NENV number of envelopes for all traces that include either AVG or ENV or both in their description.
CONTInuous	Acquires continuously until halted with ACQuisition STOP.
ENV	Acquires NENV number of envelopes for all traces that include ENV in their description.
GRADed	Acquires until NGRAded points reach their maximum (65535) for all acquired traces when DISPLAY TYPE is GRADed.
HIST.pt	Acquires until NHIST.pt points are in the histogram for the selected trace.
MASK*<ui>*	**Set only.** Sets the conditional acquire type to stop when at least NMAsk hits are acquired in MASK*<ui>*. The query MASK*<ui>*? NCOunt returns the hit count for the mask.
MASK	**Set only.** Sets the conditional acquire type to stop when at least NMAsk hits are acquired in all masks combined. The query MASKST? TOTAL returns the combined hit count for all masks.
RECOrd	Acquires a trace until its trace record is filled.
WAVfrm	Acquires NWAVfrm number of complete trace records for the selected trace.

NOTE

For all acquisition types, selecting CONDacq TYPe *immediately begins conditional acquisition. Set the display or acquisition mode on and set its parameters, such as* NAVg *for* AVG, *before starting conditional acquisition with a* CONDacq TYPe *command.*

Examples: COND TYP:ENV

Link: WAIT

Set Only. Causes the instrument to stop accepting and processing commands to the ASCII interfaces until the current conditional acquisition is complete.

Note: When used with GPIB, the controller's timeout must be set to infinite, or the command could time out and produce a GPIB error.

Examples: COND WAIT

COPy [*<link>***:**] *<arg>*

Group: External I/O Commands

COPy sends a copy of the front panel display to the port specified in the appropriate printer command.

Link: ABOrt

Set Only. ABOrt terminates the hardcopy output in process and clears the queue of copy requests.

Examples: COP ABO

Link: FORMat: { DIThered | DRAft | HIRes | REDuced | SCReen }

Set Only. COPY FORMat selects the output format for the currently selected printer.

DIThered improves print contrast by reducing saturation for icon and text backgrounds. HIRes improves contrast for monochrome printers with limited gray-scale capability by dithering icon and text backgrounds and increasing saturation of the foregrounds. DRAft prints black-on-white background except for selected icons or text, which are printed white-on-black background. REDuced is a quarter-size version of DRAft and prints black-on-white background only. SCReen is a one-to-one mapping of 3-bit pixel information. (Refer to the BITMap command.)

NOTE

The COPY FORMat *link is included for backward compatibility. For new applications, use the* FORMat *link of the appropriate printer command.*

Examples: COP FORM:HIR

Link: KILl

Set Only. KILl terminates the hardcopy in progress and clears the queue of all copy requests.

Examples: COP KIL

Link: PRINter: { ALTinkjet | BITMap | DSKJETC | HP560C | HPGl | LSRJET | PIN8 | PIN24 }

PRINter selects the target printer. Refer to the individual printer commands to select the printer parameters.

Examples: COP PRIN:DSKJETC

Link: STARt

Set Only. COPY START initiates a front panel copy, spooling the data into memory even if another copy request is printing or spooling.

NOTE

If you enter COPy *with no argument when no other copy request is printing or spooling, a copy is started. However, if a copy request is spooling, entering* COPy *aborts the spooling copy and does not initiate a copy.*

Examples: COP STAR

Link: ? STATUs: { IDLe | ABORTING | PRINting | SPOoling }

Query Only. ? STATUs returns the printing status of front-panel copies. IDLe means no copies are printing or spooling; ABORTING, PRINting, and SPOoling are self-explanatory.

Examples: **COP? STATU**
COPY STATUS:IDLE

CROss?

Group: Measurement Commands

Query Only. CROss? returns the time for the selected trace from the trigger point to a specified reference level crossing, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.) The reference level is set with the MPAram REFLEvel command. The crossing slope is set with the MPAram MSLOpe command.

Examples: CRO?
CROSS 6.9284065E-8,EQ

CURSor *<link>:<arg>*

Group: Cursor Commands

CURSor sets cursor operating characteristics for the selected trace, such as the cursor type (dot or bar), the reference cursor, and whether front panel readouts are displayed.

Link: ? MODE: { ABSolute | RELative }

Query only. ? MODE indicates whether horizontal readout is relative to the trigger point (ABSolute) or relative to a chosen horizontal value (RELative). Command SETZero sets the reference value.

Examples: CURS MODE:ABS

Link: REAdout: { ON | OFF }

REAdout controls whether front panel cursors and their corresponding knob readouts are displayed and active from the front panel. When REAdout is set to OFF, the cursors and their values in the Cursors menu are not displayed. However, cursors can be set or queried with remote commands regardless of the REAdout setting.

Examples: CURS REA:ON

Link: REFErence: TRAce*<ui>*

Arguments: *<ui>* can range between 1 and 8.

REFErence selects the reference trace for split cursors. When the specified REFErence trace is not the selected trace, the CURSOR TYPE is automatically set to SPLit. When the CURSOR TYPE is set to PAIred, the REFErence trace is set to the selected trace. The default REFErence for a newly-created trace is itself.

XY Note: You cannot change the REFErence trace to an XY trace.

Examples: CURS REFE:TRA2

NOTE

It is not an error if you specify a REFErence *trace that is not yet defined. The* REFErence *trace is only checked when* CURSOR READOUT *is set* ON *or at a* DOT2Abs? *query. If the* REFErence *trace is then undefined, it is changed to the selected trace.*

Examples: CURS REFE:TRA5

Link: SETZero: { CUR1 | CLEar }

Set only. Specifies the horizontal reference point used. CUR1 sets the reference value to the current position of Cursor 1 (CURS? Returns REL). CLEar sets the reference value to the trigger point (CURS? Returns ABS).

Examples: CURS SETZ:CUR1

Link: TYPe: { PAIred | SPLit | VBArs | HBArs }

TYPe selects the cursor type. Setting the TYPe to PAIred automatically sets the REFErence trace to the selected trace.

XY Note: SPLit cursors are not permitted on XY traces.

Examples: CURS TYP:VBA

Link: ? XUNit { DIVS | FEEt | INChes | METers | SEConds | VOLts }

Query Only. ? XUNit returns the horizontal units of the selected trace.

Examples: **CURS? XUN**
CURSOR XUNIT:SECONDS

Link: `? YUNit: { DIVS | RHO | VOLts }`

Query Only. `? YUNit` returns the vertical units of the selected trace.

Examples: **CURS? YUN**
CURSOR YUNIT:VOLTS

Link: `? ZEROPoint: <NRx>`

Query Only. `? ZEROPoint` returns the position of the horizontal reference point on the selected trace.

Examples: **CURS? ZEROP**
CURS ZEROPOINT:5.260805E-6

Query Note: `CURS?` returns its links and arguments in the following order:

```
CURS REA:<arg>, REFE:<arg>, TYP:<arg>, MODE:<arg>,
     ZEROPOINT:<arg>, XUN:<arg>, YUN:<arg>
```

CURVe [CRVId:*<arg>*,] *<curve data>*

Group: Data Transfer Commands

`CURVe` transfers unscaled trace data to and from the controller in binary or ASCII format. Each trace that is transferred has an associated trace preamble that contains information such as scaling factors and the number of data points transferred. Refer to the `WFMPre` command for the trace preamble.

The *query* form retrieves data *from* the instrument. The data source is specified by the `OUTput` command. The entire `CURVe?` response can be sent back to the instrument as a set command.

The *set* form sends data *to* the instrument from the controller. An incoming trace is always stored; it is never active or acquired. The `STO` (store) location for the data is specified by the `INPut` command. The power-on default `INPut` location is `STO1`.

Link: `[ CRVid: { STO | TRAce } <ui> ] | <curvedata>`

`CRVid` link is generated by a `CURVe?` query to identify the data source; it is ignored in the set form.

<Curve data> can be in ASCII (*<asc curve>*) or binary (*<bblock>*) format. The format is set by the `ENCdg WAVFRM` command.

Trace Header — Trace record data sent in ASCII or binary formats are prefixed with the same ASCII header:

> CURVE CRVID:{ STO | TRACE } *<ui>,(trace data)*

When the `LONgform` command is set `OFF`, the ASCII header is shortened to `CURV CRVI:{ STO | TRA } <ui>`.

The following example is an excerpt from an ASCII-formatted data transfer. (The shortest data transfer contains 512 points.)

Examples: `CURV?`
```
CURVE CRVID:TRA2,4022,3130,2756,1297,709,1073,822,685,111
2,777,1666,2249,3615,4180,4231,4113,988,-2241,-5609,-128,
-3076,-9924,-8434,-8112, ...
```

ASCII Transfer — Data transferred as an `<asc curve>` use the following format:

> `<asc curve> ::= <NR1> [ ,<NR1> ] ...`

where `<NR1>` values are data points within the range -32768 to $+32767$.

For most YT traces, each `<NR1>` value represents one data point in the trace record. For enveloped YT traces, every two `<NR1>` values represent one min/max pair in the trace record. For XY traces, every two consecutive `<NR1>` values represents one X,Y coordinate pair in the trace record. (The X-coordinate is the first point in the pair.) The command `WFMPRE? PT.FMT` indicates which data format will be used.

Binary Transfer — Data transferred as a binary block (`<bblock> [ ,<bblock> ]`) use the format:

> `<bblock>::=%<byte cnt><bin pt>...<checksum>`

where `<byte cnt>` is a two-byte binary integer (MSB first) giving the length in bytes of the remainder of the binary block, including checksum; `<bin pt>` is a two-byte binary data point in the range -32768 to $+32767$; `<checksum>` is an 8-bit, twos complement of the modulo 256 sum of `<byte cnt>` and all `<bin pt>` data.

The transmission order for data points is set by the `BYT.or` command. There are no separators (such as commas) between data points.

Figure 2-4 illustrates binary data transfer.

Predefined CURVe? Data Values — The data point values shown in Table 2-50 are predefined for CURVe?:

Table 2-50: Predefined CURVe? Data Values

Data Value	Meaning
+32767	Vertical Overrange. Data point is high off-screen and cannot be displayed with current scaling parameters.
−32767	Vertical Underrange. Data point is low off-screen and cannot be displayed with current scaling parameters.
−32768	Null Data. Data point that has not been acquired.

Data Header

Setting LONGFORM to OFF will shorten this ASCII header information.

CURVE
<space>
CRVID:
{**STO**|**TRACE**}
<ui>,

Data

Byte count → % → 1
<MSB> → 2
<LSB> → 3
Assumes BYT.OR is MSB → *<MSB>* → 4
<LSB> → 5
Data point #1
<checksum> → 2048
EOI

If LF is used as the message terminator, it precedes the EOI signal

1024 pt. YT waveform
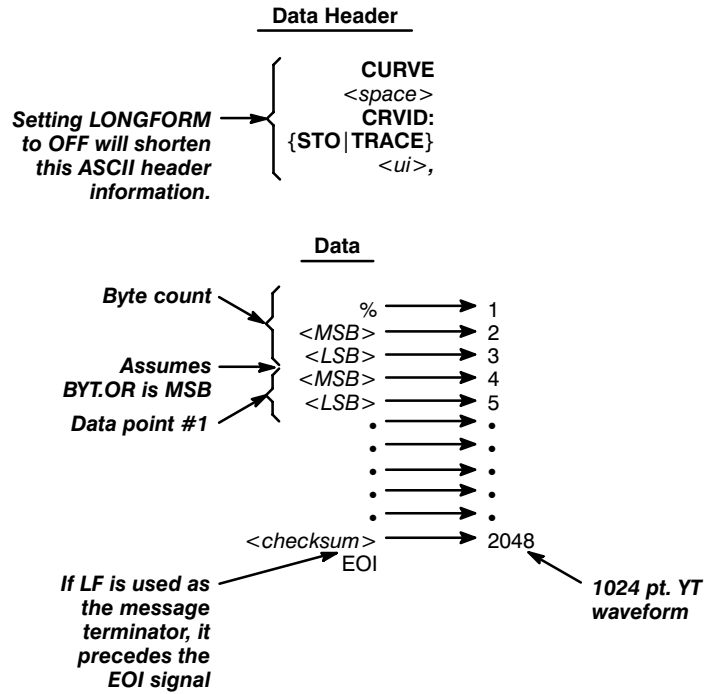
Figure 2-4: Binary Data Transfer

Trace Scaling — CURVe transfers unscaled trace data which must be scaled in order to be analyzed. The following formulas use values from the trace preamble (see the WFMPre command) to scale the coordinate values of each point transferred.

There are two scaling formulas for **YT** traces:

$$X_n = XZERO + XINCR * n$$
$$Y_n = YZERO + YMULT * data\_pt\_n$$

where X_n is the scaled horizontal coordinate of the nth data point in XUNits; Y_n is the scaled vertical coordinate of the nth data point in YUNits; XZEro, XINcr, YZEro, and YMUlt are values from the WFMPre command; n is the sequence number of the nth retrieved data point (range is 0 to WFMPRE NR.PT – 1); *data_pt_n* is the value of the nth unscaled point (as retrieved by CURVe?).

There are two scaling formulas for **XY** traces:

$$X_n = XZERO + XMULT * data\_pt\_nx$$
$$Y_n = YZERO + YMULT * data\_pt\_ny$$

where X_n is the scaled X-coordinate of the nth unscaled X,Y pair in XUNits; Y_n is the scaled Y-coordinate of the nth unscaled X,Y pair in YUNits; XZEro, XMUlt, YZEro, and YMUlt are values from the WFMPre command; *data_pt_nx* is the value of the nth unscaled X-coordinate (as retrieved by CURVe?); *data_pt_ny* is the value of the nth unscaled Y-coordinate.

Sending a Trace Without a Preamble — It is possible to send a trace to the instrument without supplying a preamble. If a stored trace exists at the INPUT STO location, it is overwritten and its preamble is used with the new trace. If no stored trace exists at the INPUT STO location, the default preambles listed in Table 2-51 are used with the new trace.

Table 2-51: Default Preamble Parameters

<link>: <arg>		*<link>: <arg>*	
NR.pt:	512	YUNit:	−current−
PT.fmt:	Y	YZEro:	0.0
XINcr:	5.0E−7	LABel:	" " (null)
XZEro:	0.0	TIMe:	−current−
YMUlt:	1.5625E−4	DATe:	−current−

These are the power-on default values. When any of these links are modified (set) with the WFMPre command, the new values are used.

DAFiltering *<alpha><ui>*:*<NR1>*

Group: Calibration Commands

NOTE

Possible Alpha values for the 11801C are A through D, for SM-11 Multi-channel Units, and M for the mainframe. The Alpha value for the CSA 803C is always M.

`DAFiltering` (Delay Adjust Filtering) sets the hardware filtering constant used by `CALibrate DADj` for the specified sampling head.

Link: `<alpha><ui>: <NR1>`

Arguments: The range for `<NR1>` is 1 to 7.

Examples: `DAF M7:2`

DAMeasref *<alpha><ui>*

Group: Calibration Commands

NOTE

Possible Alpha values for the 11801C oscilloscope are A through D. The Alpha value for the instrument is always M.

Set Only. `DAMeasref` (Delay Adjust Measurement Reference) sets the inter-head delay to mid-range for the specified channel.

NOTE

Channel must be connected to the calibrator output before issuing the `DAMeasref` command.

Examples: `DAM M5`

DATE *<qstring>*

Group: Miscellaneous/System Commands

DATE sets or queries the date on the internal calendar.

Link: *<qstring>* "*<dd>-<mon>-<yy>*"

Arguments: <qstring> is of the form "*<dd>-<mon>-<yy>*" where *<dd>* is the day of the month, *<mon>* is the first three letters of the month, and *<yy>* is the last two digits of the year.

Examples: DATE "18-JUL-93"

DCOmp { ON | OFF }

Group: Calibration Commands

Sets continuous strobe delay calibration to ON or OFF. When ON, the instrument continuously measures the inter-head delay and adjusts hardware to achieve target value set by CALIBRATE DADJ and DAMeasref.

Examples: DCO OFF

DEBug *<link>:<arg>*

Group: External I/O Commands

DEBug copies input data from the specified interface to the front panel display for program development troubleshooting. The incoming ASCII commands are displayed on the top four lines of the screen.

NOTE

Setting DEBug to ON for either interface slows system throughput considerably.

Link: GPIb: { ON | OFF }

GPIb sets DEBug to ON or OFF for the GPIB interface.

Examples: DEB GPI:OFF

Link: `RS232: { ON | OFF }`

`RS232` sets `DEBug` to `ON` or `OFF` for the RS-232-C interface.

Examples: `DEB RS232:ON`

DEFine *<qstring>,<qstring>* [**?** [*<qstring>*]

Group: Miscellaneous/System Commands

`DEFine` defines a logical name to substitute for any instrument command string.

Link: `<qstring>,<qstring>`

Arguments: The first `<qstring>` is the logical name; the second `<qstring>` is the expansion command string that is executed.

Examples: `DEF 'TB?','TBM?;TBW?'`

Once the logical name has been defined with `DEFine`, you enter the logical name without quotes the same as any other command.

Examples:
```
TB?
TBMAIN TIME:5.0E–3, LENGTH:1024,
XINCR:1.0E–10; TBWIN TIME:1.0E–3,
LENGTH:512, XINCR:5.0E–10
```

DEF Usage — Here are some rules and suggestions for using `DEFine`:

■ The first character of the logical name must be alphabetic. Case is ignored.

■ You cannot use logical names in `<qstring>` input; they may be interpreted as commands.

■ You cannot have an expansion string that is null (i.e., `''`). Also, the first character of an expansion string cannot be any of the six characters shown in Table 2-52.

Table 2-52: Restricted Expansion String Characters

Character	Character
colon (:)	space (octal 40)
comma (,)	linefeed (octal 12)
semicolon(;)	carriage return (octal 15)

■ You can define a short name for a group of concatenated commands, or you can rename a command to one or two letters. However, do not redefine the character **M**. This character represents the mainframe in various commands. If this letter is redefined, the commands that contain it will always return a syntax error. Be careful when redefining the reserved words listed in Appendix B.

■ Recursive DEFine logical names are acceptable only when recursion occurs to the right of an unquoted semicolon. All other recursive definitions are illegal.

Table 2-53: Acceptable and Illegal Recursion

Acceptable Recursion	Illegal Recursion
DEF 'z','tbmain?;z'	DEF 'z','z?'
DEF 'j','abstouch 3,10;j'	DEF 'j','text j'

NOTE

A valid recursive logical name causes an infinite command processing loop. Thus, once a recursive logical name is transmitted, the instrument will not respond to command input until a DCL (Device Clear) signal is sent to the port that received the recursive logical name. (Refer also to the FEOi command.)

Logical names and expansion strings are not stored in nonvolatile RAM. Therefore they are lost when the instrument is powered off.

Predefined Logical Names — Each time the instrument is turned on, the two logical names given in Table 2-54 are automatically placed in the definition table.

Table 2-54: Predefined Logical Names

Logical Name	Expansion String
e	RS232 ECHO:ON
v	RS232 VERBOSE:ON

DELete [*<link>***:**] *<arg>*

        Group: Trace and Settings Commands

                Set Only. `DELete` removes stored front panel setting(s) or stored traces from memory.

NOTE

              You cannot delete a stored trace that is a combined component of an active trace. (However, you can delete a stored trace if it is the only component of an active trace.)

        Link: { `ALLFps` | `ALLSTD` | `ALLSTO` }

                Set Only. `ALLFps` deletes all stored front panel settings. `ALLSTO` deletes all stored trace. It is not an error if `DEL ALLFps` or `DEL ALLSTO` is issued when no settings or traces are stored.

                `ALLSTD` *(11801C only)* deletes all saved trace descriptions. It is not an error to issue `DEL ALLSTD` when there are no saved trace descriptions.

      Examples: `DEL ALLSTO`

        Link: `FPS<ui>`

   Arguments: The range for `<ui>` is 1 to 9999.

                Set Only. `FPS<ui>` deletes the specified front panel setting.

      Examples: `DEL FPS2`

        Link: `<qstring>`

                Set Only. `<qstring>` deletes the stored trace or front panel setting that matches the label. Wildcard characters are interpreted; refer to page 2-149 for wildcard definitions. If the label matches both a stored trace and a front panel setting, the stored trace is deleted. To delete the labeled front panel setting, you must send `DEL <qstring>` again. (You cannot delete only the labeled FPS label when both the stored trace label and FPS match.)

      Examples: `DEL "TRIGGER39"`

        Link: `STD<ui>`

Arguments: The range for *<ui>* is 9 to MAXTranum.

Set Only. STD*<ui>* deletes the specified saved trace description. *(11801C only.)*

Link: STO*<ui>*

Arguments: The range for *<ui>* is 1 to 9999.

Set Only. STO*<ui>* deletes the specified stored trace.

Examples: DEL STO150

DIАg?

Group: Diagnostics Commands

Query Only. DIAg? returns pass/fail information from Self-tests Diagnostics or Extended Diagnostics. Power-on Diagnostics are always performed unless bypassed with hardware jumpers. DIAg? returns pass/fail/bypassed information and a list of the tests which were not performed.

The passing DIAg? response is:

DIAg PASsed:"{ NONe | *<omitted test>* }"

where:

NONe   means no tests were omitted

<omitted test> is a comma-delimited list of tests that were not per-
formed because of missing (optional) hardware

Examples: **DIA?**
DIAG PASSED:"NONE"

The failing DIAg? response is:

DIAg FAIled:"{ *<failed test>* | *<omitted test>* }"

where:

<failed test>   is a comma-delimited list of tests that failed diagnostics

Examples: **DIA?**
DIAG FAILED:"D1211,,a????"

NOTE

The DIAg? FAILED *response can include both failed and omitted tests. In the preceding example, D1211 is a failed tests and a???? is an omitted test.*

Refer to the *Service Manual* for your instrument for information on the syntax and meaning of omitted tests and failed tests.

The bypassed test DIAg? response is:

DIAg BYPassed

where:

BYPassed   means Self-tests Diagnostics were bypassed with hard-
ware jumpers at power-on

Examples: **DIA?**
DIAG BYPASSED

DISPlay *<link>:<arg>*

Group:	Display and Color Commands

`DISPlay` sets a variety of display options including persistence and the number of graticules.

Link:	`CLEar`

Set Only. `CLEar` removes all trace and histogram data from the display and restarts all acquisitions. `CLEar` is equivalent to the `CLEAR ALLTRACE` command.

Examples:	`DISP CLE`

Link:	`C.WINBottom:` *<NRx>*

Arguments:	The range for *<NRx>* is described below.

`C.WINBottom` specifies the bottom edge of a data selection window that is used when the `DISPLAY? DATA` command is issued. The following three `C.WIN` links specify the other three sides of the data selection window in scale units. The following illustration shows the data window parameters.

The data selection window defines the area of the screen from which pixel bins will be transferred. The data window is not visible or definable from the front panel. The window lets you to limit the amount of data transferred with the `DISPLAY? DATA` command.

The `C.WIN` links specify the sides in the current vertical and horizontal scale units (i.e., volts, seconds, rho, feet, etc.) of the selected trace. The `D.WIN` links specify the window in absolute divisions independent of the current scale settings.
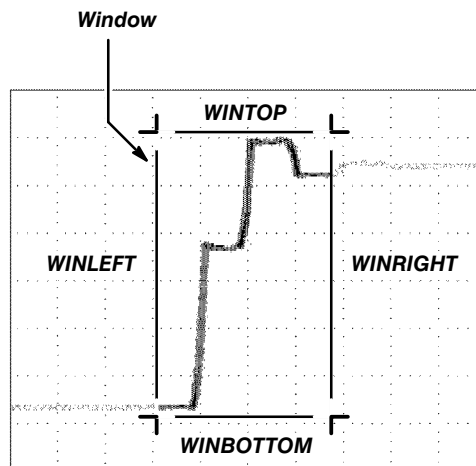
Figure 2-5: Data Window Parameters

C.WINBottom is a vertical value in units of the vertical scale for the selected trace. The range is defined by the graticule limits. The default is the graticule bottom. C.WINBottom can never be greater than C.WINtop.

Examples: DISP C.WINB:-2.5

Link: C.WINLeft: *<NRx>*

Arguments: The range for *<NRx>* is described below.

C.WINLeft specifies the left edge of a data selection window that is used when the DISPLAY? DATA command is issued.

C.WINLeft is a horizontal value in units of the horizontal scale for the selected trace. The range is defined by the end points of the trace record. C.WINLeft can never be greater than C.WINRight.

Examples: DISP C.WINL:1.15

Link: C.WINRight: *<NRx>*

Arguments: The range for *<NRx>* is described below.

C.WINRight specifies the right edge of a data selection window that is used when the DISPLAY? DATA command is issued.

C.WINRight is a horizontal value in units of the horizontal scale for the selected trace. The range is defined by the end points of the trace record. C.WINRight can never be less than C.WINLeft.

Examples: DISP C.WINR:4.05

Link: C.WINTop: *<NRx>*

Arguments: The range for *<NRx>* is described below.

C.WINTop specifies the top edge of a data selection window that is used when the DISPLAY? DATA command is issued.

C.WINTop is a vertical value in units of the vertical scale. The range is defined by the graticule limits. C.WINTop can never be less than C.WINBottom.

Examples: DISP C.WINT:1.5

Link: `? DATA`

Query Only. `? DATA` transfers the bin count for each pixel on a color graded display to the controller in binary or ASCII format. The pixel bin count is sent from a specified screen region (or window) that is initially set to the full height of the selected graticule and the full width of the trace record. The window can be sized with the `C.WIN` and `D.WIN` links. The size of the window will determine how much data is sent.

Pixel bin counts are sent as 16-bit values, a row at a time, starting from the upper left of the screen. *<Pixel data>* can be in ASCII (*<asc bin>*) or binary (*<bblock>*) format. The format is set by the `ENCDG DISPLAY` command. Use the `DISPLAY? NR.PT` query to get the number of pixel bin values to expect from the `DATA` query.

ASCII Transfer — Data transferred as an *<asc curve>* use the following format:

   <asc bin> ::= *<NR1>* [,*<NR1>*] ... EOI

where *<NR1>* values are pixel bin counts within the range 1 to 65535.

Binary Transfer — Data is transferred as comma separated binary blocks in the format:

   (*<bblock>* [,*<bblock>*, ...]) EOI

where

   <bblock>::=%*<byte cnt><bin pt>*...*<checksum>*

<bin pt> is a two-byte unsigned binary integer (MSB first) and *<byte cnt>* is an arbitrary number of binary bytes. This binary format is identical to that used for trace transfers with the `CURVe` command which is discussed on page 2-99.

The order of bytes within a bin count value is set with the `BYT.or` command. You can specify whether the least (`LSB`) or most (`MSB`) significant byte is sent first. There are no separators (such as commas) between data points.

Examples: **DISP? DATA**
DISPLAY DATA:14,10,14,223,1474,27,0,74,89,22,...

Link: `D.WINBottom: <NRx>`

Arguments: The range for `<NRx>` is described below.

`D.WINBottom` specifies the bottom edge of a data selection window that is used when the `DISPLAY? DATA` command is issued. The following three `D.WIN` links specify the other three sides of the data selection window in divisions.

The illustration on page 2-109 shows the data window parameters and their associated `WIN` link. The following illustration shows the coordinate system used to define `D.WIN` parameters. Because trace records extend slightly beyond the left and right graticule limits, the `D.WIN` limits slightly exceed the −5.12 and +5.10 values shown.

See the `C.WINBottom` discussion on page 2-109 for more information on the data selection window.
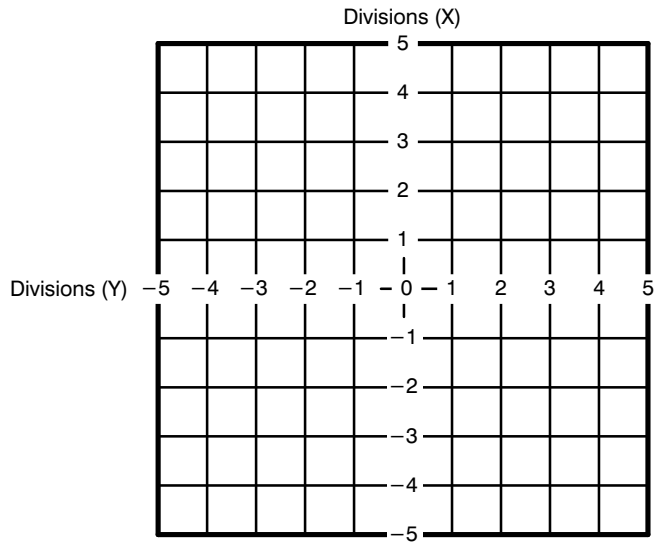
Figure 2-6: Graticule X, Y Coordinates

The `D.WINBottom` range is −5.12 to + 5.10 divisions, though `D.WINBottom`, can never be greater than `D.WINTop`.

Examples: `DISP D.WINB:−2.5`

Link: `D.WINLeft <NRx>`

Arguments: The range for `<NRx>` is described below.

`D.WINLeft` specifies the left edge of a data selection window that is used when the `DISPLAY? DATA` command is issued.

`D.WINLeft` is a horizontal value in divisions within the range of −5.12 to +5.10. `D.WINLeft` can never be greater than `D.WINRight`.

Examples: `DISP D.WINL:−1.15`

Link: `D.WINRight <NRx>`

Arguments: The range for `<NRx>` is described below.

`D.WINRight` specifies the right edge of a data selection window that is used when the `DISPLAY? DATA` command is issued.

`D.WINRight` is a horizontal value in divisions within the range of −5.12 to +5.10. `D.WINRight` can never be less than `D.WINLeft`.

Examples: `DISP D.WINR:4.05`

Link: `D.WINTop: <NRx>`

Arguments: The range for `<NRx>` is described below.

`D.WINTop` specifies the top edge of a data selection window that is used when the `DISPLAY? DATA` command is issued.

`D.WINTop` is a vertical value in divisions within the range of −5.12 to +5.10. `D.WINTop` can never be greater than `D.WINBottom`.

Examples: `DISP D.WINT:1.5`

Link: `GRAticule: { DUAl | SINgle }`

`GRAticule` selects dual or single graticules.

Examples: `DISP GRA:SIN`

Link: GRADFirst: { ON | OFF }

If the screen is cleared when GRADFirst is ON the screen is immediately redrawn. However, if the screen is cleared when GRADFirst is OFF, the first trace is not drawn on the screen until the display refresh time (set with DISP REFRE *<NRx>*) elapses. This increases the number of traces processed into the color graded database.

Examples: DISP GRADF:ON

Link: ? GRADScale:

Query Only. ? GRADScale returns the current color boundaries when the display is color graded (using the DISPLAY TYPE:GRADED command). Each number returned is an unsigned 16-bit value, *<ui>*, specifying the high cutoff of the number of hits in each boundary.

Examples: **DISP? GRADS**

Link: INTENsity: *<NRx>*

Arguments: *<NRx>* can range between 0 and 100 percent.

INTENsity sets the overall display intensity.

Examples: DISP INT:65

Link: MODe: { DOTs | VECtors }

MODe selects a DOTs or VECtors type display. DOTs displays individual data points, while VECtors connects adjacent data points.

NOTE

When more than 512 data points are acquired, the points are compressed to fit the 512- point scan line of the display. The largest and smallest adjacent vertical values are displayed as a single scan line connected with a vector. Thus to get a true dots display, you may need to set TBMain LENgth or TBWin LENgth to 512.

Examples: DISP MOD:VEC

Link: ? NR.PT *<NRx>*

Query Only. ? NR.PT returns the number of pixel bins that will be returned by DISPLAY? DATA.

Examples: **DISP? NR.PT**

Link: PERSistence: *<NRx>*

Arguments: *<NRx>* can range between 200 ms and 20 s, in steps of 200 ms.

PERSistence sets the length of time that data points are displayed when variable persistence is selected (DISPLAY TYPE set to VARiable).

Examples: DISP PERS:2.2E+0

Link: REFREsh: { 0 | *<NRx>* }

Arguments: *<NRx>* can be zero (0) or it can range between 5 and 180 s, in steps of 1 s. A zero value suppresses color-graded screen updates and the updates for histograms, histogram statistics, and measurements performed in statistics mode. A REFREsh value of zero does not affect mask counts.

REFREsh sets the time between display updates for histogram and color-graded displays.

NOTE

Screen updates for histogram information occur when the DSYs *command is issued and when the histogram limits change. Screen updates for measurements performed in* MMOde:STAT *mode occur when the* MSYs *command is issued and when any measurement parameter (for example, mesial, proximal, or zone) is changed.*

Examples: DISP REFRE:15

Link: STATIstics: { HISTogram | MASK }

STATIstics selects whether HISTogram or MASK statistics are displayed in the Display Mode menu when a trace is defined. Selecting DISPLAY STATISTICS with no trace defined produces an error, Error 250: No traces defined. When the last trace is removed, DISPLAY STATISTICS is set to HISTogram, the default value.

Examples: DISP STATI:HIST

Link: TYPe: { INFinite | NORmal | VARiable | GRADed }

TYPe selects the type of display persistence for all displayed traces with channel components. INFinite accumulates data points on the display indefinitely. VARiable leaves acquired data points on the display for a period of time specified by DISPlay PERSistence. GRADed produces a trace display similar to INFinite but the data points are color graded to represent the number of hits on each data point.

NOTE

> *The trace record length must be set to 512 points when using* INFinite, VARiable, *and* GRADed *display types. If the record length is greater than 512 when a non-*NORmal *type is selected, the record length will be set to 512, the new type will start, and Execution Warning Event 572 will be generated.*

If Histogram or Mask testing is started when the display type is NORmal or VARiable, the TYPe will change to INFinite.

Examples: DISP TYP:VAR

Link: ? XSize: *<NRx>*

Query Only. ? XSize returns the number of pixel bins in the width defined by the data window.

Examples: **DISP? XSIZE**
DISPLAY XSIZE:512

Link: ? YSize: *<NRx>*

Query Only. ? YSize returns the number of pixel bins in the height defined by the data window.

Examples: **DISP? YSIZE**
DISPLAY YSIZE:256

DISPNum?

Group: Trace and Setting Commands

DISPNum? returns the number of traces currently displayed on the screen.

Examples: **DISPN?**
DISPNUM 4

DIV2 { ON | OFF }

Group: Calibration Commands

When set to ON, DIV2 halves the internal calibrator oscillation rate, providing a convenient signal source for adjusting loop gain.

Examples: DIV2 ON

DOT1Abs *<link>*:*<arg>*;
DOT2Abs *<link>*:*<arg>*

Group: Cursor Commands

DOT1Abs and DOT2Abs set absolute horizontal positions (with respect to the trace record) for split or paired (dot) cursors. DOT1Abs and DOT2Abs have the same parameters. Figure 2-7 illustrates the graticule coordinates.
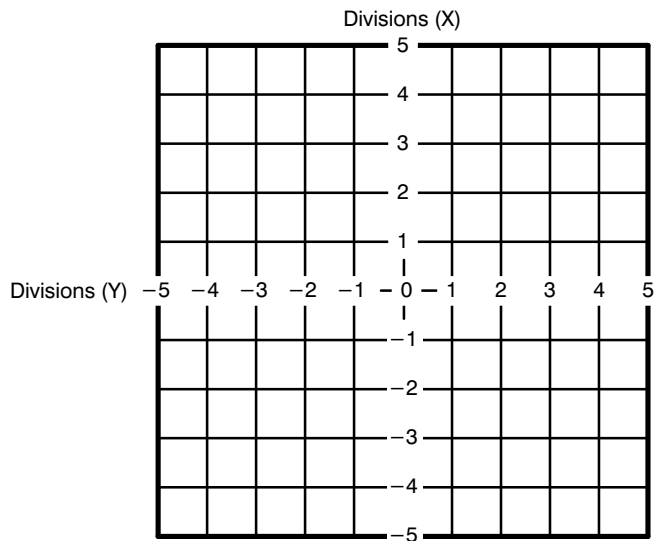
Figure 2-7: Graticule X, Y Coordinates

Link: ? OHMS: *<NR3>*

Query Only. OHMS returns the vertical cursor value in ohms, when the scale units are set to RHO (GRAticule YUNit command).

Examples: **DOT1A? OHMS**
DOT1ABS OHMS:1.0E+2

Link: ? OQUal: { EQ | LT | GT | UN | ER }

Query Only. Returns a qualifier indicating the accuracy and appropriateness of the value returned with the ? OHMS query link.

EQ   Returned value equals actual value.

LT   Returned value less than actual value.

GT   Returned value greater than actual value.

UN   Returned value is uncertain.

ER   Returned value is meaningless, not rho scale trace.

Examples: **DOT1A? OQUAL**
DOT1ABS OQUAL:EQ

Link: PCTg: *<NRx>*

Arguments: *<NRx>* can range between 0 and 100 percent.

PCTg positions the first or second dot cursor as a percentage of the trace record.

XY Note: You should use the PCTg link to position the cursors for XY traces. Attempting to use XCOord or XDIV will give unpredictable results.

Examples: DOT2A PCT:10

Link: XCOord: *<NRx>*

Arguments: The range for *<NRx>* is described below.

XCOord positions the first or second dot cursor with respect to horizontal units of the selected trace.

(The following range formulas assume ADJtrace PANzoom is set to OFF and the trace is acquired. Refer to the cursor positioning discussion on page 2-120 for calculating XCOord range when PANzoom is set to ON or the trace is unacquired. Refer to page 2-214 for formulas to calculate *duration*.)

XCOord range when the selected trace record is MAIN:

MAINPos to (MAINPos + *main_duration*)

XCOord range when the selected trace record is WIN:

WINPos to (WINPos + *win_duration*)

Examples: DOT1A XCO:1.2E-2

Link: XDIv: *<NRx>*

Arguments: The range for *<NRx>* is described below.

XDIv positions the first or second dot cursor in graticule divisions (refer to the graticule illustration on page 2-117). Range depends on record (TBMain or TBWin) LENgth:

Table 2-55: XDIV Ranges

Record LENGTH	XDIV Range
512	−5.12 to +5.10
1024	−5.12 to +5.10
2048	−5.12 to +5.10
4096	−5.12 to +3.07
5120	−5.12 to +5.10

These ranges are valid only when ADJtrace PANzoom is OFF and the selected trace is acquired. (Refer to the Range of Cursor Positioning discussion on page 2-120 for calculating XCOord range when PANzoom is set to ON or the trace is unacquired.)

Examples: DOT2A XDI:2.85

Link: ? XQUal: { EQ | LT | GT | UN }

Query Only. ? XQUal returns the accuracy of XCOord or XDIv positioning information. YT traces always return the EQ qualifier because the cursor horizontal position is always known.

Table 2-56: Positioning Accuracy Qualifiers

Qualifier	Meaning
EQ	True position and response are equal
LT	True position is lower than response (i.e., the cursor is below the bottom of the screen)
GT	True position is greater than response (i.e., the cursor is above the top of the screen)
UN	True position is uncertain (i.e., the cursor is on an unacquired trace point)

Examples: **DOT1A? XQU**
DOT1ABS XQUAL:EQ

Link: `? YCOord: <NR3>`

Query Only. `? YCOord` returns the vertical position of the first or second dot cursor, in units of the selected trace.

Examples: **`DOT2A? YCO`**
`DOT2ABS YCOORD:2.22E-4`

Link: `? YDIv: <NR3>`

Query Only. `? YDIv` returns the vertical position of the first or second dot cursor in graticule divisions. (See Figure 2-7.)

Examples: **`DOT1A? YDI`**
`DOT1ABS YDIV:-1.4`

Link: `? YQUal: { EQ | LT | GT | UN }`

Query Only. `? YQUal` returns the accuracy of `YCOord` or `YDIv` positioning information. Refer to the description of the link `DOT1Abs XQUal` for the meanings of the qualifiers.

Examples: **`DOT1A? YQU`**
`DOT1ABS YQUAL:EQ`

Range of Cursor Positioning. Under some circumstances, such as when `PANzoom` is set to `ON`, you cannot conveniently compute the valid range of cursor positions. However, you can force the cursors to their minimum and maximum values (use the `PCTG:0` and `PCTG:100` links) and then query the instrument for the cursor positions. These new positions constitute the valid range of cursor positions for that particular instrument setup.

The following example demonstrates this technique. This method applies to both dot and bar cursors and is always successful, regardless of instrument settings.

Examples: `DOT1A PCT:0`
`DOT2A PCT:100`
`DOT1A? XCO;DOT2A? XCO`
`DOT1ABS XCOORD:-6.0E-6;`
`DOT2ABS XCOORD:5.055E-4`

DOT1REl *<link>***:***<arg>* ;
DOT2REl *<link>***:***<arg>*

Group:	Cursor Commands
	Set Only. DOT1Rel and DOT2Rel set the paired or split (dot) cursor position relative to (offset to the right of) the absolute cursor location. DOT1Rel and DOT2Rel have the same links.
Link:	PCTg: *<NRx>*
Arguments:	See the discussion for DOT1A PCTg for the range for *<NRx>*.
	Set Only. PCTg positions the first or second dot cursor as a percentage of the trace record, relative to but not exceeding the DOT1Abs/DOT2Abs value.
Examples:	DOT1R PCT:50
Link:	XCOord: *<NRx>*
Arguments:	See the discussion for DOT1A XCO for the range for *<NRx>*.
	Set Only. XCOord positions the first or second dot cursor with respect to the units of the selected trace, relative to but not exceeding the DOT1Abs/DOT2Abs value.
Examples:	DOT2R XCO:0.5
Link:	XDIv: *<NRx>*
Arguments:	See the discussion for DOT1A XDI for the range for *<NRx>*.
	Set Only. XDIv positions the first or second dot cursor in graticule divisions with respect to the selected trace, relative to but not exceeding the DOT1Abs/DOT2Abs value.
Examples:	DOT2R XDI:2.85

DSKJETC *<link>***:***<arg>*

Group: External I/O Commands

DSKJETC specifies printing parameters for HP DeskJet color printers operating in HP graphics mode. The command is identical to HP560C.

NOTE

DSKJETC *does not support DeskJet printers operating in Epson emulation mode.*

Link: DIRection: { HORiz | VERt }

DIRection selects the printing orientation. HORiz prints rows left-to-right and top-to-bottom. VERt prints columns bottom-to-top and left-to-right.

Examples: DSKJETC DIR:HOR

Link: FORMat: { DIThered | DRAft | REDuced | SCReen }

FORMat selects the printing format. DIThered modifies print contrast for the DSKJETC. DRAft prints monochrome. REDuced is a quarter the size of DRAft. SCReen is a one-to-one mapping of 3-bit pixel information.

Examples: DSKJETC FORM:DRA

Link: PORt: { CENTRonics | GPIb | RS232 }

PORt specifies the output port for the printer.

Examples: DSKJETC POR:RS232

Link: ? COLor

Query Only. ? COLor returns the mapping of printer colors and the instrument color index.

Examples: **DSKJETC? COL**
DSKJETC COLOR0:4095, COLOR1:0, COLOR2:3920, COLOR3:241, COLOR4:1295, COLOR5:255, COLOR6:2457, COLOR7:3841

Link: ? COLORn

Query Only. ? COLORn returns the mapping of printer color n and the instrument color index.

Examples: `DSKJETC? COLOR`
`DSKJETC COLOR0:4095, COLOR1:0, COLOR2:3920, COLOR3:241,`
`COLOR4:1295, COLOR5:255, COLOR6:2457, COLOR7:3841`

Link: `COLor<ui>: <NRx>`

Arguments: `<ui>` can range between 0 and 7. `<NRx>` can range between 0 and 4095.

Set Only. `color<ui>` assigns colors to the instrument color index as shown in Table 2-59.

Table 2-57: Default DSKJETC Color Assignments

Color Index	DSKJETC Color	Color Index	DSKJETC Color
0	4095 (0xFFF)	4	1295 (0x50F)
1	0 (0x000)	5	255 (0x0FF)
2	3920 (0xF50)	6	2457 (0x999)
3	241 (0x0F1)	7	3841 (0xF01)

The DSKJETC color is specified by the value `<NRx>` which maps to a hexadecimal $0xh_1h_2h_3$. The variables $h_1h_2h_3$ specify the RGB components of the color.

DSYmenu**?**

Group: Miscellaneous/System Commands

Query Only. DSYmenu? returns the major menu active on the front panel display.

Possible DSYmenu? responses:

```
CURSOR          DISPLAY_MODES   ALL_WAVFRM
MEAS            STORE_RECALL    UTILITY1
TRIGGER         WAVFRM          ENH_ACCURACY
UTILITY2
```

NOTE

ALL_WAVFRM is the "More..." trace status menu, ENH_ACCURACY is the "Page to" second page of the Utility menu, and UTILITY2 is the "Page to" third page of the Utility menu.

Examples: **DSY?**
DSYMENU CURSOR

DSYS

Group: Display and Color Commands

DSYS enables the histogram and mask acquisition functions by selecting the Display Modes major menu on the front panel display.

Link: { ON | OFF }

ON selects the Display Modes major menu. OFF selects the Waveform major menu. The query form DSYS? returns the current ON or OFF status.

Examples: **DSYS?**
DSYS ON

DUTy**?**

Group: Measurement Commands

Query Only. DUTy? returns the percentage of a period that the selected trace spends above the mesial level, followed by an accuracy qualifier. (See page 2-168 for qualifier definitions.)

Examples: **DUT?**
DUTY 5.071E+1,EQ

ENCdg *<link>:<arg>*

Group: Data Transfer Commands

ENCdg determines the data encoding for information returned by CURVe?, DISPlay?, HISTogram?, WAVfrm?, and SET? queries.

Link: DISPlay: { ASCii | BINary }

DISPlay sets the encoding for pixel point counts transferred with the DISPLAY? DATA query.

Examples: ENC DISP:ASC

Link: HISTogram: { ASCii | BINary }

HISTogram sets the encoding for data points in a histogram curve transferred with the HISTOGRAM? DATA query.

Examples: ENC HIST:ASC

Link: SET: { ASCii | BINary }

SET sets the encoding for front panel setting (FPS) transfers with the SET? query.

Examples: ENC SET:ASC

Link: WAVfrm: { ASCii | BINary }

WAVfrm sets the encoding for trace transfers with the CURVe? and WAVfrm? queries.

Examples: ENC WAV:BIN

ENV { ON | OFF }

Group: Acquisition Commands

ENV sets enveloping ON or OFF for the vertical expression component *<y exp>* (e.g., "M1") of the trace description of the selected trace. (Refer also to the TRAce and AVG commands.)

■ When *<y exp>* is not enclosed with AVG and ENV is set to ON, *<y exp>* is enclosed with ENV().

■ When *<y exp>* is enclosed with AVG and ENV is set to ON , ENV() replaces AVG().

■ When *<y exp>* is enclosed with ENV() and ENV is set to OFF, the enclosing ENV() is removed.

NOTE

You cannot set ENV *to* OFF *when the* <y exp> *is not enclosed with* ENV()*. You cannot set* ENV *to* ON *if the selected trace is XY or has only stored and/or scalar components.*

Table 2-58: Examples of ENV Usage

<y exp> **Before**	**Command**	*<y exp>* **After**
M2	ENV ON	ENV(M2)
M1	ENV OFF	−error−
AVG(M1−M2)	ENV ON	ENV(M1−M2)
ENV(M1)	ENV OFF	M1
ENV(M4)	ENV ON	ENV(ENV(M4))

Query Note: ENV? returns the state of enveloping. ENV ON means the entire *<y exp>* is enclosed by ENV. ENV OFF means the entire *<y exp>* is not enclosed by ENV, though the ENV() function may be embedded within the description.

EVENT?

Group: Status and Event Commands

Query Only. EVENT? returns the event code <*NR1*> if LONgform is set to OFF, or returns the event code and a descriptive <*qstring*> if LONgform is set to ON.

Refer to Status and Events, later in this manual, for a list of event codes.

Examples: **EVENT?**
EVENT 269,"NO SUCH TRACE"

EXTInction?

Group: Measurement Commands

Query Only. EXTInction? returns the selected trace's extinction ratio. Depending upon the setting of the MPAram ERMethod link, the extinction ratio can be calculated using any of three formulas. Refer to the MPAram command description. EXTInction? is available only when MMOde is set to either SW or STAT.

Examples: **EXTI?**
EXTINCTION 8.2353E-1, EQ

EYEPattern { ON | OFF }

Group: Measurement Commands

EYEPattern ON specifies that the selected trace is an eye pattern and causes the instrument to automatically set the mesial level at the calculated eye-crossing level when STAT mode measurements are performed. When the selected trace changes, then the current EYEPattern mode applies to the new trace.

FALltime?

Group: Measurement Commands

Query Only. FALltime? returns the transition time for the selected trace of a falling pulse edge, from the DISTal to PROXimal level, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **FAL?**
FALLTIME 5.883E-9,EQ

FEOi

Group: External I/O Commands

Set Only. `FEOi` forces the instrument to output a message terminator for any pending query response. (The message terminator for GPIB is an EOI signal with or without LF; the message terminator for RS232 is the EOL string. Refer to the `RS232` command for the EOL options.) `FEOi` is useful to force the output of a recursive query (created with the DEF command) onto individual lines.

NOTE

`FEOi` *has no argument.*

Examples: FEO

FFT {*<link>*:*<arg>* | **?** *<link>*:*<arg>*}

Group: Trace and Settings Commands

`FFT` controls the Fast Fourier Transform (FFT) parameters. The FFT function is part of the waveform description. (Refer to the `TRAce` command.)

Link: FORMat: { DBM | LINear | VPEak }

`FORMat` specifies the magnitude output format.

`DBM` causes the FFT magnitude to be displayed in dB (decibel units relative to 1 mW; for example, a sine wave of 0.316 V_{peak} (0.224 V_{RMS}) will give 1 mW into 50 Ω and will display an FFT magnitude of 0 dB. Signals of a lesser magnitude have a negative dB value.

`LINear` and `VPEak` cause display of the FFT magnitude in volts.

Examples: FFT FORM:LIN

Link: WINDow: { BLACKman | BLHarris | HAMming | HANning | RECTangular | TRIAngular }

`WINDow` specifies the window used to remove the effects of time sampling. The algorithms associated with these windows are included in the *User Manual*.

Examples: FFT WIND: HAM

FMOde { CENter | SHIft }

Group: Acquisition Commands

`FMOde` sets the mode for the waveform filtering function. The filtering operation replaces each point with the average of some number of adjacent points, depending on the requested filter risetime. A `CENter` filter is non-causal and uses an average that is centered on each point. A `SHIft` filter is causal and forms the average for each point based only on preceding points.

FPAnel { ON | OFF }

Group: Miscellaneous/System Commands

`FPANEL OFF` functionally mimics the GPIB RWLS (Remote With Lockout State) and `FPANEL ON` mimics the GPIB LOCS (Local State).

When `FPAnel` is set to `OFF`, the front panel is locked out and only these controls are operable:

■   RQS icon, if it was enabled (displayed) with the `SRQMASK USER:ON` command. (The RQS icon is not displayed at power on.) If enabled, you can disable the RQS icon with `SRQMASK USER:OFF`.

Front panel controls can be locked out either by setting `FPAnel` to `OFF` or by using IEEE−488 interface commands to enter the GPIB RWLS state. If the instrument is in RWLS, the front panel will be inoperative even if `FPAnel` is set to `ON`.

When `FPAnel` is set to `ON`, all front panel controls are operable, assuming the **TOUCH PANEL ON**/**OFF** button is set to `ON`.

The differences between the `FPAnel` command and the **TOUCH PANEL ON**/**OFF** button are:

■   `FPAnel` provides a way to lock out active front panel controls (knobs, buttons, and screen touches) from the remote interfaces. There is no front panel equivalent to `FPAnel`.

■   The **TOUCH PANEL ON**/**OFF** button only locks out screen touches. No command mimics the effect of this button. However, you can use the `ABSTouch` command to simulate a touch to this button from the remote interfaces.

Examples: `FPA ON`

FPSList**?**

Group: Trace and Settings Commands

Query Only. `FPSList?` returns a list of all front panel settings stored in nonvolatile RAM (NVRAM) identified by FPS number (1 to 9999), and the amount of NVRAM used to store each setting. `FPSList?` returns `EMPty` if no settings are stored.

```
{ FPS<ui>: <len> [ { ,FPS<ui>:<len> }...] }
    | EMPty
```

Examples: **FPSL?**
FPSLIST FPS2:8192, FPS9915:7168

FPSNum**?**

Group: Trace and Settings Commands

Query Only. `FPSNum?` returns the number of front panel settings (FPS) stored in Nonvolatile RAM, in `<NR1>` form. The range is from 0 to 9999.

Examples: **FPSN?**
FPSNUM 2

FPUpdate { **ON** | **OFF** }

Group: Miscellaneous/System Commands

`FPUpdate` determines whether the front panel display readouts are updated following set command execution. The power-on default is `FPU OFF`.

With `FPU ON`, the front panel display is updated after each successful set command.

With `FPU OFF`, the front panel display is only updated when:

■ the instrument receives DCL or SDC

■ the instrument receives an incorrect query or set command

■ the instrument input buffer is empty after a successful set or query execution

NOTE

Throughput is much faster with FPUpdate *set to* OFF.

Examples: FPU OFF

FREq?

Group: Measurement Commands

Query Only. `FREq?` returns the frequency of the selected trace, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **FRE?**
FREQ 1.024E+6,EQ

GRAticule *<link>:<arg>*

Group: Display and Color Commands

Sets X and Y axis units and scaling for displayed trace.

Link: `PROPvelocity: <NRx>`

Arguments: *<NRx>* can range between 0.1 and 1.0.

`PROPvelocity` sets propagation velocity (a fraction of the speed of light in a vacuum). Used to scale X axis when `XUNit` is `FEEt`, `INChes`, or `METers`.

Examples: `GRA PROP:0.365`

Link: `REFAmplitude: <NRx>`

Arguments: *<NRx>* can range between 50 mV and 1 V.

`REFAmplitude` sets the pulse amplitude used to scale the Y axis when `YUNit` is set to `RHO`.

Examples: `GRA REFA:2.5E-1`

Link: `XUNit: { FEEt | INChes | METers | SEConds }`

`XUNit` selects scale units for the X axis.

Examples: `GRA XUN:SEC`

Link: YUNit: { VOLts | RHO }

YUNit selects scale units for the Y axis.

Examples: GRA YUN:VOL

Query Note: GRA? returns its links and arguments in the following order:

GRATICULE PROPVELOCITY:<*NR3*>, REFAMPLITUDE:<*NRx*>, XUNIT:<*arg*>, YUNIT:<*arg*>

H1Bar *<link>:<arg>;*
H2Bar *<link>:<arg>*

Group: Cursor Commands

H1Bar and H2Bar set the absolute vertical position of horizontal bar cursors. H1Bar and H2Bar have the same parameters.

Link: ? OHMS: *<NR3>*

Query Only. ? OHMS returns the vertical cursor value in ohms, when that form of readout is appropriate.

Examples: **H1B? OHMS**
H1BAR OHMS:1.0E+2

Link: ? OQUal: { EQ | LT | GT | UN | ER }

Query Only. ? OQUal returns a qualifier indicating the accuracy and appropriateness of the value returned with the ?OHMS query link.

EQ   Returned value equals actual value.

LT   Returned value less than actual value.

GT   Returned value greater than actual value.

UN   Returned value is uncertain.

ER   Returned value is meaningless, not rho scale trace.

Examples: **H2B? OQUAL**
H2BAR OQUAL:EQ

Link: YCOord: *\<NRx\>*

Arguments: The range for *\<NRx\>* is described next.

YCOord positions the first or second horizontal bar cursor with respect to the units of the selected trace. The range depends on whether the trace was created in integer mode or floating-point mode.

NOTE

For information on trace modes, see the WFMScaling *command.*

The YCOord range for an integer mode trace is:

(SEN * −5.12 + OFFS) to (SEN * 5.10 + OFFS)

where SEN and OFFS are the channel sensitivity and offset (CH*\<slot\>\<ui\>*? SEN,OFFS) of the channel(s) in the integer mode trace.

The YCOord range for a floating-point mode trace is:

(VSI * −5.12 + VPO) to (VSI * 5.10 + VPO)

where VSI and VPO are the vertical size and vertical position (ADJ*\<ui\>*? VSI,VPO) of the floating-point trace.

Examples: H2B YCO:0.75

Link: YDIv: *\<NRx\>*

Arguments: *\<NRx\>* can range between −5.12 and +5.10.

YDIv positions the first or second horizontal bar cursor in graticule divisions.

Examples: H1B YDI:−4.0

HISTogram *<link>:<arg>*

Group: Display and Color Commands

HISTogram initiates a vertical or horizontal histogram for the selected trace. It also sets a variety of histogram parameters including the dimensions of a displayed histogram window.

The histogram window selects a portion of the trace on which to perform the histogram algorithm. The histogram window appears on the display when the TYPe is set to HORiz or VERt and the Display Modes major menu is selected (see the DSYS command on page 2-124). To start accumulating statistics, set DSYS to ON. Then, set DISPlay STATIstics:HISTogram to actually display the statistics and the histogram.

Each displayed trace has a unique histogram window and a unique histogram TYPe. Once a histogram is started on a trace, selecting another trace will activate the histogram window for that trace. Histograms started on a color graded display will calculate the histogram from the accumulated pixel bin data.

The C.WIN links specify the sides in the current vertical and horizontal scale units (i.e., volts, seconds, rho, feet, etc.) of the selected trace. The D.WIN links specify the window in absolute divisions independent of the current scale settings. The histogram window can be defined with C.WIN links then queried with D.WIN links, and vice versa. Figure 2-8 shows the four histogram window parameters.
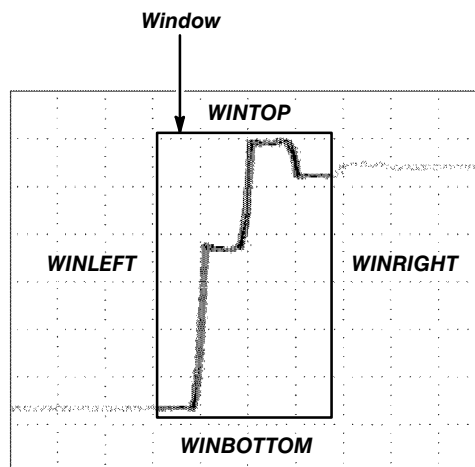
Figure 2-8: Histogram Window Parameters

The histogram window *is not* related to the data window defined with the DISPlay command.

For more information on the use of the Histogram function, refer to the *User Manual* for your instrument.

Link: `CLEar`

`CLEar` removes all trace and histogram data from the display and restarts all acquisitions. `CLEar` is equivalent to the `CLEar ALLTrace` command.

Examples: `HIST CLE`

Link: `C.WINBottom:  <NRx>`

Arguments: The range for `<NRx>` is described below.

`C.WINBottom` specifies the bottom edge of the histogram window for the selected trace. `C.WINBottom` is a vertical value in the current units of the vertical scale. The range is defined by the vertical graticule limits. See the `D.WINBottom` link for the default setting. `C.WINBottom` can never be greater than `C.WINTop`.

The `<NRx>` range for an integer mode trace is:

(SEN * −5.12 + OFFS) to (SEN * 5.10 + OFFS)

where `SEN` and `OFFS` are the channel sensitivity and offset (`CH<slot><ui>? SEN,OFFS`) of the channel(s) in the integer mode trace.

The `<NRx>` range for a floating-point mode trace is:

(VSI * −5.12 + VPO) to (VSI * 5.10 + VPO)

where `VSI` and `VPO` are the vertical size and vertical position (`ADJ<ui>? VSI,VPO`) of the floating-point trace.

Examples: `HIST C.WINB:−2.5`

Link: `C.WINLeft:  <NRx>`

Arguments: The range for `<NRx>` is described below.

`C.WINLeft` specifies the left edge of the histogram window for the selected trace. `C.WINLeft` is a horizontal value in units of the horizontal scale. The range is defined by the endpoints of the trace record. See the `D.WINLeft` link for the default setting. `C.WINLeft` can never be greater than `C.WIN-Right`.

`<NRx>` range when the selected trace record is MAIN:

MAINPos to (MAINPos + *main_duration*)

`<NRx>` range when the selected trace record is WIN:

WINPos to (WINPos + *win_duration*)

Examples: `HIST C.WINL:1.15`

Link: `C.WINRight: <NRx>`

Arguments: The range for `<NRx>` is described below.

`C.WINRight` specifies the right edge of the histogram window for the selected trace. `C.WINRight` is a horizontal value in units of the current horizontal scale. The range is defined by the endpoints of the trace record. See the `D.WINLeft` link for the default setting. `C.WINRight` can never be less than `C.WINLeft`.

`<NRx>` range when the selected trace record is MAIN:

MAINPos to (MAINPos + *main_duration*)

`<NRx>` range when the selected trace record is WIN:

WINPos to (WINPos + *win_duration*)

Examples: `HIST C.WINR:4.05`

Link: `C.WINTop: <NRx>`

Arguments: The range for `<NRx>` is described below.

`C.WINTop` specifies the top edge of the histogram window for the selected trace. `C.WINTop` is a vertical value in units of the vertical scale. The range is defined by the vertical graticule limits. See the `D.WINTop` link for the default setting. `C.WINTop` can never be less than `C.WINBottom`.

The `<NRx>` range for an integer mode trace is:

(SEN * −5.12 + OFFS) to (SEN * 5.10 + OFFS)

where `SEN` and `OFFS` are the channel sensitivity and offset (CH*<slot><ui>*? SEN,OFFS) of the channel(s) in the integer mode trace.

The `<NRx>` range for a floating-point mode trace is:

(VSI * −5.12 + VPO) to (VSI * 5.10 + VPO)

where `VSI` and `VPO` are the vertical size and vertical position (ADJ*<ui>*? VSI,VPO) of the floating-point trace.

Examples: `HIST C.WINT:1.5`

Link: `? DATA`

Query Only. `? DATA` transfers the value of each point on the histogram curve to the controller in binary or ASCII format. The histogram window determines what portion of the trace will be incorporated in the histogram. It also determines how much histogram data will be transferred.

Histogram data points are sent as unsigned 32-bit values starting from the left of the screen for horizontal histograms and from the bottom for vertical histograms. You can specify the data format to be either ASCII or binary with the `ENCDG HISTOGRAM` command.

`<Histogram data>` can be in ASCII (*`<asc>`*) or binary (*`<bblock>`*) format. The format is set by the `ENCDG HISTOGRAM` command. Use the `HISTO-GRAM? NR.PT` query to get the number of histogram points to expect from the `DATA` query.

ASCII Transfer. Data transferred as an *`<asc curve>`* use the following format:

   `<asc bin>` `::= ` *`<NR1>`* ` [ ,` *`<NR1>`* ` ] ... EOI`

where *`<NR1>`* values are pixel bin counts within the range 1 to 4294967295.

Binary Transfer. Data is transferred as a single binary block (*`<bblock>`*) in the format:

   `<bblock>` ` EOI`

where

   `<bblock>``::=%`*`<byte cnt>`**`<bin pt>`*`...`*`<checksum>`*

`<bin pt>` is a four-byte unsigned binary integer (MSB first) and *`<byte cnt>`* is an arbitrary number of binary bytes. This binary format is similar to that used for trace transfers with the `CURVe` command which is discussed on page 2-99.

The order of bytes within a bin count value is set with the `BYT.or` command. You can set either the least significant byte (LSB) to be sent first followed by bytes of greater significance or the most significant byte (MSB) first followed by bytes of lesser significance. There are no separators (such as commas) between binary bin counts.

Examples: **HIST? DATA**
HISTOGRAM DATA:0,1,552,8580,32321,
110778,209786,442794,939335,...

Link: D.WINBottom: *<NRx>*

Arguments: The range for *<NRx>* is described below.

D.WINBottom specifies the bottom edge of the histogram window for the selected trace. D.WINBottom is a vertical value in divisions.

The following illustration shows the coordinate system used to define D.WIN parameters. Because trace records extend slightly beyond the left and right graticule limits, the D.WIN limits slightly exceed the −5.0 and +5.0 values shown. The illustration on page 2-134 shows the data window parameters and their associated WIN link.

See the histogram discussion on page 2-134 for more information on the data selection window.
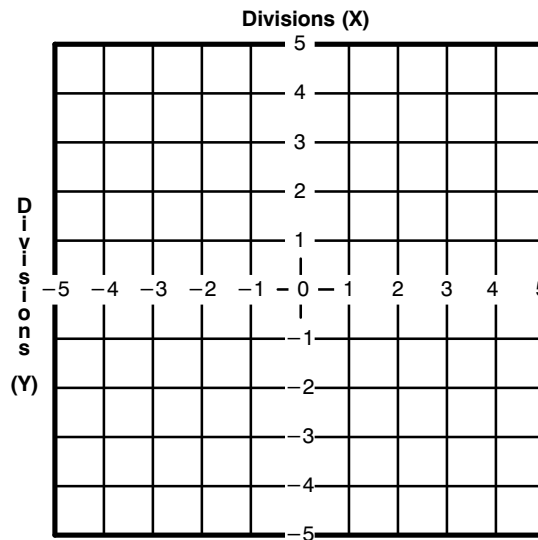
Figure 2-9: Graticule X, Y Coordinates

The D.WINBottom range (*<NRx>*) is −5.12 to + 5.10 divisions, though D.WINBottom can never be greater than D.WINTop.

Examples: HIST D.WINB:−2.5

Link: `D.WINLeft: <NRx>`

Arguments: The range for `<NRx>` is described below.

`D.WINLeft` specifies the left edge of the histogram window for the selected trace.

`D.WINLeft` is a horizontal value in divisions within the range of −5.12 to +5.10. `D.WINLeft` can never be greater than `D.WINRight`.

Examples: `HIST D.WINL:−1.15`

Link: `D.WINRight: <NRx>`

Arguments: The range for `<NRx>` is described below.

`D.WINRight` specifies the right edge of the histogram window for the selected trace.

`D.WINRight` is a horizontal value in divisions within the range of −5.12 to +5.10. `D.WINRight` can never be less than `D.WINLeft`.

Examples: `HIST D.WINR:4.05`

Link: `D.WINTop: <NRx>`

Arguments: The range for `<NRx>` is described below.

`D.WINTop` specifies the top edge of the histogram window for the selected trace.

`D.WINTop` is a vertical value in divisions within the range of −5.12 to +5.10. `D.WINTop` can never be greater than `D.WINBottom`.

Examples: `HIST D.WINT:1.5`

Link: `HISTScaling: { LINear | LOG10 }`

`HISTSCaling` selects either linear or logarithm base 10 scaling for the histogram display. All traces are affected. The initial default is `LINear`.

Examples: `HIST HISTS:LOG`

Link: `? NR.PT <NRx>`

Query Only. `?` `NR.PT` returns the number of histogram curve points that will be returned by `HISTOGRAM? DATA`.

Examples: **HIST? NR.PT**
`HISTOGRAM? NR.PT:201`

Link: `TYPe: { HORiz | VERt | NONe }`

`TYPe` selects the type of histogram display for the selected trace. The selected trace must be an infinite persistence or color graded trace display.

`HORiz` accumulates bin counts for each data point along the horizontal axis. `VERt` accumulates bin counts for each data point along the vertical axis. `NONe` disables the histogram function for the selected trace. A histogram `TYPe` can be specified for each trace. The histogram window (`C.WIN` and `D.WIN` links) determines what portion of the trace is included in the histogram calculation.

Select the Display Modes major menu with the `DSYS ON` command before issuing `HISTogram` commands. `HISTogram` commands are not allowed when any other major menu is active.

NOTE

If Histogram is started when the display type is `NORmal` *or* `VARi-able`, *the* `DISPlay TYPe` *will change to* `INfinite` *persistence while* `DSYS` *is* `ON`.

Examples: `HIST TYP:VER`

HP560C *<link>***:***<arg>*

Group: External I/O Commands

          HP560C specifies printing parameters for HP DeskJet 560C printers operating in HP graphics mode. This command is identical to DSKJETC.

NOTE

          HP560C *does not support DeskJet printers operating in Epson emulation mode.*

Link: DIRection: { HORiz | VERt }

          DIRection selects the printing orientation. HORiz prints rows left-to-right and top-to-bottom. VERt prints columns bottom-to-top and left-to-right.

Examples: HP560C DIR:HOR

Link: FORMat: { DIThered | DRAft | REDuced | SCReen }

          FORMat selects the printing format. DIThered modifies print contrast for the HP 560C. DRAft prints monochrome. REDuced is a quarter the size of DRAft. SCReen is a one-to-one mapping of 3-bit pixel information.

Examples: HP560C FORM:DRA

Link: PORt: { CENTRonics | GPIb | RS232 }

          PORt specifies the output port for the printer.

Examples: HP560C POR:RS232

Link: ? COLor

          Query Only. ? COLor returns the mapping of printer colors and the instrument color index.

Examples: **HP560C? COL**
          HP560C COLOR0:4095, COLOR1:0, COLOR2:3920, COLOR3:241, COLOR4:1295, COLOR5:255, COLOR6:2457, COLOR7:3841

Link: ? COLORn

          Query Only. ? COLORn returns the mapping of printer color n and the instrument color index.

Examples: `HP560C? COLOR`
`HP560C COLOR0:4095, COLOR1:0, COLOR2:3920, COLOR3:241,`
`COLOR4:1295, COLOR5:255, COLOR6:2457, COLOR7:3841`

Link: `COLor<ui>: <NRx>`

Arguments: `<ui>` can range between 0 and 7. `<NRx>` can range between 0 and 4095.

Set Only. `color<ui>` assigns colors to the instrument color index as shown in Table 2-59.

Table 2-59: Default HP 560C Color Assignments

Color Index	560C Color	Color Index	560C Color
0	4095 (0xFFF)	4	1295 (0x50F)
1	0 (0x000)	5	255 (0x0FF)
2	3920 (0xF50)	6	2457 (0x999)
3	241 (0x0F1)	7	3841 (0xF01)

The HP 560C color is specified by the value `<NRx>` which maps to a hexadecimal $0xh_1h_2h_3$. The variables $h_1h_2h_3$ specify the RGB components of the color.

HPGl *<link>***:***<arg>*

Group: External I/O Commands

HPGl specifies printing parameters for the Tek HC100 plotter or other devices that conform to the HPGl format.

Link: COLor*<ui>*: *<NRx>*

Arguments: *<ui>* can range between 0 and 7. *<NRx>* can range between 1 and 8.

COLor*<ui>* assigns plotter pens to the instrument color index. The pen range is specified by *<NRx>*. Refer to page 2-90 for the color index.

NOTE

Assigning pen 0 to the color index means that color is not plotted (no pen is assigned).

Examples: HPG COL3:1

Link: COLor: DEFAult

Set Only. COLOR:DEFAULT assigns the default pens shown in Table 2-60 to the color index.

Table 2-60: Default Plotter Pen Assignments

Color Index	Pen No.	Color Index	Pen No.
0	0	4	4
1	1	5	5
2	2	6	6
3	3	7	7

Examples: HPG COL:DEFA

Link: `FORMat: { DRAft | HIRes | SCReen }`

`FORMat` selects the output format. `HIRes` plots the entire screen, including every trace point. `SCReen` plots the entire screen, but includes only the min/max point-pairs of each YT trace column (XY and PA traces are not affected.) `DRAft` is the same as `SCReen` except the front panel status menu is not plotted.

NOTE

Pop-up menus are not plotted.

Plotting variable and infinite persistence traces is very time consuming and tends to wear down plotter pen points more rapidly than other types of plots. Each point is plotted separately.

Examples: `HPG FORM:DRA`

Link: `PORt: { CENTRonics | GPIb | RS232 }`

`PORt` specifies the output port for the plotter.

Examples: `HPG POR:CENTR`

HREfpt { LEFt | CENter | RIGht }

Group: Trace and Setting Commands

`HREfpt` selects the horizontal point about which the trace expands when the horizontal size is adjusted for Main and Window traces.

Examples: `HREFPT LEFT`

ID?

Group: Status and Event Commands

Query Only. `ID?` returns identifying information about the instrument and its firmware, delimited by commas. The list contains the following items:

- Instrument model number

- TEK Codes & Formats version number

- Time base processor (TBC) firmware version number

- Display Processor (DSY) firmware version number

- Executive Processor (EXP) firmware version number

- Acquisition processor (ACQM for the CSA 803C, ACQM1−ACQD8 for the 11801C) firmware version numbers

```
ID TEK/CSA 803C, V81.1, TBC/<NR2>, DSY/<NR2>,
    EXP/<NR2>, ACQM1/<NR2>
```

Examples: `ID?`
```
ID TEK/CSA 803C,V81.1,TBC/3.72,
DSY/3.13,EXP/3.18,ACQM1/9.02
```

INIt

Group: Miscellaneous/System Commands

Set Only. `INIt` initializes the instrument to its factory-assigned default parameters and settings. Completion of `INIt` is signaled by event code 474, `INIT complete`.

For both GPIB and RS-232-C, the defaults are:

- `SRQMask USEr` is `OFF`; this removes the RQS icon if it was displayed

- All pending events except Power On are discarded

- All user `TEXt` is cleared from the display

- For GPIB only, `RQS` is set to `ON`

NOTE

`INIt` *has no argument.*

Refer to the *User Manual* for your instrument for a complete list of `INIt` effects.

Examples: `INI`

INPut { STO*<ui>*|*<qstring>*}

Group:	Data Transfer Commands
	INPut selects the destination for preamble and trace data sent to the instrument by the WFMPre and CURVe commands.
Link:	{ STO*<ui>* | *<qstring>* }
Arguments:	The power-on default INPut location is STO1. *<ui>* can range between 1 and 9999. *<qstring>* is a label that identifies the stored trace destination
	Query Note: INPut? always returns STO*<ui>*, even if the location was specified with a label.
Examples:	INP STO92

JITter?

Group:	Measurement Commands
	Query Only. JITter? returns the jitter calculated on the selected trace at JITLOcation. JITter? is available only when MMOde is set to STAT.
Examples:	**JIT?** JITTER 2.9669E-10, EQ

LABAbs *<link>*:*<arg>*

Group:	Label and Text Commands
	LABAbs positions the label associated with the selected trace.
Link:	PCTg: *<NRx>*
Arguments:	*<NRx>* can range between 0 and 100 percent.
	PCTg sets the horizontal position of the label as a percentage of the trace record.
Examples:	LABA PCT:50

Link: XCOord: *<NRx>*

Arguments: The range for *<NRx>* is described below.

XCOord sets the horizontal position of the label in horizontal units. The label maintains the specified position, tracking changes in the trace.

(The following range formulas assume ADJtrace PANzoom is set to OFF and the trace is acquired. Refer to the discussion on cursor positioning on page 2-120 for a method to calculate XCOord range when PANzoom is set to ON or the trace is unacquired. Refer to page 2-214 for formulas to calculate *duration*.)

The XCOord range when the selected trace record is MAin is calculated:

MAINPos to (MAINPos + *main_duration*)

The XCOord range when the selected trace record is WIN is calculated:

WINPos to (WINPos + *win_duration*)

Examples: LABA XCO:0.5

Link: YDIv: *<NRx>*

Arguments: *<NRx>* can range between −10.22 and +10.22.

YDIv sets the vertical position of the label in divisions, relative to the point specified by the XCOord link. The label maintains the specified vertical distance, tracking changes in the trace.

Examples: LABA YDI:2.85

LABel *<link>***:***<arg>*

Group: Label and Text Commands

LABel defines and deletes labels, and controls label display.

Link: DELete: { ALL | FPS [*<ui>*] | *<qstring>* | STO [*<ui>*] | TRAce [*<ui>*] }

Arguments: FPS*<ui>* can range between 1 and 9999. STO*<ui>* can range between 1 and 9999. TRAce*<ui>* can range between 1 and 8.

Set Only. DELete deletes labels for active traces, stored traces, stored settings, or ALL labels. Specifying FPS, STO, or TRAce with *<ui>* deletes the label associated with the specified argument. Specifying FPS, STO, or TRAce without *<ui>* deletes all labels associated with the argument type. Specifying *<qstring>* deletes that label. Wildcard characters are interpreted. (Refer to page 2-149 for wildcards.)

Examples: LAB DEL:TRA2

Link: DISPlay: { ON | OFF }

DISPlay controls the display of labels associated with active traces. When DISPlay is set to ON, labels are displayed. When DISPlay is set to OFF, labels are not displayed but all labels are retained. OFF is the default.

Examples: LAB DISP:ON

Link: FPS*<ui>*: *<qstring>*

Arguments: *<ui>* can range between 1 and 9999. *<qstring>* must be less than or equal to 10 characters in length.

FPS*<ui>* defines a label for a stored front panel setting.

Examples: LAB FPS1:'SETUP1'

Link: MODe: { AUTO | MANual }

MODe selects automatic or manual trace labeling. AUTO mode produces labels for new traces based on their trace descriptions. In MANual mode no labels appear on a traces unless you specifically assign them (LABel TRAce*<ui>*). LABel DISPlay must be set ON for the labels to be visible on the front panel. MANual is the default mode.

Examples: LAB MOD:AUTO

Link: `STO<ui>: <qstring>`

Arguments: `<ui>` can range between 1 and 9999. `<qstring>` must be less than or equal to 10 characters in length.

`STO<ui>` defines the label for a stored trace.

Examples: `LAB STO1:'DATA1'`

Link: `TRAce<ui>: <qstring>`

Arguments: `<ui>` can range between 1 and 8. `<qstring>` must be less than or equal to 10 characters in length.

`TRAce<ui>` defines the label for an active trace.

Examples: `LAB TRA1:'CLOCK'`

Label Wildcard Characters. For some commands that take labels, the characters **?** and ***** have a special meaning in a `<qstring>` when searching for a matching label. The **?** will match any single character. The ***** will match any number (including 0) of any character. To search for a literal ? or *, use a backslash \ in front of the ? or *.

a?c	matches	abc, axc, a2c, aEc, etc.
rep1?	matches	rep11, rep12, rep1b, etc.
rep*	matches	rep, rep65, rep1a92, repZZ, etc.
a*c	matches	abc, a3478c, axyzc, etc.
a*c	matches	a*c

Figure 2-10: Examples of Wildcard Usage

LABRel *<link>***:***<arg>*

Group:	Label and Text Commands
	Set Only. `LABRel` positions the label of the selected trace relative to its position prior to the command.
Link:	`PCTg: <NRx>`
Arguments:	See the description for `LABA PCT` for the range for *<NRx>*.
	Set Only. `PCTg` changes the horizontal position of the label, relative to its previous horizontal position, in units of percent of record length, but not exceeding the `LABAbs PCTg` range.
Examples:	`LABR PCT:50`
Link:	`XCOord: <NRx>`
Arguments:	See the description for `LABA XCO` for the range for *<NRx>*.
	Set Only. `XCOord` changes the horizontal position of the label, relative to its previous horizontal position, but not exceeding the `LABAbs XCOord` range.
Examples:	`LABR XCO:0.5`
Link:	`YDIv: <NRx>`
Arguments:	See the description for `LABA YDI` for the range for *<NRx>*.
	Set Only. `YDIv` changes the vertical position of the label relative to its previous vertical position, but not exceeding the `LABAbs YDIv` range.
Examples:	`LABR YDI:2.85`

LONgform { ON | OFF }

Group: Miscellaneous/System Commands

LONgform controls the return of the longer versions of query responses. With LONgform set to ON, queries respond with full header and link spellings; the EVENT? and RS232 VERB:ON commands return a descriptive *<qstring>* in addition to the event code. With LONgform set to OFF, query responses are in abbreviated form, and EVENT? and RS232 VERB:ON responses include only the event codes. The power-on default is LONgform ON.

Examples: LON ON

LSRJET *<link>:<arg>*

Group: External I/O Commands

LSRJET specifies printing parameters for HP Thinkjet and LaserJet printers operating in HP graphics mode. The command is identical to ALTinkjet.

NOTE

LSRJET *does not support Thinkjet and LaserJet printers operating in Epson emulation mode.*

Link: DIRection: { HORiz | VERt }

DIRection selects the printing orientation. HORiz prints rows left-to-right and top-to-bottom. VERt prints columns bottom-to-top and left-to-right.

Examples: LSRJET DIR:HOR

Link: FORMat: { DIThered | DRAft | HIRes | REDuced | SCReen }

FORMat selects the printing format. DIThered modifies print contrast for the printer. DRAft prints selected fields in reverse video. REDuced is a quarter the size of DRAft but does not show intensified regions. SCReen is a one-to-one mapping of 3-bit pixel information.

NOTE

Due to graphics imaging constraints on standard LaserJet printers, HIRes *format may not generate a suitable copy on a standard LaserJet printer*

Examples: LSRJET FORM:DRA

Link: PORt: { CENTRonics | GPIb | RS232 }

PORt specifies the output port for the printer.

Examples: LSRJET POR:RS232

Link: ? COLor

Query Only. ? COLor returns the mapping of printer colors and the instrument color index.

Examples: `LSRJET? COL`
                `LSRJET COLOR0:4095, COLOR1:0; COLOR2:3203, COLOR3:341`
                `COLOR4:1046, COLOR5:637, COLOR6:1710, COLOR7:2323`

Link: `? COLORn`

Query Only. `? COLORn` returns the mapping of printer color n and the instrument color index.

Examples: `LSRJET? COLOR`
                `LSRJET COLOR0:4095, COLOR1:0; COLOR2:3203, COLOR3:341;`
                `COLOR4:1046, COLOR5:637, COLOR6:1710, COLOR7:2323`

Link: `COLor<ui>: <NRx>`

Arguments: `<ui>` can range between 0 and 7. `<NRx>` can range between 0 and 4095.

Set Only. `COLor<ui>` assigns colors to the instrument color index as shown in Table 2-44.

Table 2-61: Default LSRJET Color Assignments

Color Index	LSRJET Color	Color Index	LSRJET Color
0	4095 (0xFFF)	4	1046 (0x416)
1	0 (0x000)	5	637 (0x27D)
2	3203 (0xC83)	6	1710 (0x6AE)
3	341 (0x155)	7	2323 (0x913)

MAINPOS *<NRx>*

Group: Time Base/Horizontal Commands

MAINPos sets the horizontal position of the Main trace record with respect to the Main trigger.

Link: *<NRx>*

Arguments: *<NRx>* can range from the minimum horizontal position to 50 ms.

The range depends upon the setting of HREfpt as shown in Table 2-62.

Table 2-62: MAINPos Range

HREfpt	lower MAINPos	upper MAINPos
LEFt	21 ns (CSA 803C) 23 ns (11801C) 45.5 ns (11801C with 1M Option)	50 ms
CEN- ter	21 ns + duration/2 (CSA 803C) 23 ns + duration/2 (11801C) 45.5 ns + duration/2 (11801C with 1M Option)	50 ms + duration/2
RIGht	21 ns + duration (CSA 803C) 23 ns + duration (11801C) 45.5 ns + duration (11801C with 1M Option)	50 ms + duration

NOTE

The minimum MAINPos *setting is dependent on internal instrument calibration. If a* MAINPos *value is selected that is less than the instrument minimum,* MAINPos *will be set to the minimum limit.*

Examples: MAINP 7.9E-8

MASK*<ui> <link>***:***<arg>*

Group: Display and Color Commands

MASK*<ui>* provides the means to create or delete any of 10 test masks available for trace analysis. The masks are polygons defined by up to 50 vertices each. Sample points falling within the masks are counted (with MASKSTAT COUNT:ON). MASK allows you to query the count for a specific mask. You must set DSYS to ON for counting to start; and set DISPlay STATistics to MASK in order to see the mask and the mask statistics on the screen. For more information on Mask Testing, refer to the *User Manual* for your instrument.

The links C.Point and D.Point let you specify the mask vertices in current scale units and divisions, respectively. The specified vertices are automatically connected to form the polygon mask. Polygons with concave shapes are not supported.

Mask testing is available with either infinite persistence or color graded displays. See the DISPlay TYPe command. Refer to the MASKStat command for mask testing statistics, starting mask count, and other related functions.

Link: C.Point: *<mpoint>*

Arguments: The range for *<mpoint>* is described below.

C.Point creates a set of XY coordinates that define a vertex of the mask. The coordinates are specified in the current horizontal and vertical scale units. The specified vertices are automatically connected to form the polygon mask. Note that the D.Point link specifies coordinates in divisions.

```
MASK<ui> C.POINT:<xcord>,<ycord>
    [ ,<xcord>,<ycord>...]
```

The *<xcord>* range is the duration of the selected trace record. You can use the left and right graticule limits as they are very close to the trace endpoints. The *<ycord>* range is the vertical displacement between the bottom and top graticule limits. The MASK?*<ui>* C.P query must be specified to read the mask definition.

The following example would define MASK2 as a rectangle two units wide by four units high.

Examples: MASK2 C.P:2.1,2,4.1,2,2.1,-2,4.1,-2

Link: DELete

DELete removes the definition for MASK*<ui>*.

Examples: MASK7 DEL

Link: D.Point: *<mpoint>*

Arguments: The range for *<mpoint>* is described below.

D.Point creates a set of XY coordinates (*<mpoint>*) that define a vertex of the mask. The X and Y coordinates are specified in divisions. The specified vertices are automatically connected to form a polygon.

```
MASK<ui> D.POINT:<xcord>,<ycord>
    [ ,<xcord>,<ycord>...]
```

The *<xcord>* and *<ycord>* ranges are based on the same coordinates used for the Histogram window and Cursors. Refer to the illustration on page 2-138 for the coordinates. The MASK?*<ui>* D.P query must be specified to read the mask definition.

The following example would define MASK2 as a rectangle two divisions wide by two divisions high.

Examples: MASK2 D.P:-2,2,2,2,2,-2,-2,-2

Link: ? NCOunt *<NRx>*

? NCOunt returns the point count for MASK*<ui>*.

Examples: **MASK2? NCO**
MASK2 NCOUNT:457

Link: ? NR.PT *<NRx>*

? NR.PT returns the number of vertices or XY coordinates that define MASK*<ui>*.

Examples: **MASK2? NR.PT**
MASK2 NR.PT:4

MASKDefine <link>

Syntax:　{ OC1 | OC3 | OC9 | OC12 | OC18 | OC24 | OC36 | OC48 |
CC.SI.64kb | CC.DO.64kb | CC.DA.64kb | CC.TI.64kb |
CC.1.5mb | CC.SY.2mb | CC.CO.2mb | CC.SY.6mb |
CC.CO.6mb | CC.8mb | CC.32mb | CC.34mb | CC.44mb |
CC.97mb | CC.ZEro.139mb | CC.ONe.139mb | DS1 | DS1.Old |
DS1C | DS2 | DS3 | DS4NA | DS4XNA | FC133 | FC133E |
FC266 | FC266E | FC531 | FC531E | FC1063 | FC1063E |
ENET1250 | STS1 | STS3 | STSX3 | STM1 | FDDI | CEPT |
USERMask | USEREye | NONe }

Group:　Display and Color Commands

MASKDefine defines standard masks. Refer to the *User Manual* for descriptions of the standard masks.

Before or after you execute a MASKDefine command, execute a DSYS ON to select the Display Mode menu, followed by a DISPlay STATistics: MASK to see the standard mask, completed by an AUTOSet STARt to invoke the special autoset algorithm for the standard mask. Also see the AUTOSet MASK:AUTO command description, which greatly affects what MASKDefine does when DSYS is ON prior to issuing the MASKDefine command.

Link:　CC.CO.2mb

CC.CO.2mb defines an ITU-T G.703 standard coaxial pair mask with a data rate of 2.048 megabits/second.

Examples:　MASKD CC.CO.2

Link:　CC.CO.6mb

CC.CO.6mb defines an ITU-T G.703 standard coaxial pair mask with a data rate of 6.312 megabits/second.

Examples:　MASKD CC.CO.6

Link:　CC.DA.64kb

CC.DA.64kb defines an ITU-T G.703 standard data pulse mask with a data rate of 64 kilobits/second.

Examples:　MASKD CC.DA.64

Link: `CC.DO.64kb`

`CC.DO.64kb` defines an ITU-T G.703 standard double pulse mask with a data rate of 64 kilobits/second.

Examples: `MASKD CC.DO.64`

Link: `CC.ONe.139mb`

`CC.ONe.139mb` defines an ITU-T G.703 standard binary 1 pulse mask with a data rate of 139.26 megabits/second.

Examples: `MASKD CC.ON`

Link: `CC.SI.64kb`

`CC.SI.64kb` defines an ITU-T G.703 standard single pulse mask with a data rate of 64 kilobits/second.

Examples: `MASKD CC.SI.64`

Link: `CC.SY.2mb`

`CC.SY.2mb` defines an ITU-T G.703 standard symmetrical pair mask with a data rate of 2.048 megabits/second.

Examples: `MASKD CC.SY.2`

Link: `CC.SY.6mb`

`CC.SY.6mb` defines an ITU-T G.703 standard symmetrical pair mask with a data rate of 6.312 megabits/second.

Examples: `MASKD CC.SY.6`

Link: `CC.TI.64kb`

`CC.TI.64kb` defines an ITU-T G.703 standard timing pulse mask with a data rate of 64 kilobits/second.

Examples: `MASKD CC.TI.64`

Link: `CC.ZEro.139mb`

`CC.ZEro.139mb` defines an ITU-T G.703 standard binary 0 pulse mask with a data rate of 139.26 megabits/second.

Examples: `MASKD CC.ZE`

Link: `CC.1.5mb`

`CC.1.5mb` defines an ITU-T G.703 standard pulse mask with a data rate of 1.544 megabits/second.

Examples: `MASKD CC.1.5`

Link: `CC.8mb`

`CC.8mb` defines an ITU-T G.703 standard pulse mask with a data rate of 8.448 megabits/second.

Examples: `MASKD CC.8`

Link: `CC.32mb`

`CC.32mb` defines an ITU-T G.703 standard pulse mask with a data rate of 32.064 megabits/second.

Examples: `MASKD CC.32`

Link: `CC.34mb`

`CC.34mb` defines an ITU-T G.703 standard pulse mask with a data rate of 34.368 megabits/second.

Examples: `MASKD CC.34`

Link: `CC.44mb`

`CC.44mb` defines an ITU-T G.703 standard pulse mask with a data rate of 44.736 megabits/second.

Examples: `MASKD CC.44`

Link: `CC.97mb`

`CC.97mb` defines an ITU-T G.703 standard pulse mask with a data rate of 97.728 megabits/second.

Examples: `MASKD CC.97`

Link: `CEPT`

`CEPT` defines an ITU-T G.703 standard CEPT mask with a data rate of 565 megabits/second.

Examples: `MASKD CEPT`

Link: `DS1`

`DS1` defines an ANSI T1.102 standard mask with a data rate of 1.544 megabits/second.

Examples: `MASKD DS1`

Link: `DS1.Old`

`DS1.Old` defines an ANSI T1.102 "Obsolete DS1" standard mask with a data rate of 1.544 megabits/second.

Examples: `MASKD DS1.O`

Link: `DS1C`

`DS1C` defines an ANSI T1.102 standard mask with a data rate of 3.152 megabits/second.

Examples: `MASKD DS1C`

Link: `DS2`

`DS2` defines an ANSI T1.102 standard mask with a data rate of 6.312 megabits/second.

Examples: `MASKD DS2`

Link: DS3

DS3 defines an ANSI T1.102 standard mask with a data rate of 44.736 megabits/second.

Examples: MASKD DS3

Link: DS4NA

DS4NA defines an ANSI T1.102 standard mask with a data rate of 139.26 megabits/second. (Zero feet of office cable between the equipment and the interface.)

Examples: MASKD DS4NA

Link: DS4XNA

DS4XNA defines an ANSI T1.102 standard mask with a data rate of 139.26 megabits/second. (225 feet of office cable between the equipment and the interface.)

Examples: MASKD DS4XNA

Link: FC133

FC133 defines an ANSI X3.230 standard mask as defined by the Fibre Channel 133 Optical standard (132.81 Mb/s).

Examples: MASKD FC133

Link: FC133E

FC133E defines an ANSI X3.230 standard mask as defined by the Fibre Channel 133 Electrical standard (132.81 Mb/s).

Examples: MASKD FC133E

Link: FC266

FC266 defines an ANSI X3.230 standard mask as defined by the Fibre Channel 266 Optical standard (265.6 Mb/s).

Examples: MASKD FC266

Link: `FC266E`

`FC266E` defines an ANSI X3.230 standard mask as defined by the Fibre Channel 266 Electrical standard (265.6 Mb/s).

Examples: `MASKD FC266E`

Link: `FC531`

`FC531` defines an ANSI X3.230 standard mask as defined by the Fibre Channel 531 Optical standard (531.2 Mb/s).

Examples: `MASKD FC531`

Link: `FC531E`

`FC531E` defines an ANSI X3.230 standard mask as defined by the Fibre Channel 266 Electrical standard (531.25 Mb/s).

Examples: `MASKD FC531E`

Link: `FC1063`

`FC1063` defines an ANSI X3.230 standard mask as defined by the Fibre Channel 1063 Optical standard (1062.5 Mb/s).

Examples: `MASKD FC1063`

Link: `FC1063E`

`FC1063E` defines an ANSI X3.230 standard mask as defined by the Fibre Channel 266 Electrical standard (1062.5 Mb/s).

Examples: `MASKD FC1063E`

Link: `FDDI`

`FDDI` defines an ANSI X3.166 FDDI Halt standard mask with a data rate of 125 megabits second. For FDDI, AUTOSet is unique because the standard specifies that the width of a Halt signal is 40 nanoseconds (within ± 0.5 nanosecond). The width of the mask is adjusted by up to ± 0.5 nanosecond to match the rising and falling edge of the Halt signal.

Examples: `MASKD FDDI`

Link: `ENET1250`

`ENET1250` defines an IEEE 802.3 standard mask with a data rate of 1.25 Gigabits per second.

Examples: `MASKD GBETHERNET`

Link: `NONe`

`NONe` is normally not used in a set command. In the query `MASKD?`, it is returned if there is no standard mask (that is, just user masks). `MASKD NONe` used in a set command has no affect if you have user masks; if you have standard masks, they are then deleted.

Examples: `MASKD NON`

Link: `OC1`

`OC1` defines an SONET/SDH optical mask with a data rate of 51.84 megabits/second.

Examples: `MASKD OC1`

Link: `OC3`

`OC3` defines an SONET/SDH optical mask with a data rate of 155.52 megabits/second.

Examples: `MASKD OC3`

Link: `OC9`

`OC9` defines an SONET/SDH optical mask with a data rate of 466.56 megabits/second.

Examples: `MASKD OC9`

Link: `OC12`

`OC12` defines an SONET/SDH optical mask with a data rate of 622.08 megabits/second.

Examples: `MASKD OC12`

Link: `OC18`

`OC18` defines an SONET/SDH optical mask with a data rate of 933.12 megabits/second.

Examples: `MASKD OC18`

Link: `OC24`

`OC24` defines an SONET/SDH optical mask with a data rate of 1244.2 megabits/second.

Examples: `MASKD OC24`

Link: `OC36`

`OC36` defines an SONET/SDH optical mask with a data rate of 1866.2 megabits/second.

Examples: `MASKD OC36`

Link: `OC48`

`OC48` defines an SONET/SDH optical mask with a data rate of 2488.3 megabits/second.

Examples: `MASKD OC48`

Link: `STM1`

`STM1` defines an SONET/SDH optical mask with a data rate of 155.52 megabits/second.

Examples: `MASKD STM1`

Link: `STS1`

`STS1` defines an ANSI T1.102 standard mask with a data rate of 51.84 megabits/second.

Examples: `MASKD STS1`

Link: `STS3`

`STS3` defines an ANSI T1.102 standard mask with a data rate of 155.52 megabits/second. (Zero feet of office cable between equipment and interface.)

Examples: `MASKD STS3`

Link: `STSX3`

`STSX3` defines an ANSI T1.102 standard mask with a data rate of 155.52 megabits/second. (225 feet of office cable between equipment and interface.)

Examples: `MASKD STSX3`

Link: `USEREye`

`USEREye` recalls a "standard' eye mask, which has a programmable data rate and user-settable shape. Refer to the command descriptions for `USER-Eye`.

Examples: `MASKD USEREye`

Link: `USERMask`

`USERMask` is normally not used in a set command. In the query `MASKD?`, it is returned if there is no standard mask (that is, just user masks). `MASKD USERMask` used in a set command has no affect if you have user masks; if you have standard masks, they are then deleted.

Examples: `MASKD USERM`

MASKStat *<link>:<arg>*

Group: Display and Color Commands

MASKStat starts mask testing and returns mask count statistics and status.

Link: CLEar

CLEar clears all mask counts and removes all trace data from the display. Then all acquisitions are restarted and mask testing resumes. CLEar is equivalent to the CLEar ALLTrace command.

Examples: MASKS CLE

Link: COUnt: { ON | OFF }

COUnt starts counting sample point hits within all masks on all displayed traces when set ON. When COUnt is set to OFF new hits within masks are not counted but existing count statistics are retained and may be queried. Mask counting started on a color graded display calculates the counts from the accumulated pixel binary data.

NOTE

DSYS *must be set* ON *(Display Modes major menu selected) before counting can begin.* DISPlay STATIstics *must be set to* MASK *in order to see the mask and mask statistics on the screen; however, counting continues even with* DISPlay STATIstics *set to* HISTogram. *Leaving the Display Modes major menu terminates mask counting functions.*

Examples: MASKS COU:ON

Link: ? NWFm *<NRx>*

? NWFm returns the number of traces processed to produce the count returned by the TOTal query.

Examples: **MASKS? NWF**
MASKSTAT NWFM:234

Link: `? TOTal <NRx>`

`? TOTal` returns the number of sample points that have fallen within all masks on all displayed traces. Sample points that fall where two masks overlap are counted only once.

Examples: **MASKS? TOT**
MASKSTAT TOTAL:735

MAX?

Group: Measurement Commands

Query Only. `MAX?` returns the maximum amplitude (most positive peak voltage) of the selected trace, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **MAX?**
MAX 5.04E-1,EQ

MAXTranum?

Group: Trace and Settings Commands

Query Only. `MAXTranum?` returns the largest acceptable trace number for the current configuration. For the CSA 803C, the value returned is always 8. For the 11801C, the value returned is always equal to or greater than 40. Traces greater than 8, called Saved Trace Descriptions, are not visible on the screen; refer to the *User Manual* for more information.

Examples: **MAXT?**
MAXTRANUM 8

MEAN?

Group: Measurement Commands

Query Only. `MEAN?` returns the average amplitude (arithmetic mean voltage) of the selected trace, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **MEAN?**
MEAN 2.212E-1,EQ

MEAS [*<ui>* [**TO** *<ui>*]] **?**

Group: Measurement Commands

Query Only. MEAS? executes the measurements (*<meas>*) in the current measurement list (MSLIst) of the specified trace or for the range of traces or all traces. MEAS? returns a scalar value followed by an accuracy qualifier (*<qual>*) for each measurement in the list. MEAS? returns EMPty if MSLIst contains no measurements. Note that you do not need to set MSYS to ON before using the MEAS? command.

```
{ MEAS  <ui>  {  <meas>:<NR3>,<qual>
    [ , {  <meas>:<NR3>,<qual>...} ] } } | EMPty
```

The *<qual>* accuracy qualifier indicates whether or not the underlying trace data contain null, overrange, or underrange values.

The measurement *<qual>* accuracy qualifiers are defined in Table 2-63.

Table 2-63: Software Measurement Accuracy Qualifiers

<qual>	Meaning
EQ	Measurement was successfully made
ER	Error
NF	Measurement not found
OR	Measurement was out of range

The UN qualifier is returned for the following conditions:

- Attempted any timing measurement when the measurement zone of the selected trace contained null (unacquired) values.

- Attempted a FALltime?, FREq?, PERiod?, RISetime?, WIDth?, or an area/energy measurement when the trace description for the selected trace is enveloped or contains enveloped components.

- Attempted a MEAN? or RMS? measurement when DAInt was set to SINgle and the trace description of the selected trace was enveloped or contained enveloped components.

The ER qualifier is returned for the following conditions:

- Attempted DUTy?, FREq?, or PERiod? measurement and no period was found within the specified measurement zone.

- Attempted a MEAN?, RMS?, YTPls_area?, YTMns_area?, or YTEnergy? measurement when DAInt was SINgle and no period was found within the specified measurement zone.

- Attempted a CROss? measurement and no transition of the specified slope was found.

- Attempted a `CROss?` measurement and `REFLEvel` did not fall within the computed `MAX` and `MIN` of the specified measurement zone.

- Attempted a `RISetime?` measurement and the measurement system could not compute a valid `PROXimal` time, followed by a valid `DISTal` time, within the specified measurement zone.

- Attempted a `FALltime?` measurement and the measurement system could not compute a valid `DISTal` time followed by a valid `PROXimal` time, within the specified measurement zone.

- Attempted a `WIDth?` measurement and two `MESial` crossings of opposite slope could not be found within the specified measurement zone.

- Attempted any measurement when the selected trace was an XY trace or in a display persistence mode.

- Attempted any measurement when no traces were displayed.

Examples: `MEAS?`
`MEAS1 MEAN:7.3333E-4,EQ,`
`CROSS:7.6685E-4,EQ`

<meas>?

Group: Measurement Commands

Query Only. *<meas>*? is shorthand for a query of any of the measurements listed below. Querying a specific measurement executes the measurement and returns its value followed by an accuracy qualifier. (Refer to the MEAS? command for the list of qualifiers.) The *<meas>* measurements are listed by function in Table 2-64.

Table 2-64: *<meas>* Measurement Types

Amplitude	Timing	Area/Energy	Frequency Domain
AMPlitude	CROss	YTEnergy	SFRequency
EXTInction	DUTy	YTMns_area	SMAgnitude
MAX	FALltime	YTPls_area	THD
MEAN	FREq		
MID	JITter		
MIN	PDElay		
NOIse	PERiod		
OVErshoot	PHAse		
P-P	RISetime		
RMS	WIDth		
SNRatio			
UNDershoot			

All of the Timing measurements except DUTy, PHAse, and JITter can be performed in the Hardware mode. Refer to each measurement entry for specific information.

Examples: **MEAN?**
MEAN 7.3333E-4,EQ

MID?

Group: Measurement Commands

Query Only. `MID?` returns the amplitude midpoint of the selected trace, which is halfway between the maximum amplitude and the minimum amplitude, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: `MID?`
`MID 2.2E–1,EQ`

MIN?

Group: Measurement Commands

Query Only. `MIN?` returns the minimum amplitude (most negative peak voltage) of the selected trace, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: `MIN?`
`MIN –6.398E–2,EQ`

MPAram*<ui> <link>:<arg>*

Group: Measurement Commands

Defines the parameters for making measurements on the specified trace.

Link: `BASeline: <NRx>`

Arguments: `<NRx>` can be any legal value.

`BASeline` sets the vertical baseline level for measurement on the specified trace. This value is used only when `MTRack` (measurement tracking) is set to `OFF` or `TOPline`.

`BASeline` is ignored when `MTRack` is set `ON` or to `BASeline`.

Examples: `MPA6 BAS:–8.5E–1`

Link: DAInt: { WHOle | SINgle }

DAInt sets the data measurement interval to a SINgle period of the trace or to the WHOle measurement interval set by the LMZone and RMZone links. DAInt is not used in the hardware measurement mode.

DAInt affects the MEAN?, RMS?, YTEnergy?, YTMns_area?, and YTPls_area? measurements. These measurements return an ER qualifier if DAInt is set to SINgle and no period can be found.

NOTE

The measurement qualifiers are defined on page 2-168.

Examples: MPA6 DAI:WHO

Link: DISPErsion: { PP | RMSDev }

DISPErsion selects the display of JITter and NOIse calculation results as peak-to-peak or RMS deviation (Standard Deviation) values. If MMOde is set to HW or SW, DISPErsion is ignored because JITter and NOIse measurements cannot be performed in HW or SW modes.

Examples: MPA6 DISPE:PP

Link: DISTal: *<NRx>*

Arguments: The range for *<NRx>* is described below.

DISTal sets the distal (furthest from point of origin) level used by RISetime and FALltime measurements.

The DISTal range depends on the current argument to MLEvelmode as shown in Table 2-65. When MLEvelmode is set to RELative, the DISTal range is a percentage of the difference between the TOPLine and BASeline. When MLEvelmode is set to ABSolute, the DISTal range for hardware and software modes are in vertical units of the selected trace.

Examples: MPA6 DIST:85

Table 2-65: Parameter Ranges for Distal

MMOde:	**MLE**velmode: **REL**ative	**MLE**velmode: **ABS**olute	**MLE**velmode: **TOPD**elta	**MLE**velmode: **BASED**elta
SW or STAT	0 % to 100 %	1E+15 to −1E+15	1E+15 to −1E+15	1E+15 to −1E+15
HW	0 % to 100 %	−2 V to +2 V	−2 V to +2 V	−2 V to +2 V

> *Link:* `ERMethod: { NORmal | FDDI | SONet }`
>
> `ERMethod` sets the extinction ratio method. `NORmal` is equal to `TOPline/BASeline`. `FDDI` is equal to 100 times `BASeline/TOPline` in percentage units. `SONet` is equal to 10 times $\log_{10}$ of `TOPline/BASeline` in decibel units.

> *Examples:* `MPA6 ERM:NOR`

> *Link:* `FROMMLevelmode: { ABSolute | RELative | BASEDelta | TOP-Delta }`
>
> `FROMMLevelmode` sets the level mode on the "from" trace for `PDELay` and `PHAse` measurements. `FROMMLevelmode` also sets the level mode for all other measurement commands. `FROMMLevelmode` affects `DISTal`, `MESial`, `REFLEvel`, and `PROXimal` measurements, but not `REFMesial`.
>
> `ABSOlute` makes the `DISTal`, `MESial`, `REFLEvel`, and `PROXimal` measurement ranges be absolute values scaled in vertical units (typically volts) of the selected waveform. Measurement parameter tracking (`MTRack`) is not active when `ABSolute` mode is selected. (That is, the baseline and topline don't track when `ABSolute` mode is selected.)
>
> `BASEDelta` makes the `DISTal`, `MESial`, `REFLEvel`, and `PROXimal` measurement ranges be "delta" values which are added to the current `BASeline` value to give the value used for measurements. The "delta" values are scaled in vertical units.
>
> `RELative` makes the `DISTal`, `MESial`, `REFLEvel`, and `PROXimal` measurement ranges be a percentage of the difference between the current `TOPline` and `BASeline` values.
>
> `TOPDelta` makes the `DISTal`, `MESial`, `REFLEvel`, and `PROXimal` measurement ranges be "delta" values which are added to the current `TOPline` value to give the value used for measurements. The "delta" values are scaled in vertical units.

> *Examples:* `MPA6 FROMML:ABS`

Link: FROMMTrack: { ON | OFF | BASeline | TOPline | BOTh }

FROMMTrack controls measurement tracking (continuous building of histograms) for the "from" trace for the PDELay and PHAse measurements. FROMMTrack also controls the measurement tracking for all other measurement commands.

When BASeline is selected, the baseline is determined by the instrument, you set the topline value.

When BOTh is specified, the instrument determines both the BASeline and TOPline values.

When OFF is selected, you set both the BASeline and TOPline values.

BOTh may be substituted for ON when FROMMTrack is used to set measurement tracking, but the query FROMMTrack? will return ON.

When TOPline is selected, the topline is determined by the instrument, you set the baseline value.

In ABSolute mode, FROMMTrack is effectively treated as if OFF were selected. (That is, tracking is not used to set the topline and baseline, regardless of how FROMMTrack is set.)

Examples: MPA6 FROMMT:OFF

Link: HNUmber: *<ui>*

Arguments: *<ui>* can range between 1 and 1000.

HNUmber selects the harmonic number on which the SMAgnitude and SFRequency measurements are made when SMOde is set to HARmonic.

Examples: MPA6 HNU:3

Link: JITTLOcation: { CROss | MESial }

JITTLOcation selects the location of the jitter measurement on the trace. Jitter can be measured at a signal level crossing or at the middle (mesial) reference level of a trace. If MMOde is set to HW or SW, JITTLOcation is ignored because JITter and NOIse measurements cannot be performed in HW or SW modes.

Examples: MPA6 JITTLO:CRO

Link: ? JITTLEvel: *<NRx>*

Query Only. ? JITTLEvel returns the value of the jitter measurement. ? JITTLEvel updates only if JITter is in the measurement list for the trace, the trace is selected, or MEAS? is queried. If MMOde is set to HW or SW, ? JITTLEvel returns a value that should be ignored.

Examples: **MPA6? JITTLE**
MPARAM6 JITTLEVEL:5.26997E−1

Link: ? JITT.histpt: *<NRx>*

Query Only. ? JITT.histpt returns the number of histogram points used on the trace to calculate jitter. ? JITT.histpt updates only if JITter is in the measurement list for the trace, the trace is selected, or MEAS? is queried. If MMOde is set to HW or SW, ? JITT.histpt returns a value that should be ignored.

Examples: **MPA6? JITT.**
MPARAM6 JITT.HISTPT:5408

Link: LMZone: *<NRx>*

Arguments: *<NRx>* can range between 0 and 100 percent.

LMZone sets the left measurement zone limit as a percentage of the trace record.

Examples: MPA6 LMZ:0

      Link: MESial: <*NRx*>

Arguments: The range for <*NRx*> is described below.

MESial sets the mesial (middle) reference level used to determine the endpoint of the trace period for DUTy, FREq, MEAN, PDElay, PERiod, PHAse, RMS, and WIDth measurements; and when data interval (DAInt) is set to SINgle, for YTEnergy, YTMns_area, and YTPls_area measurements. See Table 2-66.

Table 2-66: Parameter Range for Mesial

MMOde:	**MLE**velmode: **REL**ative	**MLE**velmode: **ABS**olute	**MLE**velmode: **TOPD**elta	**MLE**velmode: **BASED**elta
SW or STAT	0 % to 100 %	1E+15 to −1E+15	1E+15 to −1E+15	1E+15 to −1E+15
HW	0 % to 100 %	−2 V to+2 V	−2 V to+2 V	−2 V to+2 V

Examples: MPA6 MES:50

      Link: MFIltering: <*NRx*>

Arguments: <*NRx*> can range between 1 and 7.

MFIltering sets the filtering constant for hardware measurements. MFIltering is the number of successive samples that must occur after crossing the threshold level (MESial, DIStal, etc.) before the transition is valid (defaults to 3). If less than MFIltering number of samples occurs before the trace recrosses the MESial level the original transition is not valid. This removes the effects of noise on hardware timing measurements.

Examples: MPA6 MFI:4

Link: `MLEvelmode: { ABSolute | RELative | BASEDelta | TOPDelta }`

`MLEvelmode` controls how values are determined for `DISTal`, `MESial`, `REFLEvel`, `REFMesial`, and `PROXimal` commands. `MLEvelmode` is the equivalent to sending the `FROMMLevelmode` and `REFMLevelmode` commands with the same value set for each.

`ABSOlute` makes the `DISTal`, `MESial`, `REFLEvel`, `REFMesial`, and `PROXimal` ranges absolute values scaled in vertical units (typically volts) of the selected waveform. Measurement parameter tracking (`MTRack`) is not active when `ABSolute` mode is selected. (That is, the baseline and topline don't track when `ABSolute` mode is selected.)

`BASEDelta` makes `DISTal`, `MESial`, `REFLEvel`, `REFMesial`, and `PROXimal` ranges "delta" values which are added to the current `BASeline` value to give the `DISTal`, `MESial`, `REFLEvel`, `REFMesial`, or `PROximal` value used for measurements. The "delta" values are scaled in vertical units.

`RELative` makes `DISTal`, `MESial`, `REFLEvel`, `REFMesial`, and `PROXimal` ranges a percentage of the difference between the current `TOPline` and `BASeline` values.

`TOPDelta` makes `DISTal`, `MESial`, `REFLEvel`, `REFMesial`, and `PROXimal` ranges "delta" values which are added to the current `TOPline` value to give the `DISTAl`, `MESial` or `PROXimal` value used for measurements. The "delta" values are scaled in vertical units.

Table 2-67 shows some examples, assuming `BASeline` is 0 V and `TOPline` is 10 V.

Table 2-67: MLEvelmode Examples

MLEvelmode Arg:	Desired Parameter:	MPAram Link to Use
RELative	MESial 4.5 V	MESial 45
ABSOlute	MESial 4.5 V	MESial 4.5
TOPDelta	PROximal 1.1 V	PROximal −8.9
BASEDelta	DISTal 8.7 V	DISTal 8.7

Query Note: `? MLEvelmode` always returns the same value as the `? FROMMLevelmode` query.

Examples: `MPA6 MLE:ABSO`

Link: MMode: { HW | SW | STAT }

MMOde selects either the hardware, software, or software statistical measure-ment mode. Software mode allows any measurements on the selected trace record. Hardware mode allows only timing measurements taken with preci-sion timing circuits. The software statistical mode, for measuring random data, calculates measurement parameters using histograms.

Examples: MPA6 MMO:STAT

Link: MSLOpe: { PLUs | MINUs }

MSLOpe sets the crossing slope that begins CROss and PDElay hardware measurements.

Examples: MPA6 MSLO:PLU

Link: MTRack: { ON | OFF | BASeline | TOPline | BOTh }

MTRack controls measurement tracking (continuous building of histograms).

When BASeline is selected, the baseline is determined by the instrument, you set the topline value.

When BOTh is specified, the instrument determines both the BASeline and TOPline values.

When OFF is selected, you set both the BASeline and TOPline values.

BOTh may be substituted for ON when MTRack is used to set measurement tracking, but the query MTRack? will return ON.

When TOPline is selected, the topline is determined by the instrument, you set the baseline value.

In ABSolute mode, MTRack is effectively treated as if OFF were selected. (That is, tracking is not used to set the topline and baseline, regardless of how MTRack is set.)

Query Note: ? MTRack always returns the same value as the ? FROMMTrack query.

Examples: MPA6 MTR:OFF

Link: MTRANs: *<NRx>*

Arguments: *<NRx>* can range between 1 and 15.

MTRANs sets the number of the transition on which hardware measurements will begin. The transition is defined by the SLOpe, level (e.g., MESial), and MFIltering settings.

Examples: MPA6 MTRAN:3

Link: NOISLocation: { TOPline | BASeline }

NOISLocation selects the measurement of noise at the topline or baseline of the trace. If MMOde is set to HW or SW, NOISLocation is ignored because NOIse cannot be calculated in HW or SW modes.

Examples: MPA6 NOISL:TOP

Link: ? NOIS.histpt: *<NRx>*

Query Only. ? NOIS.histpt returns the number of histogram points used on the trace to calculate noise. ? NOIS.histpt updates only if NOIse is in the measurement list for the trace, the trace is selected, or MEAS? is queried. If MMOde is set to HW or SW, ? NOIS.histpt returns a value which should be ignored.

Examples: MPA6 NOIS.
MPARAM6 NOIS.HISTPT:2291

Link: PINdex: *<ui>*

Arguments: *<ui>* can range between 1 and 1000.

PINdex selects the peak index on which the SMAgnitude and SFRequency measurements are made when SMOde is set to PEAK.

Examples: MPA6 PIN:530

Link: PROXimal: *<NRx>*

Arguments: See the description for DISTal for the range for *<NRx>*.

PROXimal sets the proximal (near to origin) level for RISetime and FALl-time measurements.

PROXimal range depends on the current argument to MLEvelmode. When MLEvelmode is RELative, the range is a percentage of the difference between the TOPLine and BASeline. When MLEvelmode is ABSolute, the range is in vertical units of the selected trace. When MLEvelmode is BASEDelta or TOPDelta, the value is an absolute value scaled in vertical units.

Examples: MPA6 PROX:5

Link: REFBaseline: *<NRx>*

Arguments: *<NRx>* can be any legal value.

REFBaseline sets the vertical baseline level on the reference trace when MTRack (measurement tracking) is set to OFF or to TOPline. This is used for PDELay and PHAse measurements.

REFbaseline is ignored when REFMTrack is set to ON or to BASeline.

Examples: MPA6 REFB:-3.1415E-1

Link: REFFiltering: *<NRx>*

Arguments: *<NRx>* can range between 1 and 7.

REFFiltering sets the filtering constant on the reference trace for hardware measurements. REFFiltering is the number of successive samples that must cross the threshold level (REFLEvel) before a transition is valid (defaults to 3). This removes the effects of noise on measurements.

Examples: MPA6 REFF:5

Link: REFLEvel: *<NRx>*

Arguments: See the description for DISTal for the range for *<NRx>*.

REFLEvel sets the reference level for CROss, YTPls_area, SFRequency, SMAgnitude, and YTMns_area measurements.

Examples: MPA6 REFLE:55

Link: `REFMLevelmode: { ABSolute | RELative | BASEDelta | TOP-Delta }`

`REFMLevelmode` sets the level mode on the reference trace for `PDELay` and `PHASe` measurements. `REFMLevelmode` affects the `REFMesial`.

`ABSOlute` makes the `REFMesial` measurement range be absolute values scaled in vertical units (typically volts) of the selected waveform. Measurement parameter tracking (`MTRack`) is not active when `ABSolute` mode is selected. (That is, the baseline and topline don't track when `ABSolute` mode is selected.)

`BASEDelta` makes the `REFMesial` measurement range be "delta" values which are added to the current `BASeline` value to give the value used for measurements. The "delta" values are scaled in vertical units.

`RELative` makes the `REFMesial` measurement range be a percentage of the difference between the current `TOPline` and `BASeline` values.

`TOPDelta` makes the `REFMesial` measurement range be "delta" values which are added to the current `TOPline` value to give the value used for measurements. The "delta" values are scaled in vertical units.

Examples: `MPA6 REFML:ABS`

Link: `REFLMzone: <NRx>`

Arguments: `<NRx>` can range between 0 and 100 percent.

`REFLMzone` sets the left measurement zone on the reference trace for `PDELay` and `PHASe` measurements.

Examples: `MPA6 REFLM:10`

Link: `REFMesial: <NRx>`

Arguments: See the description for `DISTal` for the range for `<NRx>`.

`REFMesial` sets the mesial level on the reference trace for `PDELay` and `PHASe` measurements. See the descriptions for `REFMLevelmode` and `REFMTrack`.

Examples: `MPA6 REFM:50`

Link: REFMTrack: { ON | OFF | BASeline | TOPline | BOTh }

REFMTrack controls measurement tracking (continuous building of histo-grams) for the reference trace for the PDELay and PHAse measurements.

When BASeline is selected, the refbaseline is determined by the instru-ment, you set the reftopline value.

When BOTh is specified, the instrument determines both the refbaseline and reftopline values.

When OFF is selected, you set both the refbaseline and reftopline values.

BOTh may be substituted for ON when REFMTrack is used to set measure-ment tracking, but the query REFMTrack? will return ON.

When TOPline is selected, the reftopline is determined by the instrument, you set the refbaseline value.

In ABSolute mode, REFMTrack is effectively treated as if OFF were se-lected. (That is, tracking is not used to set the reftopline and refbaseline, regardless of how REFMTrack is set.)

Examples: MPA6 REFMT:OFF

Link: REFRzone: *<NRx>*

Arguments: *<NRx>* can range between 0 and 100 percent.

REFRmzone sets the right measurement zone on the reference trace for PDELay and PHAse measurements.

Examples: MPA6 REFR:100

Link: REFSLope: { PLUs | MINUs }

REFSLope is a hardware measurement mode parameter that sets the reference trace slope used to determine the PDELay measurement end-point.

Examples: MPA6 REFSL:PLU

Link:	REFSNratio: *\<NRx\>*
Arguments:	*\<NRx\>* can range between 1 and 99.
	REFSNratio sets the signal-to-noise ratio on the reference trace to provide a noise rejection band centered on the REFMesial level. This is used to qualify transitions for SW mode measurements. The reciprocal of the number selected is the fraction of the REFTOpline-to-REFBASeline distance the noise rejection band extends above and below the REFMesial level.
Examples:	MPA6 REFSN:33
Link:	REFTOpline: *\<NRx\>*
Arguments:	*\<NRx\>* can be any legal value.
	REFTOpline sets the topline value of the reference trace for the PDElay and PHAse measurements. This value is used only when tracking is off for the topline (REFMTrack set to OFF or BASeline).
Examples:	MPA6 REFTO:3.1415E–1
Link:	REFTRace: TRAce*\<ui\>*
Arguments:	*\<ui\>* can range between 1 and MAXTranum in HW mode and between 1 and 8 in SW or STAT mode.
	REFTRace selects the reference trace which provides the endpoint for PDElay and PHAse measurements. See page 2-167 for the value of MAX-Tranum.
Examples:	MPA6 REFTR:TRA20
Link:	REFXsition: *\<NRx\>*
Arguments:	*\<NRx\>* can range between 1 and 15.
	REFXsition is a hardware measurement mode parameter that sets the number of the transition on the reference trace on which the PDElay measurement will end.
Examples:	MPA6 REFX:6

Link: RMZone: *<NRx>*

Arguments: *<NRx>* can range between 0 and 100 percent.

RMZone sets the right measurement zone limiter as a percentage of the trace record in SW mode only.

Examples: MPA6 RMZ:75

Link: SMOde: { PEAK | HARMonic }

SMOde selects whether the SMAgnitude or SFRequency measurements are made on a selected harmonic or peak.

Examples: MPA6 SMO:HARM

Link: SNRatio: *<NRx>*

Arguments: *<NRx>* can range between 1 and 99.

SNRatio sets the signal-to-noise ratio to provide a noise rejection band centered on the MESial level. This is used qualify transitions for SW mode measurements. The reciprocal of the number selected is the fraction of the TOPline-to-BASeline distance the noise rejection band extends above and below the MESial level.

Examples: MPA6 SNR:33

Link: TOPline: *<NRx>*

Arguments: *<NRx>* can be any legal value.

TOPline sets the vertical topline level for measurement on the specified trace. This value is used only when MTRack (measurement tracking) is set to OFF or BASEline.

TOPline is ignored when MTRack is set ON or to TOPline.

Examples: MPA6 TOP:2.0

Query Note: MPAram? returns the same parameters as MPAram*<ui>*? for all defined traces in numeric trace order. MPAram*<ui>*? returns the links and arguments for the specified trace in the following order:

```
MPARAM<ui> MMODE:<arg>, MLEVELMODE:<arg>,
   MTRACK:<arg>, BASELINE:<NR3>, TOPLINE:<NR3>,
   MESIAL:<NR3>, PROXIMAL:<NR3>, DISTAL:<NR3>,
   DAINT:<arg>, SNRATIO:<NR3>, LMZONE:<NR1>,
   RMZONE:<NR1>, MSLOPE:<arg>,
   MFILTERING:<NR1>, MTRANS:<NR1>, JITTLOCATION:<arg>,
   NOISLOCATION:<arg>, DISPERSION:<arg>,
   HNUMBER: <NR1>, PINDEX:<NR1>, SMODE:<arg>,
   ERMETHOD:<arg>, JITTLEVEL:<NR3>, JITT.HISTPT:<NR1>,
   NOIS.HISTPT:<NR1>, REFTRACE:<arg>,
   REFBASELINE:<NR3>, REFTOPLINE:<NR3>,
   REFMESIAL:<NR3>, REFSNRATIO:<NR3>, REFLMZONE:<NR3>,
   REFRMZONE:<NR3>, REFSLOPE:<arg>,
   REFFILTERING:<NR3>, REFXSITION:<NR1>,
   REFLEVEL:<NR3>
```

MSLIst*<ui> <meas>*[, *<meas>*...]

Group: Measurement Commands

Selects the *<meas>* (up to six) executed once at a MEAS*<ui>*? query or continuously executed while the Measurement menu is displayed, for the specified trace. Range of *<ui>* is 1 to MAXTranum.

Examples: MSLI8 PP,FRE,WID,PER

Query Note: MSLIst*<ui>*? returns the *<meas>* list for the specified trace, or EMPty if no *<meas>* have been selected. MSLIst? returns the *<meas>* list for all defined traces in numeric order.

MSNum [*<ui>*] **?**

Group: Measurement Commands

Query Only. Returns the number of items in the current MSLIst, for the specified trace or for all traces. If MSLIst is EMPty, MSNum? returns 0.

Examples: **MSN1?**
MSNUM1 4

MSYs { ON | OFF }

Group: Measurement Commands

MSYs sets the measurement system ON or OFF at the front panel display. In effect, MSYs ON presses the front panel **Measure** button. Whether MSYs is ON or OFF has no effect on measurements taken with MEAS? or if you query a specific measurement.

Set MSYs to ON when you need to use the front panel in conjunction with remote commands (for example, semi-automatic ATE applications). Set MSYs to OFF for faster remote system throughput. OFF selects the waveform major menu.

Examples: MSYS OFF

NAVg *<NRx>*

Group: Acquisition Commands

NAVg sets the number of trace samples to be averaged when averaging is enabled either in the trace description (refer to the TRAce command) or as an acquisition condition (refer to the CONDacq command).

Link: *<NRx>*

Arguments: *<NRx>* can range between 2 and 4096.

Examples: NAV 50

NENV *<NRx>*

Group: Trace and Settings Commands

NENV sets the number of trace samples to be enveloped when enveloping is enabled as an acquisition condition (refer to the CONDacq command).

Link: *<NRx>*

Arguments: *<NRx>* can range between 2 and 4096.

Examples: NEN 300

NGRAded *<NRx>*

Group: Acquisition Commands

NGRAded sets an overflow number that determines how many pixel bins on a color graded display must exceed their maximum count (65535) to stop conditional acquisition. The NGRAded value can equal either the number of pixel bins that have overflowed or the number of overflows for a particular bin or both combined. Once a pixel bin has overflowed, all subsequent hits on it add to the current overflow count. The overflow count is compared to the value NGRAded to determine when to halt conditional acquisition.

Refer to the CONDacq TYPe:GRADed command to set this conditional acquire mode.

Link: *<NRx>*

Arguments: *<NRx>* can range between 1 and 65535.

Examples: NGRA 40

NHISt.pt *\<NRx\>*

Group:	Acquisition Commands

NHISt.pt sets the number of points that must be acquired in a histogram to stop conditional acquisition (refer to the CONDacq TYPe:HIST.pt command).

Link:	*\<NRx\>*

Arguments:	*\<NRx\>* can range between 1 and $2^{32} - 1$.

Examples:	NHIS 330

NMAsk *\<NRx\>*

Group:	Status and Event Commands

NMAsk sets the number of mask hits that must be acquired to stop the conditional acquisition set with CONDacq TYPe:MASK *\<ui\>*.

Link:	*\<NRx\>*

Arguments:	*\<NRx\>* can range between 1 and $2^{32} - 1$.

Examples:	NMA5

NOIse**?**

Group:	Measurement Commands

Query Only. NOIse? returns the noise measurement calculated on the selected trace at a point half way between the left and right crossings (or half way between the left and right measurement zones). NOIse? is available only when MMOde is set to STAT.

Examples:	**NOI?**
	NOISE 1.211E−2, EQ

NVRam**?**

Group: Trace and Settings Commands

Query Only. NVRam? returns the number of bytes, in *<NR1>* form, of unallo-cated nonvolatile RAM (NVRAM) available for storing front panel settings and stored traces.

Examples: NVR?
NVRAM 104723

NWAVfrm *<NRx>*

Group: Acquisition Commands

NWAVfrm sets the number of traces that must be processed into histogram, color graded, and/or mask data to stop conditional acquisition (refer to the CONDacq TYPe:WAVfrm command).

Link: *<NRx>*

Arguments: *<NRx>* can range between 1 and $2^{32}-1$.

Examples: NWAV 1000

OUTput *<arg>*

Group: Data Transfer Commands

OUTput selects the source of data returned by WFMPre?, CURVe? or WAVfrm? queries. The source can be a stored trace (STO*<ui>*) or a dis-played trace (TRAce*<ui>*); either source can be identified with a label (*<qstring>*). The power-on default is STO1.

Link: { ALLSTO | ALLTrace }

ALLSTO returns data for all existing stored traces. ALLTrace returns data for all existing displayed traces plus traces generated from saved trace descrip-tions.

Examples: OUT ALLSTO

Link: STO*<ui>*

Arguments: *<ui>* can range between 1 and 9999.

STO*<ui>* identifies the data source as the specified stored trace.

Examples: OUT STO55

Link: STO*<ui>*TOSTO*<ui>*

Arguments: *<ui>* can range between 1 and 9999.

STO*<ui>*TOSTO*<ui>* returns data for the specified range of stored traces. Unassigned numbers within the range are ignored.

Examples: OUT STO51TOSTO60

Link: TRAce*<ui>*

Arguments: *<ui>* can range between 1 and MAXTranum.

TRAce returns data for the specified trace. See page 2-167 for the value of MAXTranum. When CURVe? is performed and *<ui>* is greater than 8 (in the 11801C only), then that trace is acquired at that time.

Examples: OUT TRA40

Link: TRAce*<ui>*TOTRAce*<ui>*

Arguments: *<ui>* can range between 1 and MAXTranum.

TRAce*<ui>*TOTRAce*<ui>* returns data for the specified range of traces. Unassigned numbers within the range are ignored.

Examples: OUT TRA2TOTRA15

Link: *<qstring>*

<qstring> identifies the data source as the specified labeled trace. If the label matches both a stored trace and a displayed trace, the displayed trace is used by OUTput.

Examples: OUT 'CTRL44'

OVErshoot?

Group: Measurement Commands

Query Only. `OVErshoot?` returns the difference between the maximum signal amplitude and the `TOPLine` value for the selected trace. It is given as a percentage of the difference between the `TOPLine` and `BASeline` values and is followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **OVE?**
OVERSHOOT 6.221E-1,EQ

PDElay?

Group: Measurement Commands

Query Only. `PDElay?` returns the propagation delay between `MESial` crossings of the selected trace and the trace specified with the `MPAram REFTRace` command, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **PDE?**
PDELAY 6.9E-11,EQ

PERiod?

Group: Measurement Commands

Query Only. `PERiod?` returns the time taken for one complete signal cycle of the selected trace, defined by the `MESial` crossing level, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.) `PERiod` is the reciprocal of the frequency (`FREq`).

Examples: **PER?**
PERIOD 9.766E-7,EQ

PHAse?

Group: Measurement Commands

Query Only. `PHAse?` returns the phase relationship (from 0 to 360 degrees) of the selected trace to the reference trace, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: `PHA?`
`PHASE 1.064E-2,EQ`

PIN8 *<link>:<arg>*

Group: External I/O Commands

`PIN8` specifies parameters for printers that support standard Epson 8-pin Bit Image Graphics commands, such as the Tektronix 4644 and Epson EX−800.

Link: `FORMat: { DRAft | HIRes | REDuced }`

`FORMat` selects the output format. `HIRes` shows front panel intensified regions by dithering icon and text backgrounds and increasing foreground saturation. `DRAft` prints black-on-white background except for selected icons or text which are printed white-on-black background. `REDuced` is a quarter-size version of `DRAft` and prints black-on-white background only.

NOTE

Use `FORMat:HIRes` *for IBM Proprinter and Epson RX80 printers.*

Examples: `PIN8 FORM:DRA`

Link: `PORt: { CENTRonics | GPIb | RS232 }`

`PORt` specifies the output port for the printer.

Examples: `PIN8 POR:CENTR`

PIN24 *<link>:<arg>*

Group: External I/O Commands

PIN24 specifies parameters for printers that support extended Epson 24-pin Dot Graphics commands, such as the Epson LQ−1500.

Link: FORMat: { DRAft | HIRes | REDuced }

FORMat selects the output format. HIRes shows front panel intensified regions by dithering icon and text backgrounds and increasing foreground saturation. DRAft prints black-on-white background except for selected icons or text which are printed white-on-black background. REDuced is a quarter-size version of DRAft and prints black-on-white background only.

Examples: PIN24 FORM:DRA

Link: PORt: { CENTRonics | GPIb | RS232 }

PORt specifies the output port for the printer.

Examples: PIN24 POR:CENTR

POWeron?

Group: Miscellaneous/System Commands

Query Only. POWeron? returns the total number of times the instrument has been powered on.

Examples: **POW?**
POWERON 149

PP?

Group: Measurement Commands

Query Only. PP? returns the peak-to-peak voltage value (i.e., the difference between the MAX? and MIN? measurement values) for the selected trace, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **PP?**
PP 5.72E−1,EQ

RECall { **FPN**ext | **FPS**<*ui*> | <*qstring*> | **STD**<*sui*>: **TRA**<*dui*>}

Group:	Trace and Settings Commands
	Set Only. RECall recalls stored front panel settings from memory.
	Completion of RECall is signaled with event code 473, Recall complete.
Link:	FPNext
	Set Only. FPNext recalls from memory the next front panel setting in sequence. (The SETSeq command must be set to ON.)
Examples:	REC FPN
Link:	FPS<*ui*>
Arguments:	<*ui*> can range between 1 and 9999.
	Set Only. FPS<*ui*> recalls from memory the front panel settings specified by <*ui*>.
Examples:	REC FPS3
Link:	STD<*sui*>: TRAce<*dui*> (11801C only)
Arguments:	<*sui*> can range between 9 and MAXTranum. <*dui*> can range between 1 and 8.
	Set Only. STD<*sui*> creates a new displayed trace, TRA<*dui*>, using the saved trace description STD<*sui*>.
Examples:	REC STD9:TRA3 (11801C only)
Link:	<*qstring*>
	Set Only. <*qstring*> recalls from memory the front panel settings labeled by <*qstring*>.
Examples:	REC 'SETUP44'

REFSET*<ui>* *<link>***:***<arg>*

Group:	Measurement Commands
	REFSET sets reference value(s) on the specified trace for comparison measurements returned when COMpare is set to ON. (Refer to the COMpare command.)
Link:	CURRent: *<meas>*
Arguments:	*<meas>* can be any valid *<meas>* command.
	CURRent executes the specified measurement (*<meas>*), and stores the resultant value as the measurement reference.
Examples:	REFSE1 CURR:PP
Link:	*<meas>: <NRx>*
Arguments:	*<NRx>* can range between −5.0E + 20 and 5.0E + 20.
	<meas> sets the reference value for the specified measurement.
Examples:	REFSE1 PP:2.0
	Query Note: REFSET*<ui>*? returns the reference values for the specified trace. A *<meas>* without a reference value returns 0.0E + 0. REFSEt? returns the reference values for all defined traces in low-to-high numeric order.

REMove { ALLTrace | TRAce*<ui>*| *<qstring>*}

Group:	Trace and Settings Commands
	Set Only. REMove discards existing data and the trace definitions to remove traces from the display. If a trace is also stored in memory, the stored trace is not removed. (Use the DELete command to remove stored traces.)
Link:	ALLTrace
	ALLTrace removes all displayed traces. It is not an error to specify ALL-Trace when no traces are defined.
Examples:	REM ALL
Link:	TRAce*<ui>*

Arguments: `<ui>` can range between 1 and 8.

TRAce`<ui>` removes the specified trace from the display only, not from memory.

Examples: REM TRA7

Link: `<qstring>`

The `<qstring>` argument removes the trace labeled `<qstring>` from the display only, not from memory. Wildcard characters are interpreted. (Refer to page 2-149 for wildcard definitions.)

Examples: REM ′SAMPLE16′

RHOPos *<ui> <NRx>*

Group: Channel/Vertical Commands

Sets the rho calconstant on the specified trace. RHOPos is used only if you need to precisely adjust the YUNit conversion from volts to rho.

Examples: RHOP2 1.08

Query Note: RHOPos returns the rho calconstant for the specified trace in `<NR3>` form. RHOPos? returns the rho calconstant for all defined traces in numeric order.

RHOZero

Group: Cursor Commands

Set Only. Calibrates the rho scale of the selected trace. The average value of the points between the two cursors is used.

Examples: RHOZ

RISetime?

Group: Measurement Commands

Query Only. RISetime? returns the transition time on the selected trace of a rising-pulse edge, from the PROXimal to DISTal level, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **RIS?**
RISETIME 7.922E-9,EQ

RMS?

Group: Measurement Commands

Query Only. RMS? returns the true root mean square voltage of the selected trace, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **RMS?**
RMS 3.516E-1,EQ

RQS { ON | OFF }

Group: Status and Event Commands

RQS determines the instrument response to events detected during instrument operation. With RQS ON, the instrument asserts SRQ after an event; with RQS set to OFF, it does not. The power-on default for GPIB is RQS set to ON.

RQS is meaningless for the RS-232-C port; the RQS command is always set to OFF for RS-232-C.

Examples: RQS ON

RS232 *<link>***:***<arg>*

Group: External I/O Commands

RS232 sets parameters for the RS-232-C interface.

Link: BAUd: *<NRx>*

Arguments: *<NRx>* can be 110, 150, 300, 600, 1200, 2400, 4800, 9600, or 19200.

BAUd sets both the transmit and receive baud rates.

NOTE

Set the baud rate on the instrument before setting the baud rate on the controller.

Examples: RS232 BAU:9600

Link: DELAy: *<NRx>*

Arguments: *<NRx>* can range between 0 and 60 seconds. The minimum granularity is 1 ms (1.0E−3).

DELAy sets the minimum delay from receipt of a query to its response.

Examples: RS232 DELA:5.0E−3

Link: ECHo: { ON | OFF }

ECHo determines whether characters are echoed on the controller screen.

NOTE

You cannot send binary data to the instrument when ECHo *is set to* ON*.*

Examples: RS232 ECH:ON

Link: `EOL: { CR | CRLf | LF | LRCr }`

EOL selects the end of line output message terminator:

  `CR` Carriage return

  `LF` Line feed

  `CRLf` Carriage return followed by line feed

  `LFCr` Line feed followed by carriage return

All of the above are accepted as an input message terminator.

Examples: `RS232 EOL:CRL`

Link: `FLAgging: { SOFt | HARd | OFF }`

`FLAgging` controls I/O flagging. `SOFt` uses XON (DC1) and XOFF (DC3) handshaking. `HARd` uses the DTR and CTS control lines. Both `SOFt` and `HARd` flagging halt input when the buffer is three-quarters full, and restart input when the buffer is one-quarter full. `OFF` means there is no transmission control.

NOTE

  `SOFt` *flagging is usually not used with binary transfers because the binary data may contain unintended XON or XOFF controls.*

Examples: `RS232 FLA:SOF`

Link: `PARity: { ODD | EVEN | NONe }`

`PARity` sets the parity used for all RS-232-C data transfers. The instrument generates parity on output data and checks the parity on input data. An input parity error produces event code 653, `RS-232-C input parity error`.

Examples: `RS232 PAR:EVEN`

Link: `STOPBits: <NRx>`

Arguments: `<NRx>` can be either 1, 1.5, or 2.

`STOPBits` selects the number of transmission stop bits sent with each character to identify the end of data.

Examples: `RS232 STOPB:1.5`

Link: VERBose: { ON | OFF }

When VERBose is set to ON, the instrument returns error and warning messages to the controller at the time they occur. When VERBose is set to OFF, the controller must query the instrument for event messages.

Examples: RS232 VERB:ON

SAMid? [*<alpha>* [*<ui>*]]

Group: Status and Event Commands

NOTE

Possible Alpha values for the 11801C are A through D, for SM-11 Multi-channel units, and M for the mainframe. The Alpha value for the CSA 803C is always M, for mainframe.

Query Only. Returns the model number of the sampling head for all installed channels or for the channels of the specified mainframe. SAMid? returns the odd-numbered channel of each pair.

Examples: **SAM? M**
SAM M1:"SD–24",M3:"SD–24"

SELect { **TRA**ce*<ui>* | *<qstring>* }

Group:	Trace and Settings Commands
	SELect specifies the trace used by AUTOSet, HISTogram, MASK, measurement and cursor commands. By default, the most recently created trace is the selected trace until changed with SELect.
Link:	TRAce*<ui>*
Arguments:	*<ui>* can range from 0 to 8.
	The valid SELect TRAce*<ui>* *setting* range is 1 to 8. However, SELect? returns TRACE0 when no traces are defined. You can send SELECT TRACE0 to the instrument without an error; it is ignored.
Examples:	SEL TRA8
Link:	*<qstring>*
	<qstring> designates the trace labeled with *<qstring>* as the selected trace.
Examples:	SEL 'SAMPLE1'

SET?

Group: Data Transfer Commands

Query Only. SET? returns front panel settings to the controller in ASCII or binary format, depending on the state of the ENCdg SET command.

NOTE

SET? *is not strictly a query-only command. You can send settings back to the instrument (with some restrictions) to restore a previously-defined instrument state. However, the header* SET *is used only when sending binary data.*

ASCII SET? Response — SET? returns strings of instrument commands separated by semicolons. The following is an excerpt of a SET? response:

Examples: **SET?**
REMOVE ALLTRACE; GRATICULE XUNIT:SECONDS, YUNIT:VOLTS;
CHM1 OFFSET:0.0E+0, SENSITIVITY:2.0E-1,
SMOOTHING:ON, TDRDELAY:-3.91552E-1, ...

Binary SET? Response — SET? returns binary data in the following format:

<bblock>::= %*<byte cnt><settings><checksum>*

where *<byte cnt>* is a two-byte integer (MSB first) giving the length in bytes of the remainder of the binary block, including checksum; *<settings>* are binary-encoded data; and *<checksum>* is an 8-bit, twos complement of the modulo 256 sum of *<byte cnt>* and *<settings>* data.

NOTE

A binary SET? *command is not usable if the instrument's firmware has been upgraded since the binary* SET? *query was performed.*

Sending Settings Back to the Oscilloscope — Send settings as a complete set; do not edit or modify the data. For ASCII settings, simply send the entire set of strings. The binary SET? response returns the SET header at the beginning of the response; you must include the SET header when sending binary settings to the instrument. Refer to the command description for ENCdg SET:BINary.

SETSeq { ON | OFF }

Group: Trace and Settings Commands

SETSeq controls the sequencing of front panel settings. When SETSeq is set to ON, the settings are sequenced and the RECall FPNext command recalls the next set of stored front panel settings from memory.

NOTE

If SETSeq *is set to* ON *and all stored settings are deleted,* SETSeq *is set to* OFF.

Examples: SETS ON

SFRequency?

Group: Measurement Commands

Query Only. SFRequency? returns the spectral frequency (harmonic or peak), followed by an accuracy qualifier. The measurement is taken on the selected trace.

Examples: **SFR?**
SFRequency 2.33E+2, EQ

SMAgnitude?

Group: Measurement Commands

Query Only. SMAgnitude? returns the spectral magnitude (harmonic or peak), followed by an accuracy qualifier. The measurement is taken on the selected trace.

Examples: **SMA?**
SMAgnitude 2.33E+2, EQ

SNRatio**?**

Group: Measurement Commands

Query Only. `SNRatio?` returns `AMPlitude/NOIse`. (`NOIse` is always RMS.) The measurement is taken on the selected trace.

Examples: `SNR?`
`SNRatio 2.4316E+2, EQ`

SPEaker **{ ON | OFF }**

Group: Miscellaneous/System Commands

`SPEaker` controls the instrument audio feedback (that is, whether you hear a click when you touch the front panel).

Examples: `SPE ON`

SRQMask *<link>*: { ON | OFF }

Group: Status and Event Commands

SRQMask controls the reporting of selected classes of events, regardless of the state of the RQS command. If an SRQMask link is set OFF, that class of events is not reported. At power-on, all SRQMask links are set to ON except ABSTouch, IDPROBE, and USEr. Table 2-68 lists all SRQMask links, their meanings, and associated event code(s).

Table 2-68: SRQMASK Lists

Link	Meaning	Event Code(s)
CMDerr:	Controls reporting of command errors	100−199
EXErr:	Controls reporting of execution errors	200−299, 2000−2999
EXWarn:	Controls reporting of execution warnings	500−599
INErr:	Controls reporting of internal errors	300−399
INWarn:	Controls reporting of internal warnings	600−699
OPCmpl:	Controls reporting of operation-complete events	450, 460−464, 473−475
USEr:	Controls whether the RQS icon is displayed and whether RQS icon touches are reported	403

Examples: SRQM ABS:ON

STAT? [**MEAN** | **STDD**ev | **NCU**rrent]

Group: Measurement Commands

Query Only. STAT returns for the *<meas>* specified by STATIstics, the mean, standard deviation and number of *<meas>* samples used in the computation or optionally just one of these. A qualifier is returned at the end of the response. The qualifiers are:

EQ   All *<meas>* values used had an EQ qualifier

UN   Some *<meas>* values did not have an EQ qualifier

ER   Most or all *<meas>* values had an ER qualifier.

NOTE

<meas> values with an ER qualifier are not used to compute statistics.

Examples: **STAT?**
STAT MEAN:2.252E-10,EQ,STDDEV:
4.2E-13,EQ,NCURRENT:128,EQ

STATHist *<link>*

Group: Measurement Commands

STATHist provides a number of query links to access the statistical information created by the Histogram function. Refer also to the HISTogram command.

Link: ? HIST.pt

? HIST.pt returns the number of sample points processed into the histogram data.

Examples: **STATH? HIST**
STATHIST HIST.PT:79195239

Link: ? NWFm

? NWFm returns the number of traces processed into the histogram data.

Examples: **STATH? NWF**
STATHIST NWFM:197610

Link: `? MEAN`

`? MEAN` returns the statistical mean value for the histogram data.

Examples: **STATH? MEAN**
STATHIST MEAN:2.970E–3

Link: `? RMSDev`

`? RMSDev` returns the RMS (standard deviation) value for the histogram data.

Examples: **STATH? RMSD**
STATHIST RMSDEV:3.310822E–1

Link: `? PP`

`? PP` returns the peak-peak measurement for the histogram data.

Examples: **STATH? PP**
STATHIST PP:7.999844E–1

Link: `? SIGMA1`

`? SIGMA1` returns the percentage of points in the histogram that are within the area that is one STD of the MEAN.

Examples: **STATH? SIGMA1**
STATHIST SIGMA1:2.931126E+1

Link: `? SIGMA2`

`? SIGMA2` returns the percentage of points in the histogram that are within the area that is two STDs of the MEAN.

Examples: **STATH? SIGMA2**
STATHIST SIGMA2:1.0E+2

Link: `? SIGMA3`

`? SIGMA3` returns the percentage of points in the histogram that are within the area that is three STDs of the MEAN.

Examples: **STATH? SIGMA3**
STATHIST SIGMA3:1.0E+2

STATIstics *<link>***:***<arg>*

Group:	Measurement Commands

On the selected trace, specifies the `<meas>` and number of samples used to compute the mean and standard deviation. The `<meas>` must be on the `MSLIst` for the specified trace.

Link:	`MEAS: { <meas>	NONe }`

`MEAS` selects the `<meas>` on which to compute statistics. `NONe` clears the selected measurement.

Examples:	`STATI MEAS:PP`

Link:	`N: <NRx>`

Arguments:	`<NRx>` can range between 2 and 128.

`N` sets the number of samples on which to compute the mean and standard deviation.

Examples:	`STATI N:25`

Link:	`RESET`

`RESET` resets the statistics counter and begins collecting data again.

Examples:	`STATI RESET`

STByte**?**

Group: Status and Event Commands

Query Only, RS-232-C Only. STByte? enables an RS-232-C controller to read the status byte of the current RS-232-C event by mimicking a GPIB serial poll at the RS-232-C port. STByte? is not valid at the GPIB port.

NOTE

The status byte is defined in the section on Event Reporting later in this manual.

Examples: **STB?**
STBYTE 2

NOTE

In the above example, the 2 indicates an operation-complete event with RQS *set to* OFF.

STD [*<sui>*] **?** (11801C *only)*

Group: Trace and Settings Commands

Query Only. `STD<sui>?` returns the links and arguments of the four component commands required to define the saved trace description *<sui>*: `TRAce`, `WIN`, `MSLISt`, and `MPAram`. If *<sui>* is not specified, the response is given in order of the saved trace description number. The range of *<sui>* is 9 to `MAXTranum`.

NOTE

Main record trace descriptions do not include the `WIN` *response.*

`STD<sui>?` responds in the following order:

`TRACE<ui> DESCRIPTION:<qstring>, WFMCALC:<arg>;`

`WIN<ui> MODE:<arg>, TRACK:<arg>, POS:<NR3>, LMODE:<arg>,`
`    LEVEL:<NRx>, TRANS:<NR1>, SLOPE:<arg>,`
`    FILTERING:<NRx>, TOPLINE:<NRx>, BASELINE:<NR3>,`
`    ABSLEVEL:<NR3>;`

`MSLIST<ui> { EMPTY | <meas> [ ,<meas>...];`

`MPARAM<ui> MMODE:<arg>, MLEVELMODE:<arg>, MTRACK:<arg>,`
`    BASELINE:<NR3>, TOPLINE:<NR3>,`
`    MESIAL:<NR3>, PROXIMAL:<NR3>, DISTAL:<NR3>,`
`    DAINT:<arg>, SNRATIO:<NR3>, LMZONE:<NR1>,`
`    RMZONE:<NR1>, MSLOPE:<arg>,`
`    MFILTERING:<NR1>, MTRANS:<NR1>,JITTLOCATION:<arg>,`
`    NOISLOCATION:<arg>, DISPERSION:<arg>,`
`    HNUMBER: <NR1>, PINDEX:<NR1>, SMODE:<arg>,`
`    ERMETHOD:<arg>, JITTLEVEL:<NR3>, JITT.HISTPT:<NR1>,`
`    NOIS.HISTPT:<NR1>, REFTRACE:<arg>,`
`    REFBASELINE:<NR3>, REFTOPLINE:<NR3>,`
`    REFMESIAL:<NR3>, REFSNRATIO:<NR3>, REFLMZONE:<NR3>,`
`    REFRMZONE:<NR3>, REFSLOPE:<arg>,`
`    REFFILTERING:<NR3>, REFXSITION:<NR1>,`
`    REFLEVEL:<NR3>`

STOList**?**

Group: Trace and Settings Commands

Query Only. `STOList?` returns a list of all stored traces, or `EMPTy` if there are no stored traces.

Examples: **`STOL?`**
`STOLIST STO2,STO9,STO56,STO200`

STONum?

Group: Trace and Settings Commands

                Query Only. `STONum?` returns the number of traces stored in memory.

Examples: `STON?`
                     `STONUM 4`

STORe [*<link>***:**]*<arg>*

Group: Trace and Settings Commands

                Set Only. `STORe` saves front panel settings (FPS) in nonvolatile RAM. `STORe` also copies a displayed trace to nonvolatile RAM; the trace is not removed from the display.

                STORE Constraints — You cannot store an XY trace. An existing `STO<ui>` location can be overwritten only if the record lengths of the new and stored traces are the same; the previous trace data is destroyed. If the previously stored trace was a component of a displayed trace, the displayed trace changes to include the newly stored trace.

Link: `{ FPS<ui> | <qstring> }`

Arguments: `<ui>` can range between 1 and 9999.

                Set Only. `FPS<ui>` stores the current front panel settings tagged with the specified number or `<qstring>` label. If `<ui>` is an existing FPS number, or if the label identifies an existing FPS number, the new data overwrites the previous data. If the label does not identify an existing FPS number, the data is stored in the next available FPS number with that label assigned to it. Wildcard characters are not interpreted. (Refer to page 2-149 for the definition of wildcards.)

Examples: `STOR FPS5`

Link: `TRAce<ui>: { STO<ui> | <qstring> | STD<ui> }`

Arguments: `<ui>` can range between 1 and 8 for `TRAce<ui>`, between 1 and 9999 for `STO<ui>`, and between 9 and `MAXTranum` for `STD<ui>`.

For the 11801C *only*, `TRACE<ui>:STD<ui>` creates a saved trace description with the characteristics of the specified `TRAce<ui>`. The range for `STD<ui>` is 9 to `MAXTranum`.

Set Only. `TRAce<ui>` stores a copy of the `TRAce<ui>` trace in nonvolatile RAM at the location specified either by `STO<ui>` or by the `<qstring>` label. Wildcard characters are not interpreted. If the label identifies an existing `STO` location, the new data overwrites the previous location. If the label does not identify an existing `STO` location, the data is stored in the next available `STO` location with that label assigned to it.

Examples: `STOR TRA1:STO10`

Link: `<qstring>: STO<ui>`

Arguments: `<ui>` can range between 1 and 9999.

Set Only. `<qstring>` stores a copy of the labeled trace identified by `<qstring>` in memory at the location specified by `STO<ui>`. (Wildcard characters are not interpreted.)

Examples: `STOR 'SAMPLE12': STO100`

TBCalmode { FASt | HIPrec | OFF | ONCE }

Group: Calibration Commands

Selects either a faster, lower precision on-line time base calibration procedure (`FASt`), or a slower, higher precision calibration (`HIPrec`). `ONCE` performs a single calibration in the last mode selected (either `FASt` or `HIPrec`) then sets the `TBCalmode` to `OFF`. The initialized default is `FASt` mode.

Examples: `TBC HIP`

TBMain *<link>***:***<arg>*;
TBWin *<link>***:***<arg>*

Group: Time Base/Horizontal Commands

TBMain sets the Main time base parameters and TBWin sets the Window time base parameters. Both commands use the same links and arguments.

Link: LENgth: *<NRx>*

Arguments: *<NRx>* can be 512, 1024, 2048, 4096, or 5120.

LENgth sets the selected time base to the specified record length, scaled in points per trace.

Examples: TBM LEN:1024; TBW LEN:512

Link: TIMe: *<NRx>*

Arguments: *<NRx>* can range between 1E−12 and 5E−3 in 1 ps (1E−12) intervals.

TIMe sets the horizontal scale in 1 ps (1E−12) intervals. However, 1 ps/div is only possible when the record LENgth is 1024 or less. The range must also meet this formula:

$$min\ interval \le \left(\frac{TIMe \times 10}{round\ record\ length} \right) \le max\ interval$$

where *min interval* is 10 fs (10E−15), *max interval* is 100 μs (100E−6), and *round record length* is the nearest multiple of 500 to the current value of LENgth (except 4096 is rounded up to 5000.)

The minimum value for TIMe is:

$$1ps \le min\ interval \times \frac{round\ record\ length}{10}$$

The maximum value for TIMe is:

$$5ms \ge max\ interval \times \frac{round\ record\ length}{10}$$

NOTE

TBWin TIMe ≤ TBMain TIMe.

Examples: TBW TIM:5.0E−9

Link: ? XINcr <*NR3*>

Query Only. ? XINcr returns the sample interval of the selected time base. Units are seconds per point, feet per point, or meters per point.

Examples: **TBW? XIN**
TBWIN XINCR:1.0E-10

Calculating *Duration* — *Duration* is used when calculating the range of other commands, such as MAINPos.

Use the following formula for *main duration*:

(TBMain XINcr) * (TBMain LENgth − 1)

Use the following formula for *window duration*:

(TBWin XINcr) * (TBWin LENgth − 1)

TESt [XTNd]

Group: Diagnostics Commands

Set Only. TESt initiates the Self-tests diagnostics or, with the XTNd argument, the Extended Diagnostics.

Completion of diagnostics is signaled with either event code 460 or 474, successful completion of tests, or event code 394, completion with failed tests.

NOTE

TESt *deletes all user-defined expansion strings created with the* DEFine *command, resets the* TEXT X:, Y: *coordinates to 0,0, and removes user-entered text from the display.*

Examples: TES XTN

TEXt [*<link>***:**]*<arg>*

Group:	Display and Color Commands

Set Only. `TEXt` writes user defined character(s) to a selected area of the screen. The large selection of characters are shown in Appendix C.

Colored Text — Text can be written to the screen in any of the seven colors normally used for trace and information display. The eighth color is the background color and is not useful for text. The Escape Character set provides this "color font" capability. Refer to the Escape Character set in Appendix C.

The default colors listed in Table 2-69 can be used when creating text. The escaped character is listed along with its associated color. The colors can be changed with the `COLor` command. All colored text will change accordingly.

Table 2-69: Escape Character Default Colors

Escape Character	Color
^ [(space)	White
^ [!	Rose
^ ["	Green
^ [#	Purple
^ [$	Lt Blue
^ [%	Gray
^ [&	Red

The following example would create a series of words across the screen each having its named color. Single quotes were used to delimit the string.

Examples:	`TEX X:1,Y:1,STRING:'[ WHITE ^[ !ROSE ^[ "GREEN ^[ #PURPLE` `^[ $LTBLUE'`

Link:	`CLEar`

Set Only. `CLEar` removes all user-defined text from the display.

Examples:	`TEX CLE`

Link: STRing: *<qstring>*

Set Only. STRing specifies the text that is to be displayed at the X: and Y: coordinates.

Examples: TEX STR:'Select a trace'

Link: X: *<NRx>*

Arguments: *<NRx>* can range between 0 and 49.

Set Only. X specifies the horizontal position (X coordinate) of a character in discrete character cells. The range is 0 (left edge of the graticule) to 49 (right edge of the graticule).

Examples: TEX X:10

Link: Y: *<NRx>*

Arguments: *<NRx>* can range between 0 and 31.

Set Only. Y specifies the vertical position (Y coordinate) of a character in discrete character cells. The range is 0 (top edge of the graticule) to 31 (bottom edge of the graticule).

Examples: TEX Y:20

Figure 2-11 shows some TEX X:,Y: cell coordinates.

```
0,0                              49,0




                    25,15




0,31                            49,31
```
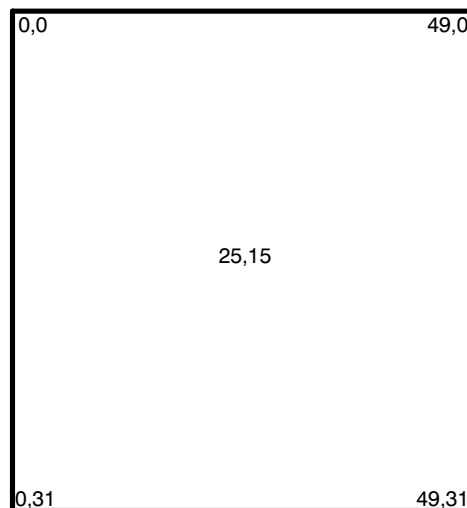
Figure 2-11: TEXT X:,Y: Display Coordinates

THD?

Group: Measurement Commands

Query Only. `THD?` returns the total harmonic distortion of the selected trace, that is, the magnitude of the fundamental divided by the sum of the magnitudes of all other harmonics, followed by an accuracy qualifier. The measurement is taken on the selected trace.

Examples: `THD?`
`THD 6.084EdD+0, EQ`

TIMe *<qstring>*

Group: Miscellaneous/System Commands

`TIMe` sets the time of day on the internal clock.

Link: *<qstring>*

Arguments: *<qstring>* specifies the time in the format *<hh>:<mm>:<ss>.*

where *<hh>* is the hour, *<mm>* is minutes, and *<ss>* is seconds in 24-hour format.

Examples: `TIM '17:25:30'`

TOPBase *<arg>*

Group: Measurement Commands

Selects the method of determining topline and baseline.

Link: `{ IEEE | ALTErnate }`

`IEEE` uses the standard IEEE histogram method. `ALTErnate` uses a method that smooths and differentiates trace data before calculations. `STAT` measurement mode always uses `IEEE`, regardless of the `TOPBase` value.

TRAce*<ui> <link>:<arg>*

Group:	Trace and Settings Commands
	TRAce*<ui>* defines a trace and its characteristics. The range of *<ui>* is 1 to MAXTranum.
Link:	DEScription: *<qstring>*
Arguments:	*<qstring>* must be less than or equal to 55 characters in length.
	DEScription defines the source expression(s) of the selected trace.

    <y exp> [VS *<x exp>*] [ON *<time base>*]

where:

<y exp>, *<x exp>* ::=	Expressions
[VS *<x exp>*] ::=	Indicates an XY trace; if omitted, the trace is YT
[ON*<time base>*]::=	Indicates time base — { Main | Win }; if omitted, defaults to Main

Terms Available to Form Expressions:

<alpha><ui>	Channel designator, e.g. M1
STO*<ui>*	Stored trace, the range is 1 to 9999
<NRx>	Scalar number

<function>	Any of the following functions:
	    ABS | AVG | DIFF | ENV | EXP |
	    INTG | FILTER | LN | LOG |
	    SIGNUM | SMOOTH | SQRT |
	    FFTMAG | FFTPHASE

Operators Available to Form Expressions:

+ (addition)	* (multiplication)
– (subtraction, negation)	/ (division)

NOTE

You cannot use a trace description that consists of only stored or scalar elements as the argument of an AVG or ENV function. You also cannot create a trace with only stored or scalar elements on the WIN time base. For a <ui> value of 9 or above (11801C only), you can only use the AVG and ON WIN operators and functions (that is, you cannot use any other operators or functions, such as STO or XY).

Examples: TRA2 DES:′ENV(M2)′;
              TRA3 DES:′STO9+M1′

XY Trace Considerations — The instrument permits only one acquired XY trace or two unacquired XY traces to be displayed via TRACE<*ui*> DESCRIPTION. (An acquired XY trace description has at least one acquired signal component; an unacquired XY trace description has only stored or scalar components.)

Components of XY Descriptions:

Acquired XY Description	Unacquired XY Description
"M1 VS M2"	"STO50 VS STO12"
"M1 VS STO3"	"STO90 VS 200"

In addition, the horizontal and vertical components ($<x\ exp>$ and $<y\ exp>$) must have the same scaling mode; both must be integer mode or both floating-point mode traces.

Link: ? WFMCalc { FASt | HIPrec }

Query Only. ? WFMCalc returns whether a trace was created in integer mode (FASt) or floating-point mode (HIPrec). Once a trace is created in one mode, you cannot change the trace to the other mode. (Refer to WFMScaling command.)

Examples: **TRA2? WFMC**
              TRACE2 WFMCALC:HIPREC

Query Note: TRAce<*ui*>? returns the links and arguments of the specified trace in the following order:

    TRACE<*ui*> DESCRIPTION:<*qstring*>,

TRAce? returns the same information as TRAce<*ui*>? for all defined traces in low-to-high order.

TRAList?

Group: Trace and Settings Commands

Query Only. Returns a list of all defined traces, displayed traces or EMPTy if no traces are defined.

Examples: **TRAL?**
              TRALIST TRA2,TRA3,

TRANUm?

Group:	Trace and Settings Commands

Query Only. TRANUm? returns the number of traces displayed on the front panel. Range is 0 to 8 in *<NR1>* form.

Examples:　**TRANU?**
TRANUM 4

TRIgger *<link>:<arg>*

Group:	Trigger Commands

Sets trigger parameters.

Link:　ATTenuation { X1 | X10 }

ATTenuation selects X1 or X10 attenuation only when the trigger SOUrce is EXTernal. When ATTenuation is set to X10, the trigger level readout values are multiplied by 10. (11801C only)

Examples:　TRI ATT:X1

Link:　ENHANced: { ON | OFF }

ENHANced controls the state of the metastable trigger reject feature. When ENHANced is set ON, the timebase detects trigger events that are metastable and rejects the acquired data. When set OFF, the timebase allows metastable trigger events to be displayed.

Examples:　TRI ENHAN:OFF

Link:　EXTCoupling { AC | DC }

EXTCoupling selects trigger coupling only when the trigger SOUrce is EXTernal. (11801C only)

Examples:　TRI EXTC:DC

Link: `HIFreq: { ON | OFF }`

`HIFreq` controls the state of the trigger hysteresis. When `HIFreq` is set `ON`, trigger hysteresis is removed, improving sensitivity. This setting should only be used when the trigger signal edge has a slew rate of 1 V/ns or greater. When `HIFreq` is set `OFF`, trigger hysteresis is retained, improving noise rejection at low frequency.

Examples: `TRI HIF:OFF`

Link: `HOLdoff: <NRx>`

`HOLdoff` sets the requested trigger holdoff period, but only takes effect if `HMOde` is set to `MANual`. The range is from 5 µs to 2.5 s.

Examples: `TRI HOL:10.0E-6`

Link: `HMOde: { AUTO | MANual }`

`HMOde` controls the trigger holdoff period. When `AUTO` is selected, trigger holdoff is automatically set to the minimum possible holdoff value, which is slightly beyond the rightmost point acquired in the Main record. The minimum holdoff value is about 5 µs. When `MANual` is selected, the Actual Holdoff setting for the trigger accounts for the value of Requested Holdoff, as well as the timebase settings. Refer to the *User Manual* for more information.

Examples: `TRI HMO:AUTO`

Link: `LEVel: <NRx>`

Arguments: The range for `<NRx>` is described below.

Sets trigger level only for the `EXTernal` trigger source or a channel with trigger source capability. Range for this link is normally −1 V to +1 V. When `ATTenuation` is X10 and `SOUrce` is `EXTernal`, the range is −10 V to +10 V.

NOTE

When using EXTCoupling:DC, set `LEVel` after the coupling. If you set the level first, the signal may become too far out of range to trigger. If this happens (i.e., no trigger), you can restore function by sending a DCL (Device Clear) signal.

Examples: `TRI LEV:0`

Link: `MODe: { AUTO | NORmal }`

Selects triggering mode. In `NORmal` mode, the instrument acquires data only if a valid trigger is found. If not found, trace acquisition halts, freezing the display and halting trace calculation and measurement. In `AUTO` mode, the instrument is set to normal triggering at the beginning of each sweep. If a valid trigger is not found within 20 ms, the instrument switches to the internal clock. At the end of that sweep, it again waits 20 ms for a valid trigger and the process is repeated. Therefore, `NORmal` should be selected if your trigger rate is very slow.

Examples: `TRI MOD:AUTO`

Link: `SLOpe: { PLUs | MINUs }`

Selects the trigger slope only for an external or channel source.

Examples: `TRI SLO:MINU`

Link: `SOUrce: { EXTernal | INTernal | PREscaler | <alpha><ui> }`

`SOUrce` selects the trigger source. `EXTernal` argument selects the external trigger input. `INTernal` selects the internal 200 kHz clock. `PREscaler` selects the prescaler trigger input. If your instrument is configured without the prescaler trigger input an event error will be reported.

`<alpha><ui>` selects a sampling head channel for sampling heads that support trigger source. Currently, no sampling heads support this feature.

Examples: `TRI SOU:EXT`

Link: ? STATUs { TRG | NOTrg }

Query Only. ? STATUs returns the trigger status. TRG means the time base is triggered. NOTrg means the time base is not triggered.

Examples: **TRI? STATU**
TRIGGER STATUS:TRG

Query Note: The TRIgger? query returns all links and their arguments, in the following order:

For CSA 803C Only: TRIGGER MODE:*<arg>*, STATUS:*<arg>*,
    SOURCE:*<arg>*, LEVEL:*<NRx>*, HIFREQ:*<arg>*,
    ENHANCED:*<arg>*, HOLDOFF:*<NRx>*, HMODE:*<arg>*,
    SLOPE:*<arg>*

For 11801C Only: TRIGGER MODE:*<arg>*, STATUS:*<arg>*,
    ATTENUATION:*<arg>*, SOURCE:*<arg>*, LEVEL:*<NRx>*,
    EXTCOUPLING:*<arg>*, HIFREQ:*<arg>*, ENHANCED:*<arg>*,
    HOLDOFF:*<NRx>*, HMODE:*<arg>*, SLOPE:*<arg>*

UID {*<link>:<arg>*}

Group: Status and Event Commands

UID queries or sets the serial numbers of the instrument and its sampling heads.

Link: `? <alpha><ui> <qstring>`

Query Only. `? <alpha> <ui>` returns the serial number of the specified channel.

Examples: **UID? M3**
UID M3:"B010100"

Link: `? MAIn: <qstring>`

Arguments: `<qstring>` must be less than or equal to 10 characters in length.

MAIn queries the serial number of the instrument.

Examples: **UID? MAI**
UID MAIN:"B010400"

Query Note: The UID? query returns its links in the following order:

UID MAIN:*<qstring>*,M*<ui>*:*<qstring>*

UNDEF {*<qstring>*| **ALL** }

Group: Miscellaneous/System Commands

Set Only. UNDEF removes from the list of logical names defined by DEF either the specified logical name or ALL defined logical names.

Examples: UNDEF 'TB?'

UNDershoot**?**

Group: Measurement Commands

Query Only. `UNDershoot?` returns the difference between the `BASeline` value and the minimum signal amplitude of the selected trace, given as a percentage of the difference between the `TOPLine` and `BASeline` values, and followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: `UND?`
`UNDERSHOOT 2.334E-9,EQ`

UPTime**?**

Group: Miscellaneous/ Commands

Query Only. `UPTime?` returns the total number of hours the instrument has been powered on, in `<NR3>` form.

Examples: `UPT?`
`UPTIME 1.243E+3`

USEREye *<link>:<arg>*

Group: Display and Color Commands

Link: `CREate`

Set Only. Creates the standard mask User Eye. Refer to the *User Manual*.

Link: `{?} RATe:<NRx>`

`RATe` sets the rate that will be used when the next `USEREye CREate` command is executed.

Examples: `RAT:1E+9`

V1Bar *<link>***:***<arg>*;
V2Bar *<link>***:***<arg>*

Group:	Cursor Commands

V1Bar and V2Bar set the absolute position of the vertical bar cursors.

Link: XCOord: *<NRx>*

Arguments: The range for *<NRx>* is described below.

XCOord positions the first or second vertical bar cursor using the units of the selected trace.

The XCOord range for a Main trace is from:

    MAINPos to (MAINPos + 10.22 * TBMain TIMe)

The XCOord range for a Window trace is from:

    WINPos to (WINPos + 10.22 * TBWin TIMe)

Examples: V1B XCO:3.8E−4

Link: XDIv: *<NRx>*

Arguments: *<NRx>* can range between −5.12 and +5.10.

XDIv positions the first or second vertical bar cursor in graticule divisions. (−5.12 is the left edge of the display.)

Examples: V2B XDI:−4.1

VPCurve? { NWFm }

Group: Data Transfer Commands

Query Only. VPCurve? transfers all trace records that make up a variable persistence trace. All trace records are transferred using the consecutive CURV? commands separated by semicolons. The EOI is sent after the last trace record. Refer to the CURVe command on page 2-98 for detailed data format information. Also refer to the ENCdg encoding command. VPCurve? returns:

> VPCURVE NWFM:*<NR1>*; CURVE...; CURVE...; CURVE...*<EOI>*

Link: ? NWFm *<NR1>*

? NWFm returns the current number of trace records that are used to create the variable persistence trace. This is also the number of trace records that would be returned by VPCurve though more traces may be added to the ? NWFm total after this query.

Examples: **VPC? NWF**
VPCURVE NWFM:25

WAVfrm?

Group: Data Transfer Commands

Query Only. WAVfrm? returns the trace preamble and data points for the trace specified by OUTput. WAVfrm? is equivalent to entering: WFMPRE?;CURVE?.

Refer to the WFMPre and CURVe commands for information on what is returned by WAVfrm?

WFMPre *<link>***:***<arg>*

Group: Data Transfer Commands

WFMPre transmits a Tek Codes and Formats preamble for each trace sent to or from the controller. The preamble is generated by the instrument and provides scaling and other information for the trace data transferred with the CURVe command. The trace sent to the instrument with CURVe is specified with the INPut command. The trace returned to the controller with CURVe? is specified with the OUTput command.

NOTE

Sending WFMPre *implicitly deletes any existing trace data at* INPut STO*<ui> and replaces it with null (unacquired) data points. If* STO*<ui> is the sole component of a displayed trace (e.g., TRA3 DES:* "STO22"*), that trace is removed from the display. If* STO*<ui> is one component of a complex trace (e.g., TRA4 DES:* "STO22+L1"*), you cannot send a trace preamble to that* INPut STO*<ui> location because you cannot delete a stored trace that is part of a complex trace.*

XY Note: The instrument does not support stored XY traces. Therefore, although XY traces can be transferred to the controller, they cannot be sent back to the instrument.

Link: ? BIT/nr 16

Query Only. ? BIT/nr returns the number of bits per binary trace point (always 16).

Examples: **WFMP? BIT**
WFMPRE BIT/NR:16

Link: ? BN.fmt RI

Query Only. ? BN.fmt returns the Tek Codes and Formats binary number format, which is always RI (right-justified, twos-complement integers).

Examples: **WFMP? BN.**
WFMPRE BN.FMT:RI

Link: `? BYT/nr 2`

Query Only. `BYT/nr` returns the binary data field width (which is always two bytes per binary trace point).

Examples: **`WFMP? BYT/`**
`WFMPRE BYT/NR:2`

Link: `? BYT.or { LSB | MSB }`

Query Only. `? BYT.or` returns the transmission order of binary trace data returned by `CURVe?`. The transmission order is set by the `BYT.or` command.

Examples: **`WFMP? BYT`**
`WFMPRE BYT.OR:LSB`

Link: `? CRVChk { CHKsm0 | NONe | NULl }`

Query Only. `? CRVChk` returns the type of checksum appended to the trace data after it is returned by a `CURVe?` query. The types are defined in Table 2-70.

Table 2-70: Checksum Types

Type	Meaning
CHKsm0	Standard Tek Codes and Formats checksum. Returned when ENCDG WAVFRM is set to BINary and OUTput is set to STO<*ui*>.
NONe	No checksum appended. Returned when ENCDG WAVFRM is set to ASCii.
NULl	Zero checksum value appended. Returned when ENCDG WAVFRM is set to BINary and OUTput is set to TRAce<*ui*>.

Examples: **`WFMP? CRVC`**
`WFMPRE CRVCHK:CHKSM0`

Link: DATE: *<qstring>*

Arguments: *<qstring>* is of the form *<'dd-mon-yy'>* where *dd* represents the day of the month, *mon* represents the first three letters of the month, and *yy* represents the last two digits of the year.

    DATE is the date stamp for the trace. The date stamp is recorded when a trace is stored, or you can set it with this link. If WFMPre? DATE is queried when OUTput is TRAce*<ui>* (i.e., a displayed trace), the current date is returned.

Examples: WFMP DATE: '05-NOV-93'

Link: ? ENCdg { ASCii | BINary }

    Query Only. ? ENCdg returns the state of the data encoding set with the ENCdg command. This link is equivalent to an ENCDG? WAVFRM query.

Examples: **WFMP? ENC**
    WFMPRE ENCDG:ASCII

Link: LABel: *<qstring>*

Arguments: *<qstring>* must be less than or equal to 10 characters in length.

    LABel is the optional label associated with the trace. If the trace has no label, querying WFMPre? LABel returns a null string (LABEL:"")

Examples: WFMP LAB:'SAMPLE3'

Link: `NR.pt: { 512 | 1024 | 2048 | 4096 | 5120 }`

`NR.pt` specifies the number of points in the transmitted trace record. It is normally the same as `{ TBMain | TBWin } LENgth`.

NOTE

If `OUTput` specifies a displayed trace when Pan/Zoom mode is set to `ON` and `HMAg` is greater than 1 for that trace, then the value returned by `WFMPRE? NR.PT` equals the number of points displayed on the front panel, rather than the value of `{ TBMain | TBWin } LENgth.`

For example, under the following conditions the `WFMPRE? NR.PT` query returns 512:

```
TRACE1 DESCRIPTION:"M1 ON MAIN"
    TBMAIN LENGTH:2048
    ADJTRACE1 PANZOOM:ON,HMAG:4
    OUTPUT TRACE1
```

Examples: `WFMP NR.:1024`

Link: `? PT.fmt: { ENV | Y | XY }`

Query Only. `? PT.fmt` indicates the point format of the trace data. `ENV` applies to YT traces transmitted as maximum-minimum point-pairs, with the maximum point transmitted first. `Y` indicates a YT trace, which returns one ASCII or binary data point for each point in the trace record. `XY` is an XY trace which returns an X, Y point-pair for each point in the trace record.

NOTE

You cannot send XY traces to the instrument.

Examples: **WFMP? PT.**
`WFMPRE PT.FMT:Y`

Link: `RHOFactor: <NRx>`

A generalization of the concept of TDR polarity. For trace expressions (i.e., M1−M2), `RHOFactor` produces rho values from −1 to +1.

Examples: `WFMP RHOF:0.23E-1`

Link: `RHOPos: <NRx>`

A calculated value used in the conversion from volts to rho. When the reference voltage changes, baseline correction will calculate a new `RHOPos`.

Examples: `WFMP RHOP:0.835E-1`

Link: `TIMe: <qstring>`

Arguments: `<qstring>` is of the form `<'hh:mm:ss'>` where `hh` is the hour in 24-hour format, `mm` is the minute, and `ss` is the second.

`TIMe` is the time stamp for the trace. The time stamp is recorded when a trace is stored, or you can set it with this link. If `WFMPre? TIMe` is queried when `OUTput` is `TRAce<ui>`, the system clock time is returned.

Examples: `WFMP TIM: '07:15:13'`

Link: `? WFId { STO<ui> | TRAce<ui> }`

Query Only. `? WFId` identifies the source trace for this preamble. (The information returned by this link is the same as that returned by an `OUTput?` query for a single trace.)

Examples: **`WFMP? WFI`**
`WFMPRE WFID:TRACE7`

Link: `XINcr: <NRx>`

Arguments: `<NRx>` must be greater than or equal to 1.0E −14 seconds/point.

`XINcr` specifies the horizontal sample interval of a YT trace.

Examples: `WFMP XIN:1.0E-9`

Link: `XMUlt: <NRx>`

`XMULt` specifies the vertical scale factor, in `XUNit` per unscaled data point value, of the horizontal component of an XY trace.

NOTE

For `XMULt` *usage, refer to the trace scaling formulas in the* `CURVe` *entry.*

Examples: `WFMP XMULT:1.0E-1`

Link: `? XUNit { DIVS | FEEt | INChes | METers | SEConds | VOLts }`

Query Only. `?` `XUNit` returns the horizontal units (X-axis) of the trace data at the time of trace creation. For XY traces, `XUNit` is the vertical units of the horizontal component. `XUNit` returns `DIVS` when the units of the trace are indeterminate or undefined.

Examples: **WFMP? XUN**
WFMPRE XUNIT:SECONDS

Link: `XZEro:` *<NRx>*

Arguments: *<NRx>* can range between 22E−9 and 1E + 15.

`XZEro` specifies the number of seconds of post-trigger of a YT trace; or specifies the vertical offset of the horizontal component of an XY trace.

Examples: `WFMP XZE:2.5E−2`

Link: `YMUlt:` *<NRx>*

Arguments: *<NRx>* can range between 1E−15 and 1E + 15.

`YMUlt` specifies the vertical scale factor, in `YUNit` per unscaled data point value, of a YT trace, or specifies the vertical scale factor, in `YUNit` per un-scaled data point value, of the vertical component of an XY trace. (`YMUlt` is equal to the vertical units-per-division, such as volts, divided by 6400.)

Examples: `WFMP YMU:1.5625E−4`

Link: `YUNit: { DIVS | RHO | VOLts }`

`YUNit` specifies the vertical units (Y-axis) of the trace data (YT or XY) to be transferred via the remote interfaces. Querying `YUNit` returns `DIVS` when the units of the trace are indeterminate or undefined.

Examples: `WFMP YUN:VOL`

Link: `YZEro: <NRx>`

Arguments: `<NRx>` can range between −1E+15 and 1E+15.

`YZEro` specifies the vertical offset of a YT trace, or specifies the vertical offset of the vertical component of an XY trace.

Examples: `WFMP YZE:6.25E+1`

Query Note: The `WFMPre?` query returns its links in the following order:

```
WFMPRE WFID:<arg>, LABEL:<qstring>, BIT/NR:16,
    BN.FMT:RI, BYT/NR:2, BYT.OR:<arg>, CRVCHK:<arg>,
    ENCDG:<arg>, NR.PT:<NR1>, PT.FMT:<arg>,
    RHOFACTOR:<NR3>, RHOPOS:<NR3>, XINCR:<NR3>,
    XMULT:<NR3>, XUNIT:<arg>, XZERO:<NR3>, YMULT:<NR3>,
    YUNIT:<arg>, YZERO:<NR3>, TIME:<qstring>,
    DATE:<qstring>
```

WFMScaling { FORce | OPTional }

Group: Trace and Settings Commands

WFMScaling determines whether a new trace is created in floating-point mode (FORce) or integer mode when possible (OPTional). When WFMScaling is set to FORce, all traces except single channel acquisitions (e.g., M1, M2), are created in floating-point mode. Integer mode implies that no floating-point operations are used to display or position traces. Certain trace types require floating-point mode or integer mode, regardless of the WFMScaling setting. (For example, stored traces are stored in floating-point mode.)

NOTE

Traces created in integer mode have faster display update rates.

You can display the trace description types listed in Table 2-71 in integer mode.

Table 2-71: Trace Types Displayable in Integer Mode

Description	Example
A channel (*<slot><ui>*)	M1
Average of a channel	AVG(M1)
Envelope of a channel	ENV(M1)
Inversion of a channel	−M1
Addition of channels	M1+M2
Subtraction of channels	M1−M2
Combinations of the above	AVG(M1+M3)

Table 2-72 lists some of the trace types that you cannot display in integer mode.

Table 2-72: Traces Not Displayable in Integer Mode

Trace Type	Example
Stored trace	STO11
Scalar value	2.23
Stored trace plus scalar value	STO11+2.23
Any trace using division	M1 /M3
Any trace using multiplication	M1 * M2
Any trace using a floating-point function	DIFF(M4)

Examples: WFMS OPT

WIDth?

Group: Measurement Commands

Query Only. WIDth? returns (for the selected trace) the time a signal takes to go from one MESial voltage level crossing to the next MESial crossing of the opposite slope, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **WID?**
WIDTH 5.009E−7,EQ

WIN<*ui*> <*link*>**:**<*arg*>

Group: Trace and Settings Commands

WIN sets and queries window parameters for the specified trace. WIN is not valid if the specified trace is a Main trace.

Link: ? ABSLevel <*NRx*>,<*qual*>

Arguments: <*NRx*> can range between −2 V and +2 V. <*qual*> is described below.

Query Only. ? ABSLevel returns the specified autowindow transition level in absolute units (volts or rho).

The <*qual*> qualifiers have the following meanings:

TF	Transition Found
OR	Out of Range
NF	Not Found

Examples: **WIN2? ABSL**
WIN2 ABSLEVEL:−1.7,TF

Link: ? BASeline <*NRx*>

Query Only. ? BASeline returns the baseline for an autowindow in non-tracking relative mode.

Examples: **WIN2? BAS**
WIN2 BASELINE:7.8E−1

Link: FILtering: <*NRx*>

Arguments: <*NRx*> can range between 1 and 7.

FILtering sets the hardware filtering constant (the minimum number of adjacent samples which must cross the threshold level before a transition is considered valid) for autowindow placement.

Examples: WIN2 FIL:6

Link: `LEVel: <NRx>`

Arguments: The range for `<NRx>` is described below.

    `LEVel` sets the transition level for autowindow placement. The range for `LEVel` when `LMOde` is `RELative` is 0% to 100%. When `LMOde` is `ABSolute`, the range is −2 V to +2 V.

Examples: `WIN2 LEV:65`

Link: `LMOde: { ABSolute | RELative }`

    `LMOde` selects the transition level mode for autowindow placement. In `RELative` mode, the transition level is a percent of the Main trace amplitude. In `ABSolute` mode, the transition level is specified in absolute units (volts).

Examples: `WIN2 LMO:REL`

Link: `LOCate`

    Set Only. Locate causes the instrument to perform a one-time track of the specified trace, which must be a non-tracking autowindow.

Examples: `WIN2 LOC`

Link: `MODe: { AUTO | MANual }`

    `MODe` selects the mode of setting the horizontal position. In `AUTO` mode, the instrument locates the window using the `FILtering`, `LEVel`, `LMOde`, `SLOpe`, and `TRANS` links. In `MANual` mode, the window position is specified by the `POS` link.

Examples: `WIN2 MOD:AUTO`

Link: POS: *<NRx>*

Arguments: The range for *<NRx>* is described below.

POS sets the horizontal position. The Window trace must be completely contained within the Main record. Therefore, the valid range for this link depends on HREfpt, MAINPos, *main duration* and *window duration*.

Range when HREfpt is LEFt:

    MAINPos to
    MAINPos + *main duration* − *win duration*

Range when HREfpt is CENter:

    MAINPos − *main duration* /2 + *win duration* /2 to
    MAINPos + *main duration* /2 − *win duration* /2

Range when HREfpt is RIGht:

    MAINPos − *main duration* + *win duration* to
    MAINPos

Examples: WIN2 POS:1.35E−1

Link: SLOpe: { PLUs | MINUs }

SLOpe selects the slope of the transition for autowindow placement.

Examples: WIN2 SLO:PLU

Link: ? TOPline *<NR3>*

Query Only. TOPline returns the topline for an autowindow in non-tracking relative mode.

Examples: **WIN2? TOP**
WIN2 TOPLINE:1.5E+0

Link: TRACK: { ON | OFF }

TRACK sets autowindow tracking to ON or OFF. When ON, the Window position is determined by the instrument each time before it acquires the Window record. When tracking is OFF, the window is located only when MODe is changed to AUTO, or when the LOCate link is sent.

Examples: WIN2 TRACK:ON

Link: TRANS: *<NRx>*

Arguments: *<NRx>* can range between 1 and 15.

TRANS sets the number of the transition for autowindow placement.

Examples: WIN2 TRANS:5

WINList?

Group: Trace and Settings Commands

Query Only. WINList? returns the trace number(s) of the defined Window traces.

Examples: **WINL?**
WINLIST WIN2,WIN6

WINNum?

Group: Trace and Settings Commands

Query Only. WINNum? returns the number of defined Window traces.

Examples: **WINN?**
WINNUM 2

YTEnergy?

Group: Measurement Commands

Query Only. YTEnergy? returns for the selected trace the energy (in squared volts) under the curve of a YT trace, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: **YTE?**
YTENERGY 8.442E-7,EQ

YTMns_area?

Group: Measurement Commands

Query Only. `YTMns_area?` returns for the selected trace the difference between the area under a YT curve above a specified reference level, and the area under the curve below that level, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.) The reference level is set with the `REFLEvel` command.

Examples: `YTM?`
`YTMNS_AREA 3.332E-7,EQ`

YTPls_area?

Group: Measurement Commands

Query Only. `YTPls_area?` returns for the selected trace the total, absolute value of all areas between a YT trace and a reference level set with `REFLEvel`, followed by an accuracy qualifier. (Refer to page 2-168 for qualifier definitions.)

Examples: `YTP?`
`YTPLS_AREA 1.052E-9,EQ`

Status and Events

The CSA 803C and 11801C provide a status and event reporting system for the GPIB and RS-232-C interfaces. The status and event system alerts you to significant conditions and events that occur within the instrument.

The status and event system has two principal subsystems:

- The status reporting subsystem is based on the service request (SRQ) function defined by IEEE STD 488 for the GPIB interface. It provides a single byte of general status information. For the RS-232-C interface, the `STByte?` query command provides essentially the same function.

- The event reporting subsystem is defined by the Tektronix Codes and Formats Standard using the `EVENT?` query command. This query provides more detailed information about the specific event that has occurred. The `EVENT?` response may be reported to either the GPIB or the RS-232-C interface.

A controller always has the option of reading or ignoring the event code(s) associated with a given status byte.

Status Reporting

The status reporting subsystem includes:

- Status Byte for conveying the type of event that has occurred

- RQS command for GPIB asynchronous service requests and status messages

- `SRQMask` command for masking event conditions

- `STByte?` query for RS-232-C polled status messages

- `RS232 VERBose` command for RS-232-C asynchronous status messages

- System Status Conditions for the categories of events that are reported, such as command errors and internal warnings.

Status Byte Definition

Table 3-1 describes the individual bits in the status byte. Bit 8 is the most significant bit of the status byte. DIO is an IEEE STD 488 abbreviation for Data Input Output.

Table 3-1: Status Byte Definitions

DIO Bit #	Meaning
1 2 3 4	System status bits. The state of these four bits varies with the type of event that is reported.
5	Busy bit. Asserted only when diagnostics are in progress.
6	Error bit. Asserted when an internal or external error condition generates an event.
7	RQS (request service) bit. Asserted when the instrument requests service from a GPIB controller.
8	Never asserted (bit DIO8 is always 0).

RQS Command

The IEEE STD 488 Service Request function (SRQ) permits a device to asynchronously request service from a GPIB controller whenever the device detects some noteworthy event. A GPIB controller services the request by serial polling each active device on the bus. A device responds to the serial poll by placing an 8-bit status byte on the bus. The controller determines which device asserted SRQ by serially reading the status byte of each device and examining bit 7. If a particular device has requested service, bit 7 of its status byte is set. Otherwise, bit 7 is clear. (Refer to the Table 3-2, Binary and Decimal Status Byte Codes). The RQS command turns on the SRQ function in the instrument.

RQS only affects status and event reporting at the GPIB port. RQS has two major effects:

- It controls bit DIO7 of the status byte. The RQS ON command enables DIO7 assertion. The RQS OFF command disables assertion for all conditions except power-on. At power-on, RQS is on at the GPIB port and off at the RS-232-C port.

- The RQS command also controls whether or not the instrument is permitted to request service from a GPIB controller. The RQS OFF command disables service requests. The RQS ON command enables service requests.

RQS for GPIB Service Requests — causes the instrument to assert the SRQ signal line whenever a new event occurs and RQS is set to on. A GPIB controller may then interrogate the instrument with an IEEE STD 488 serial poll and obtain a status byte that describes the event that occurred.

When RQS is set to OFF, the only new event that will cause the instrument to assert SRQ is power-on. Thus, a GPIB controller will not be informed asynchronously (with SRQ) that an event has occurred. In this situation, a controller may still interrogate the instrument with an IEEE STD 488 serial poll to read the most recent status byte from the serial poll register of the instrument.

RQS for RS-232-C Service Requests — is always set to off at the RS-232-C port. There is no SRQ signal line for the RS-232-C interface. No asynchronous messages are sent to the controller. Thus, an RS-232-C controller is required to query (poll) the instrument to determine the latest status condition that has occurred in the instrument.

SRQMASK Command

Regardless of whether RQS is on or off, there may be occasions when you want to disable event reporting for a specific class of system conditions. Use the SRQMask command to disable (mask off) a specific category of events. The event tables later in this section include the SRQMask for each event type.

STByte? Query-Only Command

The STBYyte? query allows RS-232-C controllers to read the status byte of the most recent event reported to the RS-232-C port.

The response to the STByte? query is:

    STBYTE <*NR1*>

where <*NR1*> is a decimal number representing a status condition.

RS232 Verbose Mode

RQS is always off for the RS-232-C interface. Therefore, no new event will cause the instrument to request service.

However, in addition to polling the instrument using the STByte? query, the RS-232-C interface includes another means to synchronously report status messages, RS232 VERBose mode. This mode is turned on or off by using either the GPIB/RS232C pop-up menu in the front panel Utility major menu, or the RS232 VERBose command.

When VERBose is set to ON, each command sent to the instrument always returns an appropriate status message. (For more information on verbose mode, see the discussion on page 2-19.)

System Status Conditions

The status byte indicates nine system status conditions. System status conditions are divided into two categories: normal (DIO6 clear) and abnormal (DIO6 set).

There are four normal conditions defined:

- **No Status To Report** reports when there is no event or device dependent status to report.

- **Power On** reports when the instrument has finished its power-on sequence.

- **Operation Complete** tells the controller that a time-consuming task has been completed.

- **User Request** reports when the RQS icon is selected at the front panel.

There are five abnormal conditions defined:

- **Command Error** reports when a message cannot be parsed or lexically analyzed.

- **Execution Error** reports when a message is parsed but cannot be executed.

- **Internal Error** reports if the instrument malfunctions.

- **Execution Warning** reports when the instrument is operating, but you should be aware of possibly inaccurate results.

- **Internal Warning** reports when the instrument detects a problem. The instrument remains operational, but the problem should be corrected.

Status Byte Codes

Nine status conditions are reported in the status byte — four normal events and five error or warning events.

Table 3-2 gives the status byte codes in binary and in decimal with both RQS set to ON and set to RQS OFF. The bits in the status byte are active high. Bits 1 through 4 are system status bits. Bit 5 is the busy status bit and is asserted only during diagnostics. Bit 6 is the error bit. Bit 7 ("R" in the table) indicates whether RQS is set to ON (high) or OFF (low). Bit 7 is low until specifically enabled with the RQS command. Bit 8 is always low.

Table 3-2: Binary and Decimal Status Byte Codes

	BINARY		DECIMAL	
	Status Bits		RQS	RQS
Condition	**8765**	**4321**	**ON**	**OFF**
Normal:				
No Status to Report	0000	0000	0	0
Power On	0R00	0001	65	1
Operation Complete	0R00	0010	66	2
User Request	0R00	0011	67	3
Abnormal:				
Command Error	0R10	0001	97	33
Execution Error	0R10	0010	98	34
Internal Error	0R10	0011	99	35
Execution Warning	0R10	0101	101	37
Internal Warning	0R10	0110	102	38

Event Reporting

The second subsystem is event code reporting. Event messages expand the description of the status condition reported by the status byte to more clearly specify the event that has occurred.

GPIB and RS-232-C controllers may read event codes generated by the instrument by using the `EVENT?` query-only command.

The response to an `EVENT?` is either:

    EVENT <*NR1*>

or

    EVENT <*NR1*>,<*qstring*>

where <*NR1*> represents the numerical value of an event code, and <*qstring*> is a quoted string that describes the returned event code.

The response that includes the quoted description string is returned only when the `LONgform` command is set to `ON`.

Event Code Descriptions

All event codes and event code description strings for all event classes are listed beginning on page 3-8. The event code and event code description is in boldface. Commands that can generate the event code are listed immediately after.

Formatting Symbols — such as %A are combined in some of the description strings in the event code tables. When the event is queried, the formatting symbol(s) are expanded, as described on the next page.

Each formatting symbol begins with a percent sign (%). The symbols indicate that variable information will be substituted when `LONgfrom` is set to `ON`.

The formatting symbols and their meanings are shown in Table 3-3.

Table 3-3: Formatting Symbols

Symbol	Expand With:
%a	Channel number or unsigned integer
%A	Argument name
%b	Mainframe indicator: M
%B	Mainframe verbose indicator
%C	Calibration request string: "Calibration due"
%O	Option description string (e.g., "Prescaler deleted")
%t	Six words of data describing a mainframe malfunction intended for use by field and factory service.

For example, the following set command causes an execution warning, event code 550:

```
CHM1 OFFSET: 5000
```

Event code 550 has this entry in Table 3-5 Execution Warnings:

Code	Event Description
550	%A out of range-limit set

If `LONgform` is set to `OFF`, the event is reported:

```
EVENT? <EOI>
EVENT 550
```

If is set to `ON`, the response is:

```
EVENT? <EOI>
EVENT 550,"OFFSET OUT OF RANGE — LIMIT SET"
```

Command Errors

Command errors are reported when a message cannot be parsed or lexically analyzed. Command errors have event codes from 100 to 199. The `SRQMask` for command errors is `SRQMASK CMDERR`. The status byte for a command error returns **97** (decimal) with `RQS` set to `ON`, and **33** (decimal) with `RQS` set to `OFF`. All command errors are listed in Table 3-4.

Table 3-4: Command Errors

Event Code	Description	Commands that Generate Code	Explanation
108	Checksum error in binary block transfer	`SET <bblock>`	Checksum comparison of binary settings failed. Settings are discarded.
109	Illegal byte count value on a binary block transfer	`SET <bblock>` `CURVe <bblock>`	Binary block byte count of binary settings returned to the instrument exceeds maximum size of front panel settings.
154	Invalid number input		Floating-point value too large or excessively long.
155	Invalid string input		String is too long, is not properly terminated, or contains a NULL character.
156	Symbol not found		Oscilloscope is unable to find the input symbol in its table.
157	Syntax error	Any command	Command was typed incorrectly.
		`RQS`	Attempted to turn `RQS` on at RS-232-C port.
		`STByte?`	Attempted to use `STByte?` query from GPIB port.
		`TESt`	Set or query command appended to `TESt` command. `TESt` command is ignored; all other commands are processed normally.
		`TRAce<ui>`	Syntax error found in `TRAce` expression (for example, "M1 +", or attempted to create non-acquired trace component (for example, `STO <ui>, <NRx>`, or combinations) on `WIN` time base.
160	Expression too complex	`TRAce<ui>`	Trace description exceeds 54 characters for either the vertical or horizontal description, or cannot be parsed due to insufficient stack space.
161	Excessive number of points in binary CURVe data input	Waveform Retrieval and Scaling (Data Transfer)	More binary data points were sent than were specified with the `WFMPRE NR.PT` link.
162	Excessive number of points in ASCII CURVe data input	Waveform Retrieval and Scaling (Data Transfer)	More ASCII data points were sent than were specified with the `WFMPRE NR.PT` link.

Table 3-4: Command Errors (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
163	No input terminator seen		RS-232-C input type-ahead buffer has overflowed. All input is discarded.
164	Binary block input not allowed with ECHO ON	`CURVe <bblock>` `SET <bblock>`	Attempted to send binary block data through RS-232-C port with echo on. The data are discarded.
167	Insufficient data to satisfy binary block byte count	`SET?`	Binary settings returned to GPIB port prematurely terminated (for example, binary block byte count not satisfied when EOI line is asserted).
168	Unsupported constant		
169	Unsupported function	`TRAce<ui>`	`TRAce` expression includes unsupported function.

Execution Errors

Execution errors are reported when a message is parsed but cannot be executed. Execution errors have event codes from 200 to 299 or 2000 to 2999. The SRQMask for execution errors is SRQMASK EXERR. The status byte for an execution error returns **98** (decimal) with RQS set to ON, and **34** (decimal) with RQS set to OFF. All execution errors are listed in Table 3-5.

Table 3-5: Execution Errors

Event Code	Description	Commands that Generate Code	Explanation
200	No masks defined	MASK, MASKStat	No masks defined for mask command
201	Display type not color graded	DISPlay	User tried to query display data when not in Color graded.
202	DC coupling not allowed	TRIgger	User tried to set trigger coupling. Not supported on CSA 803C.
203	I/O buffers full		Both input and output buffers are full. Output buffer is cleared.
204	No selected measurement	MEAS	No measurement in the MSLISt.
205	%A out of range − value ignored	ABSTouch	Out-of-range ABSTouch argument.
206	No mask point near enough to delete		Front Panel only command.
207	No such mask	MASK	Mask<*ui*> is not defined.
208	Illegal number of mask vertices	MASK	Input incorrect number of mask vertices. Values not paired for XY point (i.e., nrx1, nry1, nrx2, needs an nry2), or 0 points or > 50 points.
209	Mask point not on screen	MASK	MASK point in input array is off screen.
210	Illegal mask number	MASK<*ui*>	User referred to MASK<*ui*> where *ui* was less than 1 or greater than 10.
211	Setting cursor 1 to zero not permitted on XY trace	CURSor SETZero	Tried to set cur1 zero on an XY trace.
212	Waveform not scaled in Rho units	CURSor	User specified split cursors between Rho and non-Rho wfms.
213	Set zero Rho allowed only with Rho units	RHOZero	Waveform not scaled in Rho.
214	Invalid filter argument	TRAce	Filtering argument in trace command incorrect.
215	Illegal color number	COLor, HPGl	Out-of-range color index.
216	Unsupported printer function	COPy	Format unsupported for currently selected printer.

Table 3-5: Execution Errors (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
217	Not enough memory for alternate topline-baseline calculations	`TOPBase`	Out of memory for calculating topline-baseline in Alternate method.
218	No logical names defined	`DEFine?`	No logical names currently defined.
219	Logical name not defined	`DEFine?, UNDEF`	User asked for a logical name that was not defined.
220	WFMPre links not allowed with multiple trace output	`WFMPre, WAVfrm`	User had `OUTput` set to `TRA<ui>TOTRA<ui>` or similar and asked for `WFMPre`.
221	That XY waveform has incompatible components	`TRAce`	Waveform has calculated vs stored or other incompatible type.
222	Record length too long for Non-Normal display type	`DISPlay, HISTogram MASKStat`	User changed display mode (infinite, variable, or graded) with record length > 512.
223	Cannot select Rho units	`GRAticule`	Tried to change to Rho units when either External Prescaler trigger or TDR not turned on.
225	Baseline correction failed to find −1 rho point		Baseline correction couldn't find −1 rho this calibration pass.
226	Rho scaling failed to fine −1 rho point − Cannot calibrate rho scales	`AUTOSet`	Could not scale waveform in Rho accurately.
227	Operation not allowed on a saved trace description	`ADJtrace?`	`ADJtrace<ui>?`; range of `ui` is 1 to 8.
228	Waveform expression too long	`TRAce, ENV, AVG`	Trace description exceeded 55 characters.
229	No windows are defined	`WIN, WINNum?, WINList`	No window waveforms are defined.
230	Illegal window number	`WIN`	`WIN<ui>`; `ui` out of range.
231	No such window	`WIN<ui>`	No such `WIN<ui>` trace is displayed.
232	Measurements not available in current mode have been removed	`<meas>, MPAram`	Measurement not allowed in Hardware mode.
233	Measurement not found in measurement list	`<meas>`	Measurement not in measurement list (`MSLIst`).
234	Calculated or stored traces cannot be measured in hardware mode	`<meas>, MPAram, TRAce`	Only actively acquired traces can be measured in the hardware mode.
235	Waveform cannot be measured in software mode	`MSLIst`	Some `<meas>` in list not allowed in software measurement mode.

Table 3-5: Execution Errors (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
236	No reference channel selected	`CURSor`, `MPAram`	No reference channel selected for operation.
237	Waveform is not a non-track- ing autowindow	`WIN`	Tried to do a locate or other autowindow func- tion.
238	Window is not an auto-window	`WIN`	Tried to do a locate or other autowindow func- tion.
239	Improper version number	`SET <bblock>`	Version number of received binary settings block not the same as current firmware version number.
240	Set zero Rho not allowed with split cursors	`RHOZero`	Tried to calibrate the rho scale of the selected trace when cursors were split between two traces.
241	Too many acquisitions	`TRAce<ui>`	Trace definition would cause the instrument to acquire more than eight traces.
242	Requested display type not allowed when calculating histograms, masks or statistical measurement	`DISPlay TYPe`	Cannot have Variable or Normal display type when acquiring histograms or masks. Cannot have any display type other than color graded when performing statistical measure- ments.
243	That function is disabled by a hardware strap	`UID`	Attempted to modify serial number.
244	Histogram/mask system not active	`HISTogram`, `MASK`, `MASKStat`, `STATHist`	Tried to manipulate Histograms or masks with- out `DSYS` being on.
245	No stored waveforms	`LABel? STO`	No waveforms were stored when `LABel` was queried or no free memory for stored wave- forms.
246	Can't sequence settings	`RECall`	Attempted to sequence settings with `SET- Seq OFF`.
247	No settings defined	`LABel?`, `RECall`, `SETSeq`	Attempted one of these commands when no stored settings exist.
248	Misuse of `AVG`/`ENV` function	`AVG`, `ENV`	Attempted to turn `AVG` or `ENV` on when se- lected trace is XY, or when selected trace is composed only of stored and scalar compo- nents. Or attempted to turn `AVG` or `ENV` off when selected waveform's vertical description not enclosed by the `AVG` or `ENV` function.
249	Illegal use of trace position- ing function	`ADJtrace <ui>`	Attempted to modify `HMAg`, `HPOsition`, `TRSep`, `VPOsition`, or `VSIze` values when modification is not permitted (for example, when `PANzoom` is off).

Table 3-5: Execution Errors (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
250	No traces defined	ADJtrace?, LABel?, TRAce?	Query attempted with no traces displayed.
		LABel	Attempted to label or delete a label on a trace when no traces are currently displayed.
		AVG, CURSor, DOT1Abs, DOT2Abs, DOT1Rel, DOT2Rel, ENV, H1Bar, H2Bar, V1Bar, or V2Bar	Attempted to set or query one of these commands with no traces defined.
		MPAram	Attempted to set or query a measurement parameter with no selected trace.
251	Illegal trace number	ADJtrace, CLEar, CURSor, LABel, OUTput, REMove, SELect, STORe, TRAce	Out-of-range $<ui>$ argument with one of these commands.
252	Illegal stored settings number	DELete, LABel, RECall, STORe	Setting $<ui>$ does not exist.
253	Stored setting does not exist	DELete, LABel, RECall, STORe	Out-of-range $<ui>$ argument with one of these commands.
254	Histograms and masks are not allowed on calculated or stored traces	HISTogram, MASKStat, TRAce	Histogram/mask turned on for calculated or stored trace.
255	Out of memory	STORe	Insufficient memory to store a trace or insufficient NVRAM to store settings.
		TRAce	Insufficient memory to create a new trace.
		Waveform Retrieval and Scaling (Data Transfer)	INPut command references nonexistent stored waveform, insufficient memory to create stored waveform record.
		WFMPre	Insufficient memory to create stored waveform record for preamble.
256	Label not found	CLEar, DELete, INPut, OUTput, RECall, REMOVe, SELect, STORe	No matching label found with $<qstring>$ syntax used.
257	Illegal stored waveform number	DELete, INPut, LABel, OUTput, STORe, TRAce	Out-of-range STO $<ui>$ argument for one of these commands.
258	Duplicate label — label not changed	LABel	Label specified already used.

Table 3-5: Execution Errors (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
259	No labels defined	`LABel`	No labels defined for specified links.
260	Label not defined	`LABel?`	No label is defined for requested front panel setting, stored trace, or acquired trace.
261	Trace is not a window	`WIN`	Trace referred to is not a window.
262	Waveform must have acquired components	`VPCurve, DISPlay`	The waveform is not active so Infinite, Variable, and Color Graded have no meaning.
263	Illegal channel number	`CH <alpha> <ui>`	Attempted to set parameters of sampling head channel that was out of range.
264	No further XY waveforms may be defined	`TRAce<ui>`	Attempted to define more than the maximum permissible number of XY traces.
265	Illegal `DATE/TIME`	`DATE`	Illegal date value or syntax specified.
		`TIMe`	Illegal time value or syntax specified.
266	`DEF` expansion overflow	`DEFine`	Expansion string overflowed internal expansion buffer.
267	Illegal `DEF` string	`DEFine`	Illegal logical name specified.
268	Illegal `DEF` recursion	`DEFine`	Unacceptable `DEFIine` recursion detected. Recursive logical names are acceptable only when recursion occurs to the right of an unquoted semicolon.
269	No such trace	`ADJtrace, CLEar, CURVe, LABel, REMove, SELect, STORe, TRAce, WAVfrm?, WFMPre`	Referenced, or attempted to set or query parameters of a nonexistent trace using one of these commands.
270	No such stored waveform	`CURVe?`	`CURVe?` query attempted, `OUTput` references nonexistent stored waveform.
		`DELete`	Attempted to delete nonexistent stored waveform.
		`LABel`	Attempted to label or query for a label of a nonexistent stored waveform.
		`TRAce <ui>`	`TRAce` expression referenced legal but undefined stored waveform.
		`WAVfrm?`	`WAVfrm?` query attempted, `OUTput` referenced nonexistent stored waveform.
		`WFMPre?`	`WFMPre?` query attempted, `OUTput` referenced nonexistent stored waveform.
271	No such `DEF`	`UNDEF, DEFine`	Argument not defined in current list of logical names.

Table 3-5: Execution Errors (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
272	Channel %b%a is not capable of this function	CH, CALibrate	Tried to adjust a function not supported by that sampling head channel.
273	No such FPS	DELete	Attempted to delete undefined stored settings number.
		LABel	Attempted to label or query undefined stored settings number.
		RECall	Attempted to recall undefined stored settings number. In this context, "undefined" refers to previously deleted settings or settings that have never been initialized.
274	Automatic window mode not allowed for calculated traces	WIN	Auto-windows not allowed on calculated traces.
275	%B not installed		
276	No variable persistence traces defined	VPCurve	Asked for Variable Persistence data when not in Variable Persistence display mode.
277	No histogram selected for trace	STATHist	No histogram is defined on the selected waveform.
278	Measurement system not active	STATIstics, <*meas*>?	A measurement query was sent when the Measure major menu was not selected. Use the MSYs ON command.
279	Bad trace description	TRAce<*ui*>	Trace description was contained unrecongizable arguments.
280	Invalid smoothing argument	TRAce<*ui*>	TRAce expression contains out-of-range SMOOTH argument.
281	Can't delete active stored waveform	DELete	Attempted to delete stored waveform that is a component of a combined active trace.
		WFMPre	Returning WFMPre data would cause deletion of a stored waveform that is not the sole component of a waveform description of a displayed trace. The WFMPre data are discarded.
282	Can't store trace	STORe	Attempted to store XY trace, or attempted to copy a trace over an existing stored waveform when the two waveforms do not have equal record lengths.
283	Can't clear nonacquired waveform	CLEar	Attempted to clear trace that has only stored trace components (for example, TRACE1 DESCRIPTION:"STO3") or is scalar (LOG(5)).
284	Can't change setting for a tracking/autowindow	MPAram	Tried to set top/base lines for auto/tracking window.

Table 3-5: Execution Errors (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
285	Cannot perform measurements on XY waveforms	`<meas>`, `MSYs`, `MPAram`	No automatic measurements are allowed on XY traces.
286	Too many measurements specified	`MSLIst`	More than six measurements specified.
287	Hardcopy absent or off line	`COPy`	`CENTRonics` port specified as `COPy` output port, printer not connected to port or currently connected printer is offline.
288	Inappropriate trigger level units	`TRIgger`	Improper `LEVel` units specified.
289	Split cursors not permitted on XY trace	`CURSor`	Attempted to `SPLit` cursors across XY trace.
290	Current reference measurement failed	`REFSEt`	`CURRent` reference cannot be computed due to one of the following conditions: ■ Selected waveform is XY (regardless of measurement). ■ Reference measurement specified as `DUTy`, `FREq`, or `PERiod`; no period can be found within specified measurement zone. ■ Reference measurement specified is `MEAN`, `RMS`, `YTEnergy`, `YTMns_area`, or `YTPls_area` while `DAint` is set to `SINgle`; no period can be found within specified measurement zone. ■ Reference measurement specified is `CROss` and `REFLEvel` does not fall between computed maximum and minimum of specified measurement zone. ■ Reference measurement specified is `RISe-time` and measurement system cannot compute valid proximal and distal time within specified measurement zone. ■ Reference measurement specified is `FALl-time` and measurement system cannot compute valid distal and proximal time within specified measurement zone. ■ Reference measurement specified is `WIDth` and two mesial crossings of opposite slope cannot be found within specified measurement zone. ■ Reference measurement specified is `PHASE` and no period can be found within the reference waveform measurement zone.

Table 3-5: Execution Errors (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
			■ Reference measurement specified is `PHASE` and valid mesial crossings cannot be found within the measurement zones of the reference and selected waveforms.
			■ Reference measurement specified is `PDELAY` and valid mesial crossings cannot be found within the measurement zones of the delayed and selected waveforms.
291	TEXT not permitted when acquired XY trace is active	`TEXt`	Attempted to place text on display when acquired XY trace is active.
292	No sampling heads are installed in %B	`UID`	Attempted to query the serial number of a sampling head channel that is nonexistent.
293	No sampling heads are installed	`UID`	Attempted to query the serial number of a sampling head channel that is nonexistent.
294	Dual graticules not permitted with XY trace	`GRAticule`	XY traces allowed only on single graticule display.
295	%O option not installed	`TRIgger`	Tried to set trigger source to Prescaler without prescaler installed.
296	Nonacquired XY waveforms are not allowed in Non-Normal display type	`TRAce, DISPlay`	Non-acquired XY waveforms only allowed in Normal display type.
297	Panzoom may not be enabled	`ADJtrace<ui>`	Attempted to enable `PANzoom` for XY trace.
298	Panzoom may not be disabled	`ADJtrace<ui>`	Attempted to disable `PANzoom` for stored or scalar trace, or for FFT magnitude phase traces.
299	CONDacq function not available	`CONDacq`	`AVG` or `ENV` conditional acquisition specified, but no traces include `AVG` or `ENV` function in trace descriptions.
			`CONDacq` set to `AVG.ENV`, but the following condition does not exist:
			■ At least one waveform description includes the `AVG` function and at least one other waveform description includes the `ENV` function.
			■ One waveform includes both `AVG` and `ENV` in its description.
			Conditional acquisition of any type except `CONtinuous` specified, with no traces defined.

Table 3-5: Execution Errors (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
2000	Statistical Measurements allowed only in Color Graded mode	`<meas>, MPAram`	Attempted to query/set statistical measurements when display type is other than Color Graded.
2001	Only one waveform per axis can be measured in Statistics mode	`TRAce DE-SCRIPTION, MSLISt, DISPLAY GRAT-ICULE`	Can measure only one waveform per axis when measurement mode is set to Statistical.
2002	%0 not available	`TRIgger, STD, RECall, STORe, CALibrate, TRAce`	Attempted to set/query an item that is not available/installed in the instrument.
2003	Smoothing is not permitted when either channel in a head is in Random Data mode	`CH <alpha><ui> SMOothing`	Attempted to turn smoothing on for a channel when Random Data mode is turned on for **either** channel in the head. Smoothing and Random Data mode are incompatible.
2004	The standard was violated. The topline (with simulated AC coupling) exceeds 0.55 V	`AUTOSet STARt`	The CITT standard at 139 megabyte has a topline that exceeded 0.55 V above ((topline + baseline) / 2).
2005	User-defined standard mask must use Mask #1, Mask #2, and Mask #3	`USEREye CREate`	User-defined standard masks must use Mask #1, Mask #2, and Mask #3.
2006	The user-defined standard mask set can contain a total of no more than 25 vertices	`USEREye CREate`	The user-defined standard mask set can contain no more than 25 vertices.
2007	Masks must contain at least 3 points to be valid	`USEREye CREate`	Masks must contain at least 3 points.
2008	Mask #1 must be above Mask #2. Mask #3 must be below Mask #2	`USEREye CREate`	Mask #1 must be above Mask #2. Mask #3 must be below Mask #2.
2009	Masks #1 and #3 must go to the extreme outside edge on their last two vertices	`USEREye CREate`	Masks #1 and #3 must go to the extreme outside edge on their last two vertices. Refer to the *User Manual*.

Internal Errors

Internal errors are reported if the instrument malfunctions. Internal errors have event codes from 300 to 399. The `SRQMAask` for internal errors is `SRQMASK INERR`. The status byte for internal errors returns **99** (decimal) with `RQS` set to `ON`, and **35** (decimal) with `RQS` set to `OFF`. All internal errors are described in Table 3-6.

Table 3-6: Internal Errors

Event Code	Description	Commands that Generate Code	Explanation
386	Minor time base calibration problem: %a		Time base couldn't calibrate to optimum *this* calibration pass.
387	Time base calibration failed: %a		Time base couldn't calibrate to optimum *this* calibration pass.
388	Time base calibration failed at powerup: %a		Time base couldn't calibrate to optimum *this* calibration pass.
390	Time base system error: %t		Time base internal error.
391	Time base processor interrupt: %t		Time base internal error.
393	Acquisition memory fault: %a		Acquisition internal error.
394	Test completed and failed	TESt	Self-tests or extended diagnostics completed and failed.
395	Error detected in acquisition system #%a: %t		Acquisition system detected an internal error.

System Events

System events are normal conditions of the system and are listed in Table 3-7. System events have event codes from 400 to 499. The `SRQMask` for each event is included in the table.

NOTE

Event 400 (system function normal) and event 401 (power on) cannot be masked with `SRQMask`*.*

Table 3-7: System Events

Event Code	Description	SRQMASK	Status Bytes		Commands that Generate Code
400	System function normal	—none—	0	0	
401	Power on	—none—	65	1	
403	Front panel RQS icon selected	USEr	67	3	
441	Store constants complete on selected channel	USEr	67	3	
442	Calibrate ONCE complete	USEr	67	3	
443	Blowby calibration complete	USEr	67	3	
444	Loop gain calibration complete	USEr	67	3	
445	Delay adjust calibration complete	USEr	67	3	
446	Delay adjust measurement on reference channel complete	USEr	67	3	
447	Offset null calibration complete	USEr	67	3	
448	TDR amplitude calibration complete	USEr	67	3	
449	All calibration complete on selected channel	USEr	67	3	
450	Conditional acquire completed	OPCmpl	66	2	CONDacq
451	Front panel setting recall complete	OPCmpl	66	2	RECall
452	Initialization complete	OPCmpl	66	2	INIt
460	Test completed and passed	OPCmpl	66	2	TESt
461	Calibration completed and passed	OPCmpl	66	2	TESt
462	Hardcopy complete	OPCmpl	66	2	COPy
463	Measurements complete	OPCmpl	66	2	
464	Autoset complete	OPCmpl	66	2	AUTOSet
465	Locate complete		70	6	WIN
467	Offset Calibration Complete				CALibrate *<alpha><ui>*

Execution Warnings

Execution warnings are reported when the instrument is operating, but may produce inaccurate results. Execution warnings have event codes from 500 to 599. The `SRQMask` for execution warnings is `SRQMASK EXWARN`. The status byte returns **101** (decimal) with `RQS` set to `ON`, and **37** (decimal) with `RQS` set to `OFF`. All execution warnings are listed in Table 3-8.

Table 3-8: Execution Warnings

Event Code	Description	Commands that Generate Code	Explanation
550	%A out of range − limit set	Any requiring numeric values	A value was included with a command or link argument that was out of range. The limit value nearest the requested value is used.
551	Insufficient data to satisfy binary block byte count	Waveform Retrieval and Scaling (Data Transfer)	Binary waveform data sent to GPIB port prematurely terminated (for example, binary block byte count not satisfied when EOI line asserted). The waveform is filled out with NULL points.
552	Checksum error in binary block transfer	Waveform Retrieval and Scaling (Data Transfer)	Checksum of received binary waveform data does not match checksum of original binary block. The waveform data is retained, regardless of the outcome of the test. *Note: If the binary data was created with a NULL checksum, the checksum test is almost certain to fail. Since the returned data is not discarded, this failure is not important.*
554	No trace defined − Autoset terminated	`AUTOSet`	`AUTOSet` initiated with no traces defined and no signal source can be found (for example, no sampling heads are installed), or the signal being autoset is DC (it has no AC component).
555	Binary curve odd data byte discarded	Waveform Retrieval and Scaling (Data Transfer)	An odd number of data bytes was sent to the instrument and the checksum comparison did not fail.
556	No active acquisitions − acquisition remains stopped	`ACQuisition`	Attempted to start acquisition when no traces are defined, or when no defined traces contain "active" components (as opposed to scalar and stored components).
557	Hardcopy aborted	`COPy`	`COPy` operation aborted.
558	Blowby Calibration failed − can't find reference step	`CALibrate`	
559	Calibration failed − unable to achieve requested value on channel %b%a	`CALibrate`	The sampling head delay adjust value could not be set properly to the requested value.
560	Front panel setting incomplete: Out of memory or change in configuration	`RECall`	

Table 3-8: Execution Warnings (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
561	Offset calibration failed − check channel terminations	CALibrate	
562	Delay adjust calibration failed − can't find transition	CALibrate	
563	TDR amplitude calibration failed − can't find TDR pulse	CALibrate	
564	Autoset failed, bad trigger level	AUTOSet	
565	Autoset vertical failed	AUTOSet	Vertical AUTOSet algorithm detects signal whose DC component is larger than offset range of least-sensitive gain setting of channel.
566	Horizontal size set to default	AUTOSet	Couldn't accurately autoscale horizontal size.
567	Reftrace defaulted to selected trace		
568	Reference trace invalid		
569	Loop gain calibration failed − can't find reference step		
570	Delay adjust measurement on reference channel failed − Can't find transition		
571	Nothing to abort	COPy	Attempted to abort with no copy in progress.
572	Waveform record length(s) reduced to 512 for Non-Normal display type	DISPlay, HISTogram, MASKStat	Record lengths greater than 512 allowed only in normal display type.
573	Persistence time reduced due to lack of memory	TRAce DES, REMOVe, DISPlay	
575	Incompatible firmware versions	ID?	Subsystem firmware versions installed are incompatible.
576	Smoothing has been turned off for both channels in the head; smoothing is incompatible with Random Data mode	CH*<alpha><ui>* DATAtype	Setting data type to random data causes smoothing to be turned off for both channels in head.

Table 3-8: Execution Warnings (Cont.)

Event Code	Description	Commands that Generate Code	Explanation
577	Color graded traces cannot be output on an HPGL device	COPy	
578	A front panel AutoSet will now use a mask-specific algorithm while in the Display major menu	MASKDefine	Using the AutoSet function from the Display major menu will results in the use of a mask-specific algorithm.
579	User mask has been modified. Mask-specific Autoset is no longer available	MASK	The user mask was modified; using the mask-specific AUTOSet is no longer available.
580	Record length reduced to power of 2 for FFT	TRAce DES	For the FFT command, the record length is reduced to a power of 2.
582	Autoset failed. Signal appears to be inappropriate for currently defined standard mask	AUTOSet STARt	The AUTOSet command failed. The signal might be inappropriate for the currently defined standard mask.
583	Vertical units are undefined. The selected math trace contains components with differing scale factors	CH SENsitivity	The vertical units are undefined. The selected math trace contains components with differing scale factors.

Internal Warnings

Internal warnings are reported when a problem has been detected. The instrument remains operational, but the problem should be corrected. Internal warnings have event codes from 600 to 699. The `SRQMask` for internal warnings is `SRQMASK INWARN`. The status byte for internal warnings returns **102** (decimal) with `RQS` set to `ON`, and **38** (decimal) with `RQS` set to `OFF`. All internal warnings are listed in Table 3-9.

Table 3-9: Internal Warnings

Event Code	Description	Commands that Generate Code	Explanation
653	RS-232 input parity error		
654	RS-232 input framing error		
655	RS-232 input buffer overrun		
656	Internal table search failed		
657	Nonvolatile RAM completely reset, probable battery failure	\<Powerup\>	
658	Nonvolatile RAM settings, stored traces, and saved trace descriptions lost; instrument ID data retained	\<Powerup\>	
661	Channel %b%a powered up during mainframe operation − Cycle power to utilize		
662	Channel %b%a powered down during mainframe operation − Cycle power to continue		
663	Change in channel %b%a configuration		
664	Channel %b%a was not maintained at desired calibration delay value		
665	Teksecure Erase Memory Status: Erased; Instrument ID, on-time, and number of power-ups retained	\<Powerup\>	

Examples

The nine examples in this section demonstrate how to perform typical CSA 803C and 11801C functions from the programmable interfaces.

The example programs are divided into sections similar to the front panel operation examples in the *User Reference* for your instrument.

These examples show how to control basic functions remotely
$\left\{\begin{array}{l} \\ \\ \\ \end{array}\right.$
1—Displaying a Trace

2—Managing Multiple Traces

3—Defining Complex Traces

4—Using Signal Processing & Transferring a Trace

These examples use the automated measurement system
$\left\{\begin{array}{l} \\ \\ \\ \end{array}\right.$
5—Taking Automated Measurements

6—Taking Delay Measurements

7—Taking Delay Measurements From a Reference Trace

These examples introduce some advanced features
$\left\{\begin{array}{l} \\ \\ \end{array}\right.$
8—Comparing Displayed Traces to Stored Traces

9—Taking TDR Measurements

Introduction to the Examples

These examples show you how to use the BASIC programming language and the CSA 803C and 11801C command set to perform a variety of remote operations over the RS-232-C interface. The examples require you to have a basic knowledge of your instrument from the front panel, of test and measurement systems, and of programming in the BASIC language.

These example programs will familiarize you with the process of controlling the instrument remotely, and enable you to extend this knowledge to your applications.

For instance, in *Example 1: Displaying a Trace*, the entire program is explained, line by line, so you can see how the CSA 803C/11801C commands combine with the controller's BASIC programming language to perform tasks within an application program.

Organization of the Examples

Each example begins with a brief explanation of its purpose and the setup of necessary equipment and accessories (for example, signal generator and cable configuration). Then each program is listed and explained. You are expected to read the program descriptions as you use the example programs from the disk that is supplied.

Since this is an interactive tutorial, each step of an example prompts you to take an action that performs some task. In most cases, the action is implemented when you press the 'Enter' key on your computer keyboard.

Software and Computer Hardware Needed

The tutorial examples are written for an IBM PC/XT/AT or other IBM PC-compatible computer configured with an RS-232-C interface.

The example programs are supplied on a single, IBM-formatted, 5¼ inch floppy disk at the front of this manual. They run under most common BASIC language implementations, including:

- IBM   BASICA.COM

- IBM   Compiled BASIC, Ver. 1.0 and 2.0

- Microsoft QuickBASIC, Ver. 1.0 to 4.0 or Compiled 6.0

- IBM BASIC.COM

NOTE

Example 4 will not work properly with IBM BASIC.COM because it does not support the WINDOW function used to plot transferred traces. Most other BASIC implementations support the WINDOW plotting function.

Equipment Needed

For these examples you will need a CSA 803C or 11801C with a two channel sampling head installed. An SD-24 is required for *Example 9: Taking TDR Measurements*. Each example begins with an equipment setup diagram that shows the cable connections for that example.

Additional equipment needed includes a signal generator (1 to 50 MHz sine wave, 0.5 V to 1 V output) with a separate trigger output, a power divider, four SMA cables (a 2 ns and a 5 ns cable are recommended), and adapters to connect the SMA cables to the signal generator. A signal-splitting T adaptor may be substituted for the power divider.

Installing the Example Software

To complete the following installation, you will need either a hard disk or a formatted floppy diskette to make a working copy of the example programs. Whenever an example command is shown, press the 'Enter' key to execute the command.

Boot up your computer, carefully install a two channel sampling head in the leftmost *sampling head* compartment, then power-on the CSA 803C/11801C.

Configure the RS-232-C interfaces on the CSA 803C/11801C and on your computer.

■ To configure the CSA 803C or 11801C, press the UTILITY major menu button, then touch the GPIB/RS232C Parameters menu selector. Set the RS-232-C parameters as follows:

Baud	9600	Echo	Off
Stop Bits	1	Parity	NONE
Flagging	HARD	Delay	0
EOL String	CR/LF	Verbose	Off

■ Consult the manual for your computer for information on configuring its RS-232-C interface.

Hard Disk Installation

If your computer has a hard disk, proceed as follows:

■ Create a new directory to receive the example programs, using the MKDIR command from MS-DOS.

```
mkdir examples
```

■ Make the new directory the current directory, using the CD command.

```
cd examples
```

■ To copy the contents of the supplied examples disk to the new directory, place the disk into drive A of your computer. Then type:

```
copy a:*.*
```

This completes the software installation.

Once installation is complete, put the examples disk into the disk jacket in the manual for safekeeping.

Floppy Disk Installation

To install the examples onto a floppy disk in a dual-floppy-drive system, insert the examples disk into drive A:, and a formatted target disk into drive B:.

■ Type the following command:

```
diskcopy a: b:
```

If your computer has a single floppy disk drive, proceed as follows:

■ Type the following command:

```
diskcopy a: b:
```

■ Follow the instructions that appear on the screen in response to the DISKCOPY command, being very careful to insert the supplied tutorial disk when the SOURCE diskette is requested, and to insert a formatted floppy disk when the DESTINATION diskette is requested.

If you intend to use the target floppy disk as a start-up disk (it must be formatted with the /S option in order to do this), copy the following additional files from your original start-up disk onto the target disk: AUTOEXEC.BAT, CONFIG.SYS, and BASIC.COM (or BASICA.COM, or the name of your BASIC program file).

Once installation is complete, put the examples disk into the disk jacket in the manual for safekeeping.

Running the Example Software

You can run the programs in either of two ways: from the MENU program, or individually.

To run the programs from the menu, make sure that the current directory is the directory where the MENU and example programs reside, for example,

```
cd examples
```

Then, enter:

```
basica menu.bas /C:5000
```

or use QB, or the correct invocation for your BASIC application.

Menu.bas is the menu program used to select the examples. The /C: 5000 argument sets up a COM buffer of 5000 bytes for use in *Example 4: Using Signal Processing & Transferring a Waveform*. As an alternative, you can invoke the MENU program with a batch file such as the HELP batch file on the Example disk. Entering

```
help
```

will automatically invoke the MENU program using BASICA MENU.BAS /C:5000. You can modify this batch file with your text editor to use the name of your BASIC application.

The MENU program displays a list of example programs for you to choose from. Type the number of the program you want to run, and press *<ENTER>*.

To run an individual program, make sure that the current directory is the directory where the program resides. Then, type the name of the example program, followed by *<ENTER>*, for example,

```
basica single.bas
```

or QB SINGLE.BAS, or the correct invocation for your BASIC application.

Exiting the Example Software

When an example program completes, you can type:

- *<ENTER>* (which returns you to the MENU program)
- **Q** (which exits the program and leaves you in BASIC)
- **S** (which exits the program and returns control to MS-DOS)

To exit a program without completing it, simply press CTRL-C. This will leave you in BASIC (most likely with a disabled front panel; see below). After re-enabling front panel operation, execute a SYSTEM command to return to MS-DOS.

MENU.BAS Program

The MENU.BAS program is the starting point for the examples. After you invoke it, a menu appears on the screen that lists the number and title of each example. Type the number of the example you wish to run, then press *<ENTER>*.

MENU.BAS Program Listing

```
100 CLS
120 PRINT "Examples Menu"
130 PRINT
140 PRINT "   1) Displaying a Single Waveform"
150 PRINT "   2) Managing Multiple Waveforms"
160 PRINT "   3) Defining Complex Waveforms"
170 PRINT "   4) Using Signal Processing & Transferring a Waveform"
180 PRINT "   5) Taking Automated Measurements"
190 PRINT "   6) Taking Delay Measurements"
195 PRINT "   7) Taking Delta-Delay Measurements From a Reference Trace"
200 PRINT "   8) Comparing Displayed Traces to Stored Waveforms"
207 PRINT "   9) Taking TDR Measurements"
210 PRINT
220 PRINT "Enter the number of the example you wish to run,"
230 PRINT "press 'Q' to quit without exiting BASIC,"
240 PRINT "or press 'S' to quit and exit BASIC."
250 INPUT PROGNUM$
260 IF LEFT$(PROGNUM$,1)="Q" OR LEFT$(PROGNUM$,1)="q" THEN END
270 IF LEFT$(PROGNUM$,1)="S" OR LEFT$(PROGNUM$,1)="s" THEN SYSTEM
280 PROGNUM=VAL(PROGNUM$)
290 IF PROGNUM<1 OR PROGNUM>9 THEN GOTO 220
300 ON PROGNUM GOTO 310,320,330,340,350,360,370,380,390
310 LOAD "EX1.BAS",R
320 LOAD "EX2.BAS",R
330 LOAD "EX3.BAS",R
340 LOAD "EX4.BAS",R
350 LOAD "EX5.BAS",R
360 LOAD "EX6.BAS",R
370 LOAD "EX7.BAS",R
380 LOAD "EX8.BAS",R
390 LOAD "EX9.BAS",R
```

Example 1: Displaying a Trace

This example shows how you can quickly display a meaningful trace. You will also become familiar with basic oscilloscope control.

For this example you will need a CSA 803C or 11801C with at least one sampling head installed and one SMA connecting cable.

CSA 803C

11801C
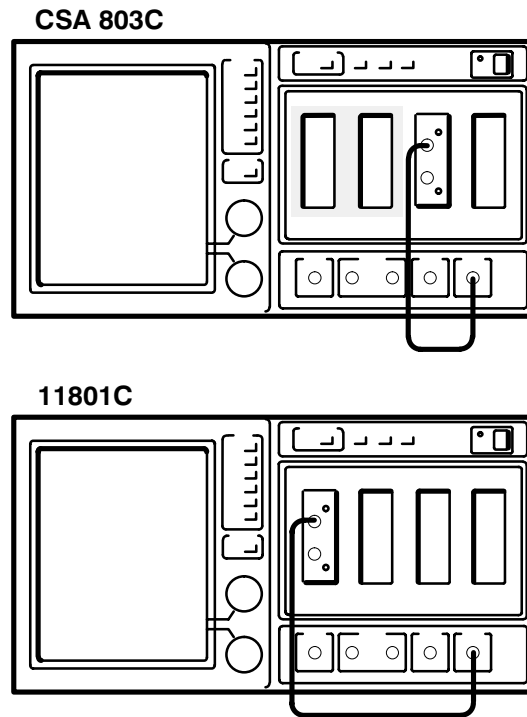
Figure 4-1: Connections for Example 1

To start the program, select Example 1 from the Examples Menu. Refer to the earlier discussion *Running the Example Software* for information on using the Examples Menu.

Program Listing

```
100 LOCATE ,,1,0,7

102 PRINT "Example 1: Displaying a Trace"
103 PRINT
110 PRINT "Ensure that the cable is connected and the scope's parameters are"
120 PRINT "set accordingly : (major menu UTILITY, minor menu RS232 Parameters)"
130 PRINT "  Baud 9600,  Echo OFF,  Stop Bits 1,  Parity NONE,  Flagging HARD"
140 PRINT "  Delay 0,    EOL String CR/LF,       Verbose OFF"
150 INPUT "  Press Enter when ready",A$

160 A$=""
170 WHILE A$<>"COM1" AND A$<>"COM2" AND A$<>"com2"
180    INPUT "Press Enter for using COM1 (default), else type COM2. ",A$
190    IF A$="" THEN A$="COM1"
200 WEND

210 PRINT "We are opening ";A$;" at 9600 baud, no parity, and 1 stop bit."
220 OPEN A$+":9600,N,8,1" AS #1

230 INPUT "Press Enter to ask for the scope's ID",A$
240 PRINT #1,"ID?"
250 LINE INPUT #1,RESPONSE$
260 PRINT RESPONSE$

270 INPUT "Press Enter after connecting the CALIBRATOR to channel 1 (head 1)",A$

280 INPUT "Press Enter to initialize scope.",A$
290 PRINT #1,"INIT"

300 A$=""
310 WHILE A$<>"y" AND A$<>"Y" AND A$<>"n" AND A$<>"N"
320 INPUT "Do you want to watch the commands in DEBUG mode ? (y/n)",A$
330 WEND

340 IF A$= "y" OR A$= "Y" THEN PRINT #1,"DEBUG RS232:ON" ELSE PRINT #1,
    "DEBUG RS232:OFF"

350 INPUT "Press Enter to setup trace1 from sampling head 1, channel 1.",A$
360 PRINT #1,"TRACE1 DESCRIPTION:'M1'"

370 INPUT "Press Enter to set the trigger to INTERNAL.",A$
380 PRINT #1,"TRIGGER SOURCE:INTERNAL"

390 INPUT "Press Enter to manually set the size and position of the trace.",A$
400 PRINT #1,"TBMAIN TIME:10E-9;MAINPOS 55E-9;CHM1 SENSI:.1,OFFSET:0"

410 INPUT "Press Enter to send 'AUTOSET' to the scope",A$
420 PRINT #1,"AUTOSET START"

430 PRINT "Example Completed."
435 CLOSE 1
440 LOAD "MENU.BAS",R
```

Program Description

Line 100 defines your computer cursor type and locations. In this case, the first two commas mean that the *x* and *y* coordinate arguments are not specified, 1 means the cursor is turned on, 0 places the cursor at the top of the cell to start, and 7 places the cursor at the bottom of the cell to finish.

Lines 102 through 150 identify the example and remind you to set up the CSA 803C/11801C RS-232-C communication parameters appropriately.

Lines 160 through 200 define your communication path to be an RS-232-C communication port (COM1 or COM2).

Lines 210 and 220 alert you that your computer is being set up to use the RS-232-C communication parameters: 9600 baud, no parity, 8-bit word length, and one stop bit.

NOTE

If this program crashes here, you should check that your RS-232-C cable is wired correctly, or that you have selected the correct RS-232-C port on your computer (preferably COM1).

Line 230 prompts you to initiate a identification query and line 240 actually sends the `ID?` command to the instrument.

Lines 250 and 260 read the `ID?` response message and print it on your computer screen.

Line 270 reminds you to make the necessary hardware setup for this example.

Line 280 prompts you to initialize the instrument and line 290 actually sends the `INIt` command to the instrument.

Line 300 initializes your forthcoming debug response to null.

Lines 310 through 330 waits for you to type Y or N to enable or disable `DEBug` mode. Turning `DEBug` on enables you to view the incoming commands in the top two lines of the front-panel display.

Line 340 sends the command to turn `DEBug` on, if you respond with an upper or lower case **Y** character.

Line 350 waits for you to press 'Enter' to initiate the definition of Trace1 and line 360 sends the command to define Trace1 to be from Channel 1 of the installed sampling head.

Line 370 prompts you to select internal triggering for this example and line 380 sends the command to set the trigger source to internal.

Line 390 prompts you to set the exact size and position of Trace1 using time base and channel commands.

Line 400 sends the commands to set the horizontal time per division of the main time base, the position of the main trace, and the vertical sensitivity and position of Channel 1.

Line 410 waits for you to press 'Enter' to initiate `AUTOSet` of the selected trace. This automatically sets the size and position parameters. Note that if triggering were set to external source, `AUTOSet` would also adjust the triggering parameters to produce a stable trace display.

Line 420 sends the command to set the vertical and horizontal parameters automatically.

Lines 430 and 435 prompt you that this example is finished and formally close the program. Line 440 returns you to the Examples Menu.

Examples

Example 2: Managing Multiple Traces

This example demonstrates multiple traces and graticules on the display. You will also become familiar with trace selection and management.

For this example you will need a CSA 803C or 11801C with at least one sampling head installed, three SMA connecting cables, a signal generator that provides a separate trigger output, a power divider, and adapters to fit the SMA cables to the signal generator. A signal-splitting T adaptor may be substituted for the power divider.
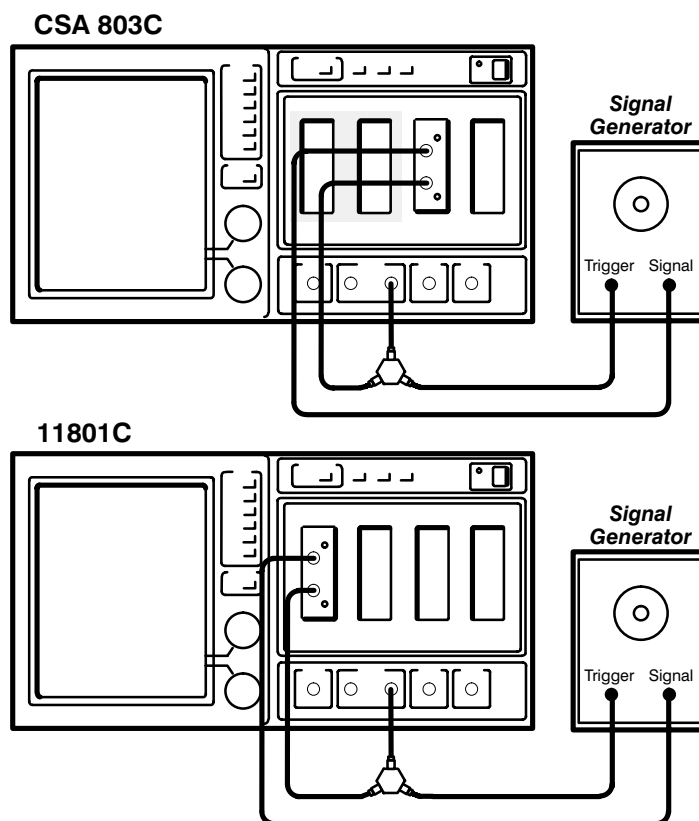
Figure 4-2: Connections for Example 2

To start the program, select Example 2 from the Examples Menu.

Program Listing

```
100 LOCATE ,,1,0,7
102 PRINT "Example 2: Managing Multiple Traces"
103 PRINT
110 PRINT "Ensure that the cable is connected and the scope's parameters are"
120 PRINT "set accordingly : (major menu UTILITY, minor menu RS232 Parameters)"
130 PRINT "  Baud 9600,  Echo OFF,  Stop Bits 1,  Parity NONE,  Flagging HARD"
140 PRINT "  Delay 0,    EOL String CR/LF,       Verbose OFF"
150 INPUT "  Press Enter when ready",A$
160 A$=""
170 WHILE A$<>"COM1" AND A$<>"COM2" AND A$<>"com2"
180    INPUT "Press Enter for using COM1 (default), else type COM2. ",A$
190    IF A$="" THEN A$="COM1"
200 WEND
210 PRINT "We are opening ";A$;" at 9600 baud, no parity, and 1 stop bit."
220 OPEN A$+":9600,N,8,1" AS #1
230 INPUT "Press Enter to ask for the scope's ID",A$
240 PRINT #1,"ID?"
250 LINE INPUT #1,RESPONSE$
260 PRINT RESPONSE$

270 PRINT "Press Enter after making the following connections :"
280 PRINT " Connect a 500kHz to 5MHz signal (.5V to 1V PP) to channel 1 (head 1)."
290 PRINT " Connect trigger out of your signal source to the TRIGGER INPUT of
    the scope."
300 PRINT " Use a power splitter or a power 'T' connecter to connect your trigger
out"
310 INPUT "  to channel 2 (head 1)",A$

320 INPUT "Press Enter to initialize scope.",A$
330 PRINT #1,"INIT"

340 A$=""
350 WHILE A$<>"y" AND A$<>"Y" AND A$<>"n" AND A$<>"N"
360    INPUT "Do you want to watch the commands in DEBUG mode ? (y/n)",A$
370 WEND
380 IF A$= "y" OR A$= "Y" THEN PRINT #1,"DEBUG RS232:ON" ELSE PRINT #1,
    "DEBUG RS232:OFF"

390 INPUT "Press Enter to setup trace1 from sampling head 1, channel 1.",A$
400 PRINT #1,"TRACE1 DESCRIPTION:'M1'"

410 INPUT "Press Enter to set the trigger to EXTERNAL",A$
420 PRINT #1,"TRIGGER SOURCE:EXTERNAL"
```

(continued)

Program Description

Lines 100 through 260 of Example 2 describe the same type of preparatory procedure as described in Example 1, program lines 100 through 260.

Lines 270 through 310 of this example remind you to make the necessary hardware setup for this example.

Lines 320 prompts you to initialize the instrument and line 330 sends the `INIt` command.

Lines 340 through 370 wait for you to type Y or N to enable or disable `DEBug` mode.

Line 380 displays the selected `DEBug` mode.

Line 390 waits for you to press 'Enter' to define Trace1 and line 400 sends the command to define Trace1 as Channel 1 of the sampling head.

Line 410 prompts you to select external triggering for this example and line 420 sends the command to set the triggering source to external.

(continued)

Program Listing (continued)

```
430 INPUT "Press Enter to manually set the size and position of the trace.",A$
440 PRINT #1,"TBMAIN TIME:2E-6;MAINPOS 55E-9;CHM1 SENSI:.2,OFFSET:-1"

450 INPUT "Press Enter to send 'AUTOSET' to the trace in period mode.",A$
460 PRINT #1,"AUTOSET MODE:PERIOD,START"

470 INPUT "Press Enter to send 'AUTOSET' in the edge mode",A$
480 PRINT #1,"AUTOSET MODE:EDGE,START"

490 INPUT "Press Enter to setup trace2 from sampling head 1, channel 2.",A$
500 PRINT #1,"TRACE2 DESCRIPTION:'M2';AUTOSET HORIZ:OFF,START,HORIZ:ON"

510 INPUT "Press Enter to select trace1 as the highlighted trace.",A$
520 PRINT #1,"SELECT TRACE1"

530 PRINT "Now we will ask for the trace count and their numbers."
540 INPUT "Also, we will ask for their descriptions. (Press Enter)",A$
550 PRINT #1,"TRANUM?;TRALIST?"
560 LINE INPUT #1,RESPONSE$
570 PRINT RESPONSE$
580 PRINT #1,"TRA1? DESCR;TRA2? DESCR"
590 LINE INPUT #1,RESPONSE$
600 PRINT RESPONSE$

610 INPUT "Press Enter to create the second graticule",A$
620 PRINT #1,"DISP GRAT:DUAL"

630 INPUT "Press Enter to move trace1 to the upper graticule",A$
640 PRINT #1,"ADJTRACE1 GRLOC:UPPER"

650 INPUT "Press Enter to reduce back to the single graticule",A$
660 PRINT #1,"DISP GRAT:SINGLE"

670 INPUT "Press Enter to remove TRACE1 and TRACE2.",A$
680 PRINT #1,"REMOVE TRACE1,TRACE2"

690 PRINT "Example Completed."
695 CLOSE 1
700 LOAD "MENU.BAS",R
```

Program Description (continued)

Line 430 prompts you to set the exact size and position of Trace1 (set values are in line 440) using timebase and channel commands.

Line 440 sends the commands to set the horizontal time per division of the main time base, the position of the main trace, and the vertical sensitivity and position of Channel 1. You can compare the results of this setting with the `AUTOSet` results from lines 450 and 460.

Line 450 prompts you to `AUTOSet` the selected trace in period mode. This is the common mode to automatically size, position, and trigger a trace when you're interested in its periodic characteristics. Note that since external trigger sourcing is selected, `AUTOSet` includes adjusting trigger parameters.

Line 460 sends the command to automatically set the trace size, position, and trigger parameters using `PERiod` mode of the `AUTOSet` function.

Line 470 prompts you to `AUTOSet` the selected trace in `EDGe` mode. This is the common mode to automatically size, position, and trigger a trace when you're interested in the characteristics of the rising or falling edge.

Line 480 sends the command to automatically set the size, position, and trigger parameters using `EDGe` mode of the `AUTOSet` function.

Line 490 waits for you to press 'Enter' to define Trace2. Line 500 sends the command to define Trace2 to be from Channel 2 of the sampling head. It also sends the commands necessary to autoset the selected trace, (Trace2), and to keep Trace1 and Trace2 at the same horizontal size and position settings.

Line 510 prompts you to select Trace1, thus highlighting it and line 520 sends the command to select Trace1.

Lines 530 and 600 identify the number of defined traces, their assigned trace numbers (this may include previously stored traces) and their trace source descriptions.

Line 610 prompts you to create a second graticule and line 620 sends the command to create it.

Line 630 prompts you to move Trace1 to the upper graticule and line 640 sends the command to do this.

Line 650 prompts you to go back to a single graticule and line 660 sends the command to do this.

Line 670 prompts you to remove Trace1 and Trace2 from the display and line 680 sends the command to remove both traces from the display.

Lines 690 and 695 prompt you that this example is finished and formally close the program. Line 700 returns you to the Examples Menu.

Examples

Example 3: Defining Complex Traces

This example shows how you can create traces that combine signals from more than one channel.

For this example you will need a CSA 803C or 11801C with at least one dual-channel sampling head installed. Also, a power divider and two SMA cables of different lengths will be used (2 ns and 5 ns cables are recommended). A signal-splitting T adaptor may be substituted for the power divider.
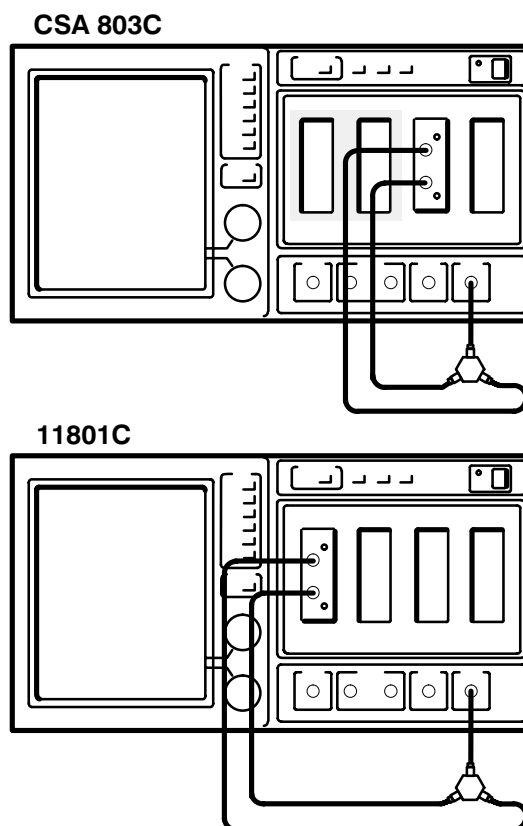

CSA 803C

11801C
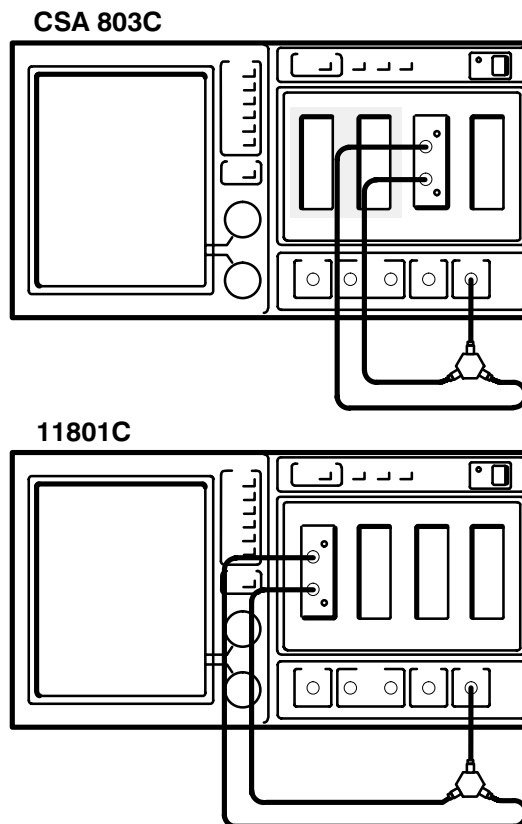
Figure 4-3: Connections for Example 3

To start the program, select Example 3 from the Examples Menu. Refer to the earlier discussion *Running the Example Software* for information on using the Examples Menu.

Program Listing

```
100 LOCATE ,,1,0,7
102 PRINT "Example 3: Defining Complex Traces"
103 PRINT
110 PRINT "Ensure that the cable is connected and the scope's parameters are"
120 PRINT "set accordingly : (major menu UTILITY, minor menu RS232 Parameters)"
130 PRINT "  Baud 9600,  Echo OFF,  Stop Bits 1,  Parity NONE,  Flagging HARD"
140 PRINT "  Delay 0,    EOL String CR/LF,        Verbose OFF"
150 INPUT "  Press Enter when ready",A$
160 A$=""
170 WHILE A$<>"COM1" AND A$<>"COM2" AND A$<>"com2"
180    INPUT "Press Enter for using COM1 (default), else type COM2. ",A$
190    IF A$="" THEN A$="COM1"
200 WEND
210 PRINT "We are opening ";A$;" at 9600 baud, no parity, and 1 stop bit."
220 OPEN A$+":9600,N,8,1" AS #1
230 INPUT "Press Enter to ask for the scope's ID",A$
240 PRINT #1,"ID?"
250 LINE INPUT #1,RESPONSE$
260 PRINT RESPONSE$

270 PRINT "Press Enter after making the following connections :"
280 PRINT "  Use a power splitter or a power 'T' connecter to connect a 2ns and
    a 5ns"
290 PRINT "  cable to channel 1 and 2 of sampling head 1."
300 INPUT "  Connect the 'T' or splitter to the CALIBRATOR",A$
310 INPUT "Press Enter to initialize scope.",A$
320 PRINT #1,"INIT"

330 A$=""
340 WHILE A$<>"y" AND A$<>"Y" AND A$<>"n" AND A$<>"N"
350    INPUT "Do you want to watch the commands in DEBUG mode ? (y/n)",A$
360 WEND
370 IF A$= "y" OR A$= "Y" THEN PRINT #1,"DEBUG RS232:ON" ELSE PRINT #1,
    "DEBUG RS232:OFF"

380 INPUT "Press Enter to setup 2 traces from channel 1 & 2.",A$
390 PRINT #1,"TRACE1 DESCRIPTION:'M1';TRACE2 DESC:'M2'"

400 INPUT "Press Enter to set the trigger to INTERNAL.",A$
410 PRINT #1,"TRIGGER SOURCE:INTERNAL"

420 INPUT "Press Enter to set the main timebase to 10ns/div.",A$
430 PRINT #1,"TBMAIN TIME:10E-9;CHM1 SENSITIVITY:100E-3;CHM2 SENSITIVITY:100E-3"
```

(continued)

Program Description

Lines 100 through 260 of Example 3 describe the same type of preparatory procedure as described in Example 1, program lines 100 through 260.

Lines 270 through 300 of this example remind you to make the necessary hardware setup for this example.

Line 310 prompts you to initialize the instrument and line 320 sends the `INIt` command.

Lines 330 through 360 wait for you to type Y or N to enable or disable `DEBug` mode.

Line 370 displays the selected `DEBug` mode.

Lines 380 through 390 wait for you to press 'Enter' to initiate the definition of Trace1 and Trace 2 from Channel 1 and 2 of the sampling head.

Line 400 prompts you to select internal triggering for this example and line 410 sends the command to set the triggering source.

Line 420 prompts you to set the main time base and line 430 sends the main time base command (10 ns/div setting) and the commands to set the sensitivity (i.e., vertical size) of Channel 1 and Channel 2 to 100 mV (this is to give Trace1 and Trace2 common scaling).

(continued)

Program Listing (continued)

```
440 INPUT "Press Enter to create trace3 as the difference of channel 1 & 2",A$
450 PRINT #1,"TRACE3 DESCRIPTION:'M1-M2'"

460 INPUT "Press Enter to remove traces 1 and 2.",A$
470 PRINT #1,"REMOVE TRACE1,TRACE2"

480 INPUT "Press Enter to make a window on trace 3.",A$
490 PRINT #1,"TRACE4 DESCRIPTION:'M1-M2 ON WIN'"

500 INPUT "Press Enter to change the window time base and position.",A$
510 PRINT #1,"TBWIN TIME:1E-9;WIN4 POS:73E-9"

520 INPUT "Press Enter to make a second window on trace 3.",A$
530 PRINT #1,"TRACE5 DESCRIPTION:'M1-M2 ON WIN'"
540 INPUT "Press Enter to change the window position on trace 5.",A$
550 PRINT #1,"WIN5 POS:86E-9"
560 INPUT "Press Enter to add trace separation to Trace 4 and 5.",A$
570 PRINT #1,"ADJTRACE4 TRSEP:-1;ADJTRACE5 TRSEP:1"

580 PRINT "Example Completed."
585 CLOSE 1
590 LOAD "MENU.BAS",R
```

Program Description (continued)

Line 440 prompts you to create a third trace, the difference between Trace1 and Trace2 and line 450 sends the command to create Trace3 by subtracting the signal from Channel 2 from the signal from Channel 1.

Line 460 prompts you to remove Trace1 and Trace2 from the display and line 470 sends the command to remove these traces, leaving only Trace3.

Line 480 prompts you to create a window trace on Trace3 and line 490 sends the command to create window Trace4 by using the window time base and subtracting the signal of Channel 2 from the signal of Channel 1.

Line 500 prompts you to change the horizontal size and position of the window time base and line 510 sends the command to set the time per division of the window time base to 1 ns per division and the horizontal position of window Trace4 to 73 ns.

Lines 520 through 570 prompt you to create a second window trace and position it by adjusting the trace separation of Trace4 and Trace5.

Lines 580 and 585 prompt you that this example is finished and formally close the program. Line 590 returns you to the Examples Menu.

Examples

Example 4: Using Signal Processing & Transferring a Waveform

This example shows how the CSA 803C and 11801C can process your signals to extract more information and how to transfer traces to your computer. Trace averaging is started on a trace and then used to halt conditional acquisition. Three other display types are invoked: *variable persistence*, *infinite persistence*, and *color graded*. Also, traces are transferred from the CSA 803C or 11801C to your computer.

You will simulate a noisy signal by adding the calibrator to a very small (50 mV$_{p-p}$), high frequency (50 MHz) sine wave that is not synchronized with the trigger. If you have a broad spectrum noise source, you should substitute it for the signal generator in this example.

For this example you will need a CSA 803C or 11801C with at least one dual-channel sampling head installed, a signal generator, and two SMA cables.
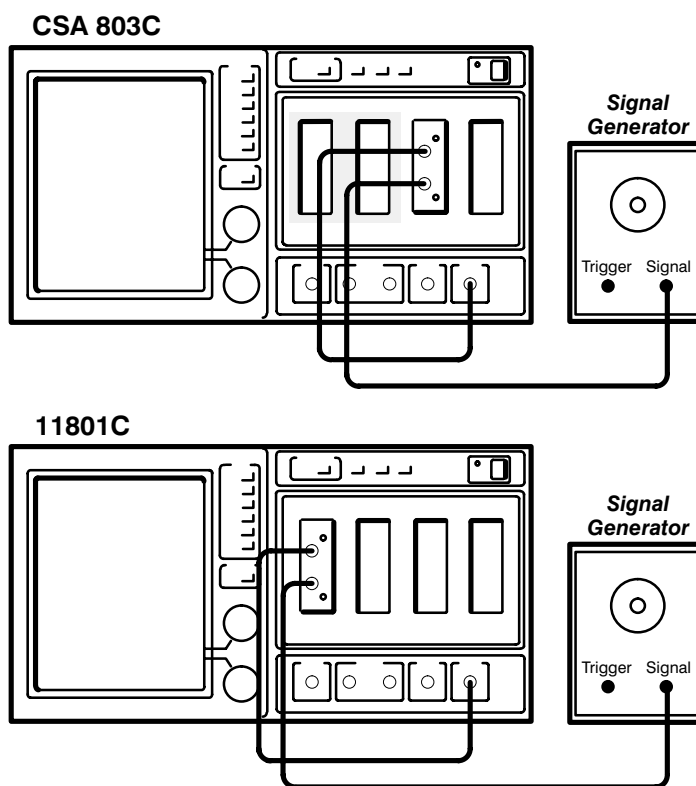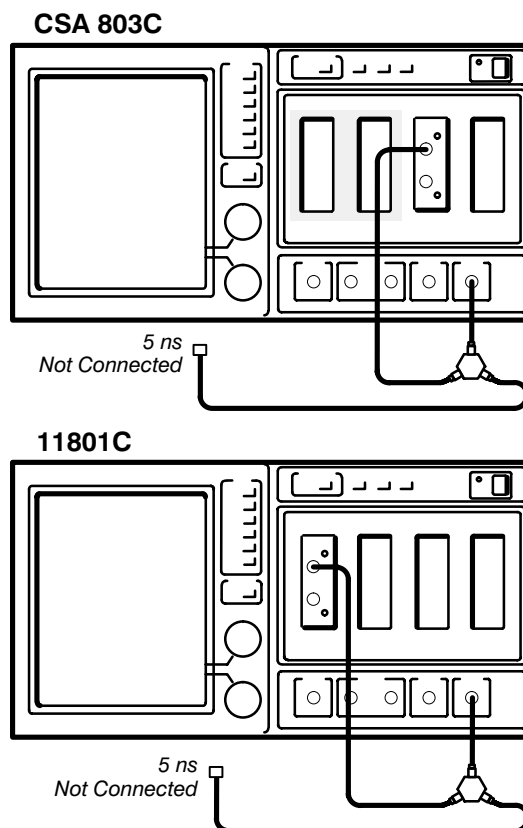

Figure 4-4: Connections for Example 4

To start the program, select Example 4 from the Examples Menu. Refer to the earlier discussion *Running the Example Software* for information on using the Examples Menu.

Program Listing

```
100 LOCATE ,,1,0,7
102 PRINT "Example 4: Using Signal Processing & Transferring a Waveform"
103 PRINT
110 PRINT "* * * You must have invoked BASIC with a COM buffer of 5000 bytes"
111 PRINT "* * * to run this example."
120 PRINT "* * * Sample invocation: BASIC /C:5000 or BASICA /C:5000 or QB /C:5000"
122 PRINT "* * * If you have not invoked BASIC in this way, this example WILL
    NOT RUN;"
123 PRINT "* * * get out of BASIC and re-invoke."
124 PRINT
130 PRINT "Ensure that the cable is connected and the scope's parameters are"
140 PRINT "set accordingly : (major menu UTILITY, minor menu RS232 Parameters)"
150 PRINT "  Baud 9600,  Echo OFF,  Stop Bits 1,  Parity NONE,  Flagging HARD"
160 PRINT "  Delay 0,    EOL String CR/LF,       Verbose OFF"
170 INPUT "  Press Enter when ready",A$
180 A$=""
190 WHILE A$<>"COM1" AND A$<>"COM2" AND A$<>"com2"
200    INPUT "Press Enter for using COM1 (default), else type COM2. ",A$
210    IF A$="" THEN A$="COM1"
220 WEND
230 PRINT "We are opening ";A$;" at 9600 baud, no parity, and 1 stop bit."
240 OPEN A$+":9600,N,8,1" AS #1
250 INPUT "Press Enter to ask for the scope's ID",A$
260 PRINT #1,"ID?"
270 LINE INPUT #1,RESPONSE$
280 PRINT RESPONSE$

290 PRINT "Press Enter after making the following connections :"
300 PRINT " Connect a 50 MHz sine wave, 50 mV p-p, to channel 2."
310 PRINT " Connect the Calibrator signal to channel 1."
320 INPUT "Press Enter to initialize scope.",A$
330 PRINT #1,"INIT"

340 A$=""
350 WHILE A$<>"y" AND A$<>"Y" AND A$<>"n" AND A$<>"N"
360    INPUT "Do you want to watch the commands in DEBUG mode ? (y/n)",A$
370 WEND
380 IF A$= "y" OR A$= "Y" THEN PRINT #1,"DEBUG RS232:ON" ELSE PRINT #1,
    "DEBUG RS232:OFF"

390 INPUT "Press Enter to set the trigger to INTERNAL.",A$
400 PRINT #1,"TRIGGER SOURCE:INTERNAL"
```

(continued)

Program Description

Lines 100 through 280 of Example 4 describe the same type of preparatory procedure as described in Example 1, program lines 100 through 260. One exception is a warning that you must have started Basic with an argument that would set up a 5000 byte buffer used in the waveform transfer.

Lines 290 through 310 of this example remind you to make the necessary hardware setup for this example.

Line 320 prompts you to initialize the instrument and line 330 sends the `INIt` command.

Lines 340 through 370 wait for you to type Y or N to enable or disable `DEBug` mode.

Line 380 displays the selected `DEBug` mode.

Lines 380 through 390 wait for you to press 'Enter' to initiate the definition of Trace1 Trace 2 from Channel 1 and 2 of the sampling head.

Line 390 prompts you to select internal triggering for this example and line 400 sends the command to set the triggering source.

(continued)

Program Listing (continued)

```
410 INPUT "Press Enter to define an averaged and a normal summation-trace with
    M1 and M2",A$
420 PRINT #1,"TRACE1 DESCR:'M1+M2';TRACE2 DESCR:'AVG(M1+M2)'"
430 INPUT "Press Enter to set the time base and vertical size of trace 2",A$
440 PRINT #1,"TBMAIN TIME:10E-9;CHM1 SENSI:.2;CHM2 SENSI:.2"

450 INPUT "Press Enter to set the number of averages to 12.",A$
460 PRINT #1,"NAVG 12"
470 INPUT "Press Enter to begin a conditional acquisition.",A$
480 PRINT "The average countdown will be complete when 'n' = zero."
490 PRINT #1,"CONDACQ TYPE:AVG"
500 REMAINING=999
510 WHILE REMAINING>0
520    PRINT #1,"CONDACQ? REMAINING"
530    LINE INPUT #1,RESPONSE$
540    REMAINING=VAL(MID$(RESPONSE$,INSTR(RESPONSE$,":")+1))
550    PRINT REMAINING;
560 WEND
570 PRINT

580 INPUT "Press Enter to send the 512 point curve to the PC.",A$
590 GOSUB 805

600 INPUT "Press Enter to turn the average function off for TRACE2.",A$
610 PRINT #1,"AVG OFF;CONDACQ TYPE:CONTINUOUS"

620 INPUT "Press Enter to set the display for infinite persistence.",A$
630 PRINT #1,"DISPLAY TYPE:INFINITE"
640 INPUT "Press Enter to set the display to variable persistence.",A$
650 PRINT #1,"DISPLAY TYPE:VARIABLE"
652 INPUT "Press Enter to set the display to color graded.",A$
654 PRINT #1,"DISPLAY TYPE:GRADED"
656 INPUT "Press Enter to set the display back to normal.",A$
658 PRINT #1,"DISPLAY TYPE:NORMAL"

660 PRINT "Press Enter to smooth channel 1."
670 INPUT " (note: Turning on smoothing for CHM1 also smooths CHM2)",A$
680 PRINT #1,"CHM1 SMOOTHING:ON;CHM2? SMOOTHING"
690 LINE INPUT #1,RESPONSE$
700 PRINT RESPONSE$

710 INPUT "Press Enter to turn off the smoothing.",A$
720 PRINT #1,"CHM1 SMOOTHING:OFF;CHM2? SMOOTHING"
730 LINE INPUT #1,RESPONSE$
740 PRINT RESPONSE$
```

(continued)

Program Description (continued)

Line 410 waits for you to press 'Enter' to define two traces and line 420 sends the command that defines Trace1 to be the summation of Channels 1 and 2, and it sends the command that defines Trace2 to be the average of the summation of Channels 1 and 2.

Line 430 prompts you to set the horizontal and vertical size of Trace1 and Trace2. Line 440 sends the commands that set the main time base to 10 ns per division and the sensitivities of Channel 1 and 2 to 200 mV per division. Note that both channels must be set since they are summed.

Lines 450 through 570 prompts you to set 12 averages (that is, average 12 records) for a conditional trace acquisition and to initiate conditional acquisition which stops acquisition after 12 acquisitions.

Line 580 prompts you to send the acquired trace/waveform to your computer. Line 590 sends the program to a subroutine (lines 805 through 1020) that transfers the data points of your waveform to your computer. And, if you have graphics, it draws the waveform on your computer screen. Then it returns you to line 600 to continue the example.

Line 600 prompts you to turn off the average function for Trace2, and line 610 sends the command to turn off averaging and reset conditional acquire to the continuous mode.

Lines 620 and 630 prompt you to send the command that sets the display type to Infinite Persistence.

Lines 640 and 650 prompt you to send the command that sets the display type to Variable Persistence.

Lines 652 and 654 prompt you to send the command that sets the display type to Color Graded.

Lines 656 and 658 prompt you to send the command that sets the display type back to Normal.

Lines 660 and 670 prompt you to select trace smoothing for Channel 1 which also starts smoothing for Channel 2.

Line 680 sends the command to turn smoothing on for Channel 1 and sends a command to query Channel 2 to check that smoothing was turned on. Lines 690 and 700 read the query response and print it on the computer screen.

Lines 710 and 740 prompt you to turn smoothing off, and then send commands to turn smoothing off for Channel 1, and to query if Channel 2 smoothing was turned off.

(continued)

Program Listing (continued)

```
750 INPUT "Press Enter to set the main record to 2048 and the window to 5120",A$
760 PRINT #1,"TBMAIN LENGTH:2048;TBWIN LENGTH:5120"
770 INPUT "Press Enter to send the 2048 point curve to the PC.",A$
780 GOSUB 810

790 PRINT "Example Completed."
795 CLOSE 1
800 LOAD "MENU.BAS",R

805 REM The following line sets the output waveform to be TRACE2; the waveform
806 REM will be sent in 16 bit binary format and each word will be sent as low
807 REM order byte followed by high order byte; lastly, ask for the data.
810 PRINT #1,"OUTPUT TRACE2;ENCDG WAV:BIN;BYT.OR LSB;CURVE?
815 REM read one byte off the bus
820 A$=INPUT$(1,#1)
825 REM loop until we see the percent sign
830 WHILE A$<>"%" :   A$=INPUT$(1,#1) :WEND
835 REM next, read in the data length count (always in MSB,LSB word format)
840 HB$=INPUT$(1,#1) : LB$=INPUT$(1,#1)
845 REM compute the byte count
850 BYTE.CNT%=ASC(HB$)*256+ASC(LB$)
855 REM convert the byte count into waveform point count
860 NR.PT%=(BYTE.CNT%-1)/2 : DIM WFM%(NR.PT%)
865 REM read in the waveform in one word at a time
870 FOR I%=1 TO NR.PT% :   WFM%(I%)=CVI(INPUT$(2,#1)) : PRINT ".";: NEXT
875 REM lastly, read in the checksum (non-zero only for stored waveforms)
880 LINE INPUT #1,CHKSUM$
890 PRINT
900 INPUT "Waveform copy complete. Press Enter to graph.",A$
905 REM check for graphics adapter
910 DEF SEG = (&H40)
920 CRTTYPE = PEEK(&H49)
930 DEF SEG = 0
940 IF CRTTYPE =7 THEN PRINT "Graphics not available." : ERASE WFM% : RETURN
945 REM switch to 640x200 (try SCREEN 9 for EGA if you use QuickBASIC)
950 SCREEN 2
960 REM set the screen to be zero +/- 32k by record-length (e.g. 1024)
970 WINDOW(1,-32767)-(nr.pt%,32767)
980 REM plot each point as a pixel
990 FOR i%=1 TO nr.pt% : PSET(i%,wfm%(i%)) : NEXT
1000 INPUT "Press Enter to clear the screen.",A$
1005 REM remove the waveform array and set the screen back to text
1010 ERASE WFM% : SCREEN 0
1020 RETURN
```

Program Description (continued)

Line 750 prompts you to set the main record length to 2048 data points and the window record length to 5120 data points. Line 760 sends the commands to do this.

Line 770 prompts you to send the new 2048 point curve (that is, trace or waveform) to your computer.

Line 780 sends the program to the curve transfer and screen draw subroutine (lines 805 though 1020) and returns you to line 790 when it is finished.

Lines 790 and 795 prompt you that this example is finished and formally closes the program. Line 800 returns you to the Examples Menu.

Lines 805 through 1020 compose a subroutine that transfers the data points of a trace record to your computer. And, if you have graphics capability, it draws the waveform on your computer screen. It then returns to the line following the calling GOSUB command to continue the example.

Example 5: Taking Automated Measurements

This example demonstrates how quickly you can take a dynamic measurement from a displayed trace.

For this example you will need a CSA 803C or 11801C with a least one sampling head installed, a signal generator, and two SMA connecting cables.
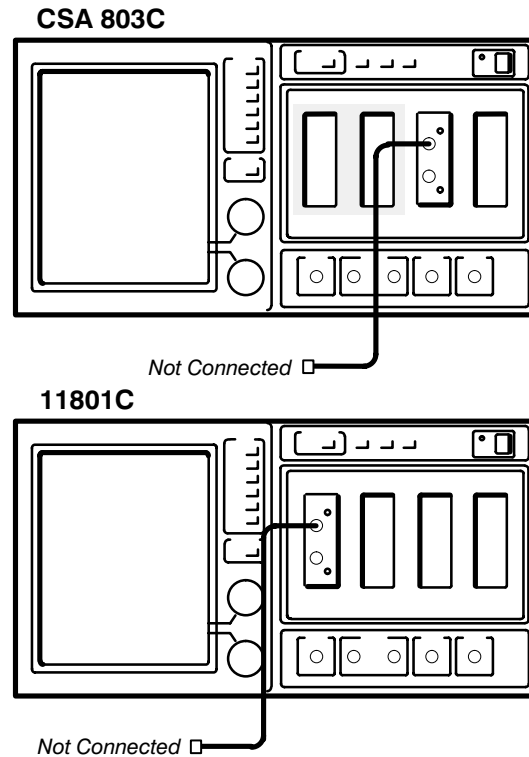
CSA 803C

11801C
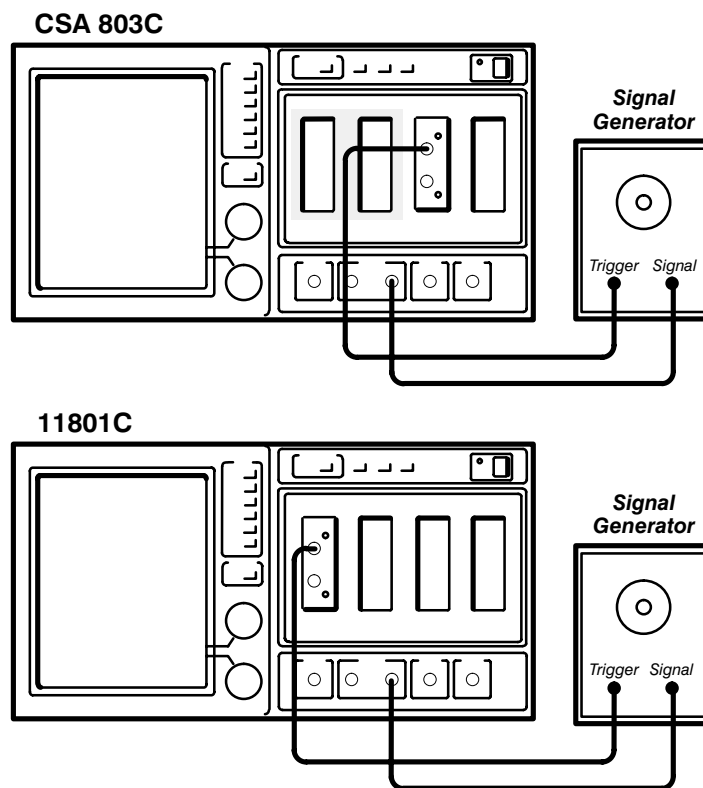
Figure 4-5: Connections for Example 5

To start the program, select Example 5 from the Examples Menu. Refer to the earlier discussion *Running the Example Software* for information on using the Examples Menu.

Program Listing

```
100 LOCATE ,,1,0,7
102 PRINT "Example 5: Taking Automated Measurements"
103 PRINT
110 PRINT "Ensure that the cable is connected and the scope's parameters are"
120 PRINT "set accordingly : (major menu UTILITY, minor menu RS232 Parameters)"
130 PRINT "  Baud 9600,  Echo OFF,  Stop Bits 1,  Parity NONE,  Flagging HARD"
140 PRINT "  Delay 0,    EOL String CR/LF,        Verbose OFF"
150 INPUT "  Press Enter when ready",A$
160 A$=""
170 WHILE A$<>"COM1" AND A$<>"COM2" AND A$<>"com2"
180    INPUT "Press Enter for using COM1 (default), else type COM2. ",A$
190    IF A$="" THEN A$="COM1"
200 WEND
210 PRINT "We are opening ";A$;" at 9600 baud, no parity, and 1 stop bit."
220 OPEN A$+":9600,N,8,1" AS #1
230 INPUT "Press Enter to ask for the scope's ID",A$
240 PRINT #1,"ID?"
250 LINE INPUT #1,RESPONSE$
260 PRINT RESPONSE$

270 PRINT "Press Enter after making the following connections :"
280 PRINT " Connect a 1 MHz sine wave, .5 to 1V P-P to channel 1."
290 PRINT " Connect the generator trigger out to the scope TRIGGER INPUT."
300 INPUT "Press Enter to initialize scope.",A$
310 PRINT #1,"INIT"

320 A$=""
330 WHILE A$<>"y" AND A$<>"Y" AND A$<>"n" AND A$<>"N"
340    INPUT "Do you want to watch the commands in DEBUG mode ? (y/n)",A$
350 WEND
360 IF A$= "y" OR A$= "Y" THEN PRINT #1,"DEBUG RS232:ON" ELSE PRINT #1,
    "DEBUG RS232:OFF"

370 INPUT "Press Enter to setup trace1 from sampling head 1, channel 1.",A$
380 PRINT #1,"TRACE1 DESCRIPTION:'M1'"

390 INPUT "Press Enter to set the trigger to EXTERNAL.",A$
400 PRINT #1,"TRIGGER SOURCE:EXTERNAL"

410 INPUT "Press Enter to send 'AUTOSET' to the scope",A$
420 PRINT #1,"AUTOSET MODE:PERIOD,START"
```

(continued)

Program Description

Lines 100 through 260 of Example 5 describe the same type of preparatory procedure as described in Example 1, program lines 100 through 260.

Lines 270 through 310 of this example remind you to make the necessary hardware setup and to initialize the instrument for this example.

Lines 320 through 360 wait for you to type Y or N to enable or disable `DEBug` mode then display the selected `DEBug` mode.

Line 370 waits for you to press 'Enter' to define Trace1 and line 380 sends the command to define Trace1 as Channel 1 of the sampling head.

Line 390 prompts you to select external triggering for this example and line 400 sends the command to do this.

Line 410 prompts you to `AUTOSet` the selected trace and line 420 sends the command to `AUTOSet` in period mode.

(continued)

Program Listing (continued)

```
430 INPUT "Press Enter to calculate RMS and FREQUENCY on the SELECTed TRACE.",A$
440 PRINT #1,"RMS?;FREQ?"
450 LINE INPUT #1,RESPONSE$
460 PRINT "(a histogram was calculated for each measurement query)"
470 PRINT RESPONSE$

480 INPUT "Press Enter to put RMS and FREQUENCY into the measure list for
    TRACE1.",A$
490 PRINT #1,"MSLIST1 RMS,FREQ"
500 PRINT "Press Enter to get the query results"
510 INPUT " (only one histogram is made for all measurements)",A$
520 PRINT #1,"MEAS1?"
530 LINE INPUT #1,RESPONSE$
540 PRINT RESPONSE$

550 INPUT "Press Enter to setup the statistical data.",A$
560 PRINT #1,"STATISTICS MEAS:RMS,N:24;MSYS ON"
570 INPUT "Press Enter to fetch the statistical data.",A$
580 PRINT #1,"STAT?"
590 LINE INPUT #1,RESPONSE$
600 PRINT RESPONSE$

610 INPUT "Press Enter to change the measurement zones on the trace.",A$
620 PRINT #1,"MPARAM1 LMZONE:15,RMZONE:85"

630 INPUT "Press Enter to measure CHM1 freq with s/w (using new zones).",A$
640 PRINT #1,"MPARAM1 MMODE:SW;MSLIST1 FREQ;MEAS1?"
650 LINE INPUT #1,RESPONSE$
660 PRINT RESPONSE$

670 INPUT "Press Enter to measure CHM1 freq with h/w (zones not used).",A$
680 PRINT #1,"MPARAM1 MMODE:HW;MEAS1?"
690 LINE INPUT #1,RESPONSE$
700 PRINT RESPONSE$

710 INPUT "Press Enter to remove TRACE 1",A$
720 PRINT #1,"REMOVE TRACE1"

770 PRINT "Example Completed."
775 CLOSE 1
780 LOAD "MENU.BAS",R
```

Program Description (continued)

Line 430 prompts you to measure the RMS and frequency of the selected trace (that is, Trace1).

Line 440 sends the commands to query RMS and frequency and return their values.

NOTE

Beginning with these independent RMS and frequency measurements, there are four more times that these measurements are taken. Each time, a different measurement method is used to demonstrate the various methods that are available to you with the CSA 803C or 11801C. The subsequent measurements are made in lines 520 (demonstrating measurement list measurements), 640 (demonstrating software system measurements), and 680 (demonstrating hardware system measurements).

Lines 450 through 470 read the query responses and print the RMS and frequency values on your computer screen.

Line 480 prompts you to add RMS and frequency to the measurement list (`MSLISt`) for Trace1 and line 490 sends the command to do this.

Line 500 through 540 query the new values for the RMS and frequency measurements just added to the measurement list for Trace1.

Lines 550 and 560 set up the instrument to take statistical data on the RMS measurement, using 24 samples, and to turn the measurement system on. Also, note that `MSYS` must be `ON` to use the statistics function.

NOTE

If you press 'Enter' too soon in line 570, line 560 will not have had time to take all 24 samples.

Line 570 prompts you to get the statistical data and lines 580 through 600 query the statistical data.

Line 610 prompts you to change the reference zones for taking measurements and line 620 sends the command changing the left and right measurement zones for Trace1. It sets them to 15 percent and 85 percent, respectively.

Lines 630 through 660 measure Trace1 frequency in software mode, using the new measurement zones.

Line 670 through 700 measure Trace1 frequency in hardware mode, without the measurement zones.

Line 710 prompts you to remove Trace1 from the display and line 720 sends the command to do this.

Lines 770 and 775 prompt you that this example is finished and formally close the program. Line 780 returns you to the Examples Menu.

Examples

Example 6: Taking Delay Measurements

This example shows another way to measure trace parameters, such as delay, using cursors. When a measurement you want to make is not included in the list of automated measurements, cursors can be used.

You can use the cursors to take two common measurements, trace amplitude and delay between traces. The CSA 803C and 11801C can do both of these as automated measurements, so you can compare the method of using automated measurements to using cursors.

For this example you will need a CSA 803C or 11801C with at least one dual-channel sampling head installed. Also, a power divider and two SMA cables of different length will be used (2 ns and 5 ns cables are recommended). A signal splitting T adaptor may be substituted for the power divider.

CSA 803C

11801C

Figure 4-6: Connections for Example 6

To start the program, select Example 6 from the Examples Menu. Refer to the earlier discussion *Running the Example Software* for information on using the Examples Menu.

Program Listing

```
100 LOCATE ,,1,0,7
102 PRINT "Example 6: Taking Delay Measurements"
103 PRINT
110 PRINT "Ensure that the cable is connected and the scope's parameters are"
120 PRINT "set accordingly : (major menu UTILITY, minor menu RS232 Parameters)"
130 PRINT "  Baud 9600,  Echo OFF,  Stop Bits 1,  Parity NONE,  Flagging HARD"
140 PRINT "  Delay 0,    EOL String CR/LF,        Verbose OFF"
150 INPUT "  Press Enter when ready",A$
160 A$=""
170 WHILE A$<>"COM1" AND A$<>"COM2" AND A$<>"com2"
180    INPUT "Press Enter for using COM1 (default), else type COM2. ",A$
190    IF A$="" THEN A$="COM1"
200 WEND
210 PRINT "We are opening ";A$;" at 9600 baud, no parity, and 1 stop bit."
220 OPEN A$+":9600,N,8,1" AS #1
230 INPUT "Press Enter to ask for the scope's ID",A$
240 PRINT #1,"ID?"
250 LINE INPUT #1,RESPONSE$
260 PRINT RESPONSE$

270 PRINT "Press Enter after making the following connections :"
280 PRINT " Use a power splitter or a power 'T' connecter to connect a 2ns and
    a 5ns"
290 PRINT " cable to channel 1 and 2 of sampling head 1."
300 INPUT " Connect the 'T' or splitter to the CALIBRATOR of the 11800",A$
310 INPUT "Press Enter to initialize scope.",A$
320 PRINT #1,"INIT"
330 A$=""

340 WHILE A$<>"y" AND A$<>"Y" AND A$<>"n" AND A$<>"N"
350    INPUT "Do you want to watch the commands in DEBUG mode ? (y/n)",A$
360 WEND
370 IF A$= "y" OR A$= "Y" THEN PRINT #1,"DEBUG RS232:ON" ELSE PRINT #1,
    "DEBUG RS232:OFF"

380 INPUT "Press Enter to setup trace1 from sampling head 1, channel 1.",A$
390 PRINT #1,"TRACE1 DESCRIPTION:'m1'"
400 INPUT "Press Enter to set the trigger to the INTERNAL.",A$
410 PRINT #1,"TRIGGER SOURCE:INTERNAL"

420 INPUT "Press Enter to set to the time base to 10ns/div",A$
430 PRINT #1,"TBMAIN TIME:10E-9"
```

(continued)

Program Description

Lines 100 through 260 of Example 6 describe the same type of preparatory procedure as described in Example 1, program lines 100 through 260.

Lines 270 through 330 of this example remind you to make the necessary hardware setup and to initialize the instrument for this example.

Lines 340 through 370 wait for you to type Y or N to enable or disable `DEBug` mode then display the selected `DEBug` mode.

Line 380 waits for you to press 'Enter' to define Trace1 and line 390 sends the command to define Trace1 as Channel 1 of the sampling head.

Line 400 prompts you to select internal triggering for this example and line 410 sends the command to do this.

Line 420 prompts you to set the time per division for the main time base and line 430 sends the command to do this.

(continued)

Program Listing (continued)

```
440 INPUT "Press Enter to turn on the cursor readout.",A$
450 PRINT #1,"CURSOR READOUT:ON"

460 INPUT "Press Enter to select horizontal bars.",A$
470 PRINT #1,"CURSOR TYPE:HBARS"

480 INPUT "Press Enter to move the bars near the middle of the CRT.",A$
490 PRINT #1,"H1BAR YDIV:1;H2BAR YDIV:-1"

500 INPUT "Press Enter to go to paired cursors on trace 1.",A$
510 PRINT #1,"CURSOR TYPE:PAIRED"
520 INPUT "Press Enter to position the cursors on divisions 1 and 9.",A$
530 PRINT #1,"DOT1ABS XDIV:-4;DOT2ABS XDIV:4"

540 PRINT #1,"DOT1ABS? YCOORD;DOT2ABS? YCOORD;PP?"
550 LINE INPUT #1,RESPONSE$
560 PRINT "Here are the vertical values from each cursor:"
570 PRINT RESPONSE$

580 M$=MID$(RESPONSE$,INSTR(RESPONSE$,":")+1)
590 PP=ABS(VAL(M$)-VAL(MID$(M$,INSTR(M$,":")+1)))
600 PRINT "Here is the Peak to Peak value from the cursors:";PP

610 M$=MID$(RESPONSE$,INSTR(RESPONSE$,"PP")+1)
620 PP=VAL(MID$(M$,INSTR(M$," ")+1))
630 PRINT "Here is the Peak to Peak value from the measurement system:";PP

640 INPUT "Press Enter to create a second trace on the lower graticule.",A$
650 PRINT #1,"TRACE2 DESC:'M2';DISPLAY GRAT:DUAL;ADJTRA1 GRLOC:UPPER;ADJTRA2
    GRLOC:LOWER"

660 PRINT #1,"SELECT TRACE1;CURSOR TYPE:SPLIT,REFERENCE:TRACE2"
670 PRINT "You may use the cursors to make a timing measurement, but we will"
680 PRINT " show you how to make accurate automatic measurements with the "
690 PRINT " measurement system."

700 PRINT "Example Completed."
705 CLOSE 1
710 LOAD "MENU.BAS",R
```

Program Description (continued)

Line 440 prompts you to turn on the cursor readout and line 450 sends the command to do this.

Line 460 prompts you to select horizontal bar cursors and line 470 sends the command to set the cursor type to horizontal bars.

Line 480 prompts you to place the cursors near the middle of the display and line 490 sends the command to place horizontal bar 1 at plus one division on the Y-axis and to place horizontal bar 2 at minus one division on the Y-axis.

Line 500 through 530 select dot-paired cursors and repositions them, setting dot 1 at an absolute value of minus four divisions on the X-axis and setting dot 2 at an absolute value of plus four divisions on the X-axis.

Lines 540 through 570 query the Y-coordinate values at dot 1 and dot 2 and the peak-to-peak measurement and displays these vertical values and their accuracy qualifiers to your computer screen.

Line 580 removes the header, link, and colon from the beginning of the returned string.

Line 590 and 600 calculates the peak-to-peak value based on cursor values extracted in line 580.

Lines 610 through 630 finds the peak-to-peak character string returned from lines 540 and 550 (that is, made with the automated measurement system) and extracts the peak-to-peak value, which is printed to your computer screen.

Lines 640 and 650 create a second trace (from Channel 2) and place it on a second lower graticule. Trace1 remains on the upper graticule.

Line 660 sends the commands to select Trace1, select split-dot cursors, and to use Trace2 as a reference.

Lines 670 through 690 inform you that you may use the cursors to make delay timing measurements, but recommend that you use the automatic measurement system, instead. In fact, a part of Example 9 demonstrates a method of taking delay measurements with the automatic measurement system (that is, the PDELAY or propagation delay measurement command).

Lines 700 and 705 prompt you that this example is finished and formally close the program. Line 710 returns you to the Examples Menu.

Example 7: Taking Delta-Delay Measurements

This example demonstrates the compare and reference features of the automated measurement system. This is particularly useful when you need to check a series of components or circuits to see if a particular measurement falls within an acceptable range.

For this example you need a CSA 803C or 11801C with at least one sampling head installed. Also, two SMA cables of different length will be used (2 ns and 5 ns cables are recommended). Use the short cable first, then later, when prompted, use the long cable.

Figure 4-7: Connections for Example 7

To start the program, select Example 7 from the Examples Menu. Refer to the earlier discussion *Running the Example Software* for information on using the Examples Menu.

Program Listing

```
100 LOCATE ,,1,0,7
102 PRINT "Example 7: Taking Delta-Delay Measurements"
103 PRINT
110 PRINT "Ensure that the cable is connected and the scope's parameters are"
120 PRINT "set accordingly : (major menu UTILITY, minor menu RS232 Parameters)"
130 PRINT "  Baud 9600,  Echo OFF,  Stop Bits 1,  Parity NONE,  Flagging HARD"
140 PRINT "  Delay 0,    EOL String CR/LF,       Verbose OFF"
150 INPUT "  Press Enter when ready",A$
160 A$=""
170 WHILE A$<>"COM1" AND A$<>"COM2" AND A$<>"com2"
180    INPUT "Press Enter for using COM1 (default), else type COM2. ",A$
190    IF A$="" THEN A$="COM1"
200 WEND
210 PRINT "We are opening ";A$;" at 9600 baud, no parity, and 1 stop bit."
220 OPEN A$+":9600,N,8,1" AS #1
230 INPUT "Press Enter to ask for the scope's ID",A$
240 PRINT #1,"ID?"
250 LINE INPUT #1,RESPONSE$
260 PRINT RESPONSE$

270 PRINT "Press Enter after making the following connections :"
280 PRINT " Use a power splitter or a power 'T' connecter to connect a 2ns and
    a 5ns"
290 PRINT " cable to channel 1 of sampling head 1 (connect the 2ns cable first)."
300 INPUT " Connect the 'T' or splitter to the CALIBRATOR of the 11800",A$
310 INPUT "Press Enter to initialize scope.",A$
320 PRINT #1,"INIT"

330 A$=""
340 WHILE A$<>"y" AND A$<>"Y" AND A$<>"n" AND A$<>"N"
350    INPUT "Do you want to watch the commands in DEBUG mode ? (y/n)",A$
360 WEND
370 IF A$= "y" OR A$= "Y" THEN PRINT #1,"DEBUG RS232:ON" ELSE PRINT #1,
    "DEBUG RS232:OFF"

380 INPUT "Press Enter to setup trace1 from sampling head 1, channel 1.",A$
390 PRINT #1,"TRACE1 DESCRIPTION:'M1'"

400 INPUT "Press Enter to set the trigger to the INTERNAL.",A$
410 PRINT #1,"TRIGGER SOURCE:INTERNAL"

420 INPUT "Press Enter to set to the time base to 10ns/div",A$
430 PRINT #1,"TBMAIN TIME:10E-9"
```

(continued)

Program Description

Lines 100 through 260 of Example 7 describe the same type of preparatory procedure as described in Example 1, program lines 100 through 260.

Lines 270 through 320 of this example remind you to make the necessary hardware setup and to initialize the instrument for this example.

Lines 330 through 370 wait for you to type Y or N to enable or disable `DEBug` mode then display the selected `DEBug` mode.

Line 380 waits for you to press 'Enter' to define Trace1 and line 390 sends the command to define Trace1 as Channel 1 of the sampling head.

Line 400 prompts you to select internal triggering for this example and line 410 sends the command to do this.

Line 420 prompts you to set the time per division for the main time base and line 430 sends the command to set the main time base to 10 ns/division.

(continued)

Program Listing (continued)

```
440 INPUT "Press Enter to change the measurement to h/w mode.",A$
450 PRINT #1,"MPARAM1 MMODE:HW"

460 INPUT "Press Enter to set the MSLIST to CROSS and save the value in memory.",A$
470 PRINT #1,"MSLIST1 CROSS;REFSET1 CURRENT:CROSS"

480 INPUT "Press Enter to turn COMPARE ON and bring up the measure menu.",A$
490 PRINT #1,"COMPARE ON;MSYS ON"

500 INPUT "Press Enter after removing the 2ns cable and connecting the 5ns
    cable.",A$

510 INPUT "Press Enter to ask the scope for the time difference between the
    cables.",A$
520 PRINT #1,"CROSS?"
530 LINE INPUT #1,RESPONSE$
540 PRINT RESPONSE$;" (in seconds)"

550 INPUT "Press Enter to change the Prop-Velocity to FEET.",A$
560 PRINT " (the default velocity is .7 * (speed of light)"
570 PRINT #1,"GRATICULE XUNIT:FEET"

580 INPUT "Press Enter after reconnecting the 2ns cable to CH1.",A$
590 INPUT "Press Enter to save the CROSS value in memory (in feet).",A$
600 PRINT #1,"REFSET1 CURRENT:CROSS"

610 INPUT "Press Enter after removing the 2ns cable and connecting the 5ns
    cable.",A$

620 INPUT "Press Enter to ask the scope for the feet difference between the
    cables.",A$
630 PRINT #1,"CROSS?"
640 LINE INPUT #1,RESPONSE$
650 PRINT RESPONSE$;" (in feet)"

660 PRINT "Example Completed."
665 CLOSE 1
670 LOAD "MENU.BAS",R
```

Program Description (continued)

Lines 440 and 450 select the hardware measurement mode for Trace1.

Lines 460 and 470 place the `CROss` measurement into the measurement list for Trace1 and save its current measurement value as a reference.

Lines 480 and 490 prompt you to turn on the compare measurements mode and then select the Measure menu.

Lines 500 through 540 prompt you to reconfigure your cable connection and send a `CROss` measurement query to the instrument to determine the difference between crossing points. This effectively measures the time difference (in seconds) between the two cables.

Lines 550 through 570 change the X-axis graticule X-axis scaling to feet which will allow the propagation delay to be measured in feet.

Line 580 prompts you to reconfigure your cable connection in preparation for measuring the length difference between the two cables.

Line 590 prompts you to save the current crossing value as the reference.

Line 600 sends the command to use the current crossing value for Trace1 as the reference measurement.

Line 610 prompts you to reconfigure your cable connection.

Lines 620 through 650 determine the difference between your new crossing value and the reference value.

Lines 660 and 670 prompt you that this example is finished and ends the program.

Lines 660 and 665 prompt you that this example is finished and formally close the program. Line 670 returns you to the Examples Menu.

Examples

Example 8: Comparing Traces

This example demonstrates storing a trace that represents a "snapshot" of a particular moment, and using that trace as a basis for comparison with other traces. This is similar to Example 7, where a reference measurement was used as a basis of comparison. This time, the entire trace will be saved as a reference.

For this example you will need a CSA 803C or 11801C with at least one sampling head installed, two SMA connecting cables, and a signal generator.

CSA 803C

11801C
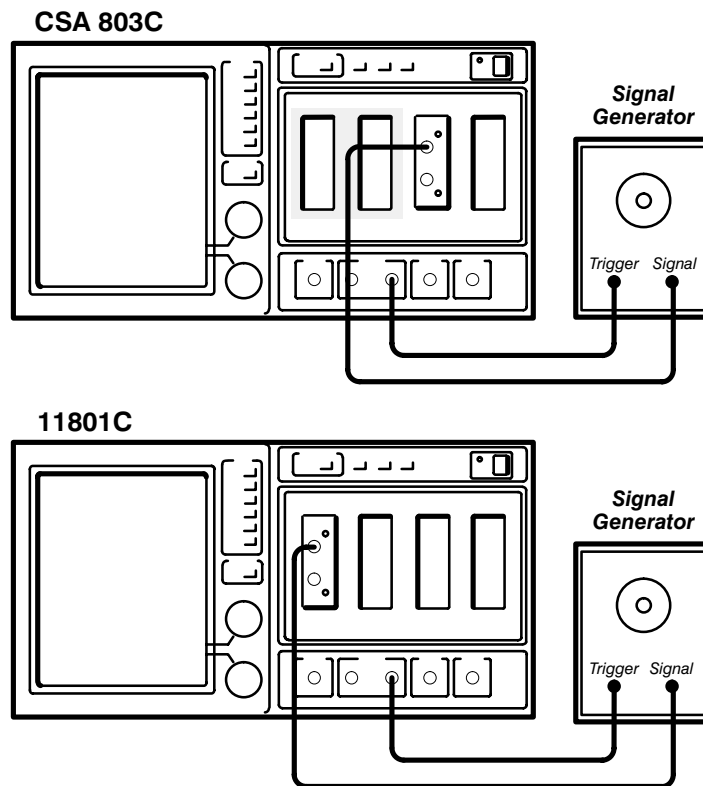
Figure 4-8: Connections for Example 8

To start the program, select Example 8 from the Examples Menu. Refer to the earlier discussion *Running the Example Software* for information on using the Examples Menu.

Program Listing

```
100 LOCATE ,,1,0,7
102 PRINT "Example 8: Comparing Traces"
103 PRINT
110 PRINT "Ensure that the cable is connected and the scope's parameters are"
120 PRINT "set accordingly : (major menu UTILITY, minor menu RS232 Parameters)"
130 PRINT "  Baud 9600,  Echo OFF,  Stop Bits 1,  Parity NONE,  Flagging HARD"
140 PRINT "  Delay 0,    EOL String CR/LF,        Verbose OFF"
150 INPUT "  Press Enter when ready",A$
160 A$=""
170 WHILE A$<>"COM1" AND A$<>"COM2" AND A$<>"com2"
180    INPUT "Press Enter for using COM1 (default), else type COM2. ",A$
190    IF A$="" THEN A$="COM1"
200 WEND
210 PRINT "We are opening ";A$;" at 9600 baud, no parity, and 1 stop bit."
220 OPEN A$+":9600,N,8,1" AS #1
230 INPUT "Press Enter to ask for the scope's ID",A$
240 PRINT #1,"ID?"
250 LINE INPUT #1,RESPONSE$
260 PRINT RESPONSE$270 PRINT "Press Enter after making the following connections :"
280 PRINT " Connect a 500kHz to 5MHz sinewave signal (.5V to 1V PP) to channel 1."
290 INPUT " Connect trigger out of your signal source to the TRIGGER INPUT of the
    scope.",A$
300 INPUT "Press Enter to initialize scope.",A$
310 PRINT #1,"INIT"320 A$=""
330 WHILE A$<>"y" AND A$<>"Y" AND A$<>"n" AND A$<>"N"
340    INPUT "Do you want to watch the commands in DEBUG mode ? (y/n)",A$
350 WEND
360 IF A$= "y" OR A$= "Y" THEN PRINT #1,"DEBUG RS232:ON" ELSE PRINT #1,
    "DEBUG RS232:OFF"
370 INPUT "Press Enter to setup trace1 from sampling head 1, channel 1.",A$
380 PRINT #1,"TRACE1 DESCRIPTION:'M1'"

390 INPUT "Press Enter to set the trigger to EXTERNAL",A$
400 PRINT #1,"TRIGGER SOURCE:EXTERNAL"

410 INPUT "Press Enter to send 'AUTOSET' to the scope",A$
420 PRINT #1,"AUTOSET MODE:PERIOD,START"
```

(continued)

Program Description

Lines 100 through 380 of Example 8 describe the same type of preparatory procedure (including the necessary hardware setup) as described in Example 1, program lines 100 through 360.

Line 390 prompts you to select external triggering for this example and line 400 sends the command to do this.

Line 410 prompts you to `AUTOSet` the selected trace and line 420 sends the command to begin autoset using period mode.

(continued)

Program Listing (continued)

```
430 INPUT "Press Enter to store trace1 to sto1.",A$
440 PRINT #1,"STORE TRA1:STO1"

450 INPUT "Press Enter to remove trace 1.",A$
460 PRINT #1,"REMOVE TRACE1"

470 INPUT "Press Enter to recall sto1.",A$
480 PRINT #1,"TRACE1 DESCR:'STO1'"

490 INPUT "Press Enter to create a difference trace showing only the noise.",A$
500 PRINT #1,"TRACE1 DESC:'STO1-M1'"
510 INPUT "Press Enter to find the RMS value of the noise over the WHOLE record.",A$
520 PRINT #1,"MPARAM1 DAINT:WHOLE;MSLIST1 RMS;MEAS1?"
530 LINE INPUT #1,RESPONSE$
540 PRINT RESPONSE$

550 PRINT "Example Completed."
555 CLOSE 1
560 LOAD "MENU.BAS",R
```

Program Description (continued)

Lines 430 and 440 store Trace1 as Stored Trace1.

Lines 450 and 460 remove Trace1 from the display.

Lines 470 and 480 recall Stored Trace1 to the display as Trace1.

Lines 490 and 500 create a trace that is the difference between Stored Trace1 and Channel 1.

Lines 510 through 540 measure the RMS value of Trace1. The measurement parameter data interval is set to WHOLE for Trace1 and the RMS measurement is included in the measurement list for Trace1, then all defined measurements for Trace1 (that is, RMS) are queried.

Lines 550 and 555 prompt you that this example is finished and formally close the program. Line 560 returns you to the Examples Menu.

Example 9: Taking TDR Measurements

This example demonstrates the TDR (time domain reflectometry) feature of the SD-24 sampling heads when they are used in combination with the CSA 803C. TDR is a method of examining and measuring a network or transmission line by sending a pulse into the network and monitoring the reflections.

For this example you will need a CSA 803C or 11801C with at least one SD-24 Sampling head installed. Also, you will need one SMA cable, preferably of 5 ns length.

CSA 803C

Not Connected □

11801C

Not Connected □

Figure 4-9: Connections for Example 9

To start the program, select Example 9 from the Examples Menu. Refer to the earlier discussion *Running the Example Software* for information on using the Examples Menu.

Program Listing

```
100 LOCATE ,,1,0,7
102 PRINT "Example 9: Taking TDR Measurements"
103 PRINT
110 PRINT "Ensure that the cable is connected and the scope's parameters are"
120 PRINT "set accordingly : (major menu UTILITY, minor menu RS232 Parameters)"
130 PRINT "  Baud 9600,  Echo OFF,  Stop Bits 1,  Parity NONE,  Flagging HARD"
140 PRINT "  Delay 0,    EOL String CR/LF,       Verbose OFF"
150 INPUT "  Press Enter when ready",A$
160 A$=""
170 WHILE A$<>"COM1" AND A$<>"COM2" AND A$<>"com2"
180    INPUT "Press Enter for using COM1 (default), else type COM2. ",A$
190    IF A$="" THEN A$="COM1"
200 WEND
210 PRINT "We are opening ";A$;" at 9600 baud, no parity, and 1 stop bit."
220 OPEN A$+":9600,N,8,1" AS #1
230 INPUT "Press Enter to ask for the scope's ID",A$
240 PRINT #1,"ID?"
250 LINE INPUT #1,RESPONSE$
260 PRINT RESPONSE$
270 PRINT "Press Enter after connecting your cables to the scope. Requirements:"
280 INPUT "   Connect a 5ns SMA cable to CHM1 (leave it open ended)",A$
290 INPUT "Press Enter to initialize scope.",A$
300 PRINT #1,"INIT"
310 A$=""
320 WHILE A$<>"y" AND A$<>"Y" AND A$<>"n" AND A$<>"N"
330    INPUT "Do you want to watch the commands in DEBUG mode ? (y/n)",A$
340 WEND
350 IF A$= "y" OR A$= "Y" THEN PRINT #1,"DEBUG RS232:ON" ELSE PRINT #1,
    "DEBUG RS232:OFF"
360 INPUT "Press Enter to setup trace1 from sampling head 1, channel 1.",A$
370 PRINT #1,"TRACE1 DESCRIPTION:'M1'"

380 INPUT "Press Enter to set the trigger to INTERNAL",A$
390 PRINT #1,"TRIGGER SOURCE:INTERNAL"

400 INPUT "Press Enter to turn on the TDR pulse",A$
410 PRINT #1,"CHM1 TDRSTATE:ON"

420 INPUT "Press Enter to change the x to FEET",A$
430 PRINT #1,"GRATICULE XUNIT:FEET"
```

(continued)

Program Description

Lines 100 through 370 of Example 9 describe the same type of preparatory procedure (including the necessary hardware setup) as described in Example 1, program lines 100 through 360.

Line 380 prompts you to select internal triggering for this example and line 390 sends the command to do this.

Line 400 prompts you to to turn on the time delay reflectometry pulse and line 410 sends the command to turn on the TDR pulse for Channel 1.

Lines 420 and 430 change the graticule X-axis scaling to feet.

(continued)

Program Listing (continued)

```
440 INPUT "Press Enter to calibrate CHM1 for TDR amplitude",A$
450 PRINT #1,"CALM1 TDRAMPLITUDE:350E-3"

460 INPUT "Press Enter to send AUTOSET",A$
470 PRINT #1,"AUTOSET START"

480 INPUT "Press Enter to put CROSS in the measurement list",A$
490 PRINT #1,"MSLIST1 CROSS"

500 INPUT "Press Enter to set the measurement parameters",A$
510 PRINT #1,"MPARAM1 MLEVELMODE:ABSOLUTE,REFLEVEL:-300E-3"

520 INPUT "Press Enter to save one timing value to memory",A$
530 PRINT #1,"REFSET1 CURRENT:CROSS"

540 INPUT "Press Enter to turn COMPARE ON",A$
550 PRINT #1,"COMPARE ON"

560 INPUT "Press Enter to adjust the reference level",A$
570 PRINT #1,"MPARAM1 REFLEVEL:-50E-3"

580 INPUT "Press Enter to ask the scope for the measurement",A$
590 PRINT #1,"MEAS1?;GRAT? XUNIT"
600 LINE INPUT #1,RESPONSE$
610 PRINT RESPONSE$

620 PRINT "Example Completed."
625 CLOSE 1
630 LOAD "MENU.BAS",R
```

Program Description (continued)

Lines 440 and 450 calibrate TDR pulse amplitude for Channel 1 by sending the command to set TDR amplitude for Channel 1 to 60% (that is, obtain a 250 mV TDR pulse amplitude).

Lines 460 and 470 `AUTOSet` Trace1.

Lines 480 and 490 add the `CROss` measurement to the measurement list for Trace1.

Lines 500 and 510 select the measurement parameters for Trace1 by sending commands to set the measurement level mode for Trace1 to absolute, and the measurement reference level to -300 mV.

Lines 520 and 530 save the Trace1 current crossing value as the reference value.

Lines 540 and 550 turn on the measurement comparison mode.

Lines 560 and 570 set the reference level for Trace1 to -50 mv.

Lines 580 through 610 query the instrument for the current crossing measurement value and the X-axis graticule scaling for Trace1. This response, in feet, represents the distance the TDR pulse must travel out the connector, through the cable, and back again. To find only the actual length of the cable, locate the start of the cable. Then measure the distance to the rising edge of the second step. Finally, take this value and divide by two.

Lines 620 and 625 prompt you that this example is finished and formally close the program. Line 630 returns you to the Examples Menu.

Examples

Examples

Appendix A:
Improving System Performance

Optimum system performance means acquiring accurate data with the fastest system throughput. This appendix discusses the components of system performance and suggests techniques to improve each.

First, you must be familiar with your instrument controller, measurement instruments, data recorders, and with your chosen software (operating system, programming language, device drivers, etc.). When you know the capabilities of your system, you are better prepared to write efficient application programs.

Then you must decide which interface (GPIB or RS-232-C) best suits your application needs.

A good way to develop a thorough understanding of your system instruments is to study their manuals. In particular, learn about the command vocabulary and data formats (for example, ASCII or binary), for each instrument and learn how each device buffers data and executes commands. This gives you information about which hardware configurations and program algorithms will be most efficient for your application.

Components of System Performance

Five major components affect the overall system performance, as summarized in Figure A-1. The sum of these components is the total time required to execute your application.
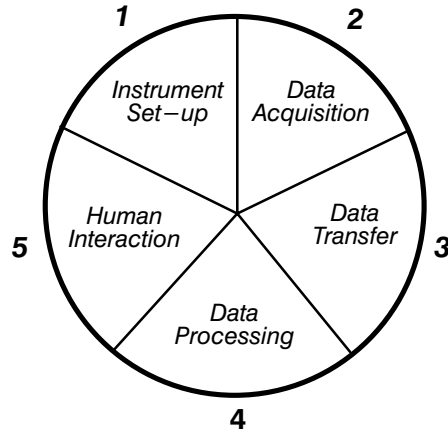
Figure A-1: System Performance Components

The contribution of each component to the total execution time varies, based on your specific system configuration.

For example, a data logging system generally requires little time to set up and doesn't require operator intervention. However, significant time is spent acquiring and transferring data. In contrast, a production test system may spend less time acquiring data, but more time processing data and interacting with the operator. Each situation requires a different focus for optimizing system performance.

The best way to determine the time that each component contributes to system performance is to measure it. You can use a real time clock in your controller to do this.

For example, to measure the time it takes to execute a `PP?` (peak-to-peak amplitude) measurement query, turn on your controller real-time clock before the command, then read the elapsed time immediately after reading the `PP?` response. Repeating this measurement a few times under varying system configurations will produce typical values you can use to judge the impact of each component on system performance.

Instrument Setup Time

Instrument setup time can be divided into two parts: the time required to decode and execute a setting command, and the time required for new settings to stabilize.

The time it takes to decode and execute a single instrument command is usually short, but if a command initiates a complex or lengthy operation, it can increase the setup time.

For instance, some commands require the instrument to check whether any settings associated with the command function have changed prior to the command. If any associated settings have changed, the instrument must load the new settings into its hardware.

The second part of the setup time is the time it takes the instrument to settle to the specified setting. For example, when vertical size is set automatically, the instrument takes a reading of the input voltage, tests for under- or over-voltage conditions, steps the vertical scale range up or down, and takes another reading. Several readings might have to be made until the correct range is determined. The process stops once the reading is within the new vertical scale range. Thus, a single change in test conditions can cause a significant change in setup time.

Optimizing Setup Time — requires reducing the number of setting changes or reducing the time required for the instrument to execute the setting changes.

Here are some suggestions to optimize setup time:

- Group tests that use common settings.

- Set your ranges explicitly. Generally, autoset takes more time.

- First set up instruments that require more settling time. While they are settling, you can be setting up other devices.

- Use the store setting features. Reconstructing a setting takes more time.

- Use low byte-count and less complex commands. For example, use the `LONgform OFF` command for abbreviated responses to queries. This can significantly reduce the byte count for data transfers.

Data Acquisition Time

The second component of system performance is the time required to acquire a full record of the input source (the selected waveform). This is the data acquisition time. See Figure A-2.

For sequential mode digitizing, the significant factors that affect the digitizing time are the frequency of the trigger events, and the horizontal size and record length selection.

The CSA 803C and 11801C sample the incoming waveform every five µs, or every valid trigger (equal to the trigger period), depending on the trigger mode.

The digitizing time is equal to the number of samples taken (equal to the record length) times the sample interval or the right-most dot/sample horizontal position (that is, acquisition end).
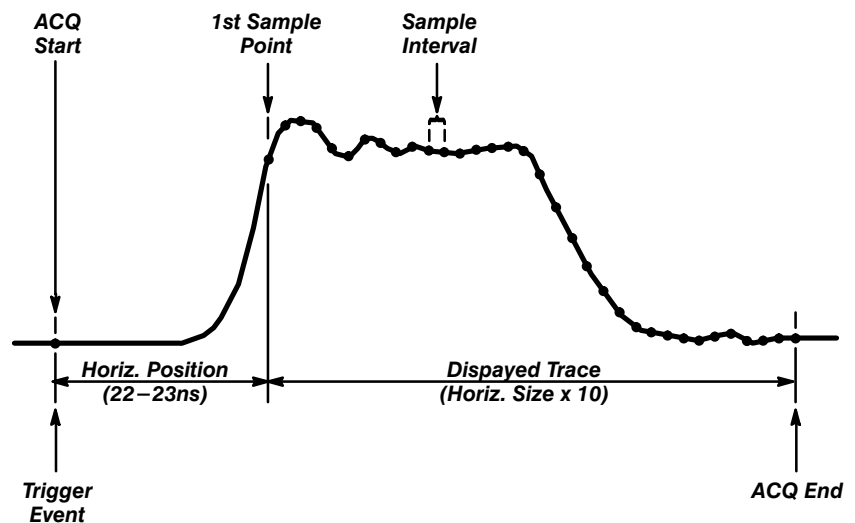
Figure A-2: Components of Data Acquisition Time

Optimizing Data Acquisition Time — requires careful attention in setting up the acquisition.

Here are some suggestions to optimize data acquisition time:

- Faster digitizing can be achieved by increasing the repetition rate of the trigger, if possible.

- Use an operation-complete SRQ interrupt instead of waiting for the acquisition to finish. You can continue processing while the acquisition completes.

Data Transfer Time

The third component of system performance is the time it takes to transfer data from one instrument to another. The data transfer time depends on two factors: the number of bytes being transferred and the time it takes to transfer each byte.

The number of bytes transferred depends on the size of the message (number of characters) and the data format (for example, ASCII or binary). For GPIB transfers, the transfer rate depends on the speed of the slowest addressed device on the bus. For RS-232-C transfers, the data transfer rate depends on the baud rate setting of the instrument and controller.

Understanding the processing of GPIB and RS-232-C I/O statements is the key to estimating data transfer times.

GPIB I/O Execution Time — consists of five parts:

- Addressing sequence

- Unaddressing sequence

- Statement overhead

- Buffer overhead

- Data overhead

The addressing and unaddressing sequences are composed of GPIB interface messages that make the instrument talk or listen to the controller. The time required depends on the data handshake rate of the slowest device connected to the bus.

Statement overhead is the time required to examine the I/O statement for content and syntax (parsing). For the controller, this includes evaluating the statement's I/O function(s) and other expressions, and the statement clauses (instrument commands).

Buffer overhead is the time it takes to fill or empty an I/O memory register with the I/O statement. This depends on the the amount of data (how many characters), and the type of data (string or numeric, ASCII or binary).

Data overhead is the time it takes to transfer data over the interface bus. Again, the time depends on the data transfer rate of the slowest device involved in the transfer, and on the amount and type of data transferred (for example, numeric arrays are a little faster than an equivalent number of scalar variables). This includes the spaces and formatting characters for each message. The total data transmission time is the number of bytes being transferred divided by the data transfer rate (in bytes/second).

RS-232-C I/O Execution Time — consists of five parts, similar to the GPIB:

- Statement overhead

- Buffer overhead

- Start message

- Data overhead

- Stop message

The RS-232-C statement and buffer overheads consist of the same elements as in GPIB I/O.

The start and stop message time consists of the time required to send one or two bits (depending on the configuration of the RS-232-C interface) before and after each byte of the message in order to synchronize the transmission.

The RS-232-C data overhead time is determined by the baud rate setting of the RS-232-C port on each device.

Since data are sent serially over the RS-232-C interface, additional time is required to convert information from serial-to-parallel (for input data) and from parallel-to-serial (for output data). Thus, throughput for an RS-232-C message tends to be slower than throughput for the same GPIB message.

Optimizing Data Transfer Time — involves two major areas. The first is the system configuration, and the second is the program that controls the transfer.

These suggestions will help you optimize the system configuration:

- Choose instruments that have an optimum transfer rate as near as possible to the bus capacity.

- If your controller has more than one GPIB port, use frequently interacting devices on one bus, or put faster devices together on one bus.

- Use direct-memory access (DMA) transfers whenever possible and keep the faster instruments on this bus.

- Be sure to unaddress slow devices when they are not required in the transfer.

- If you have two ports, put a device under test (DUT) on one bus, and the test equipment on the other bus. Then, if the DUT has an error or malfunction, it won't affect the test equipment.

Follow these suggestions to optimize transfer program parameters:

- Choose the most efficient I/O statements that your controller provides. In most cases high-level commands are fastest, except where long strings are encountered. Then use low-level transfer commands (if provided).

- Minimize bus traffic by reducing the number of bytes being sent. You can do this by abbreviating command names, deleting unnecessary spaces, and omitting unnecessary zeros.

- Minimize buffer overhead. This can be done by defining buffer size (usually possible for most controllers) to accommodate the entire data transfer. You may also store the data within a string variable; string variables store data directly from the I/O buffer and reduce overhead time.

- Use binary block data transfers if possible. Binary data is a little more complicated to handle than ASCII data, but binary transfers tend to be much faster because they involve fewer bytes than an equivalent ASCII transfer.

Data Processing Time

The fourth component of system performance is the time required to manipulate the acquired data for a desired result.

The data processing time is composed of the time it takes the instrument to manipulate the data, plus the time required by the controller to further process the data. The instrument can deliver raw, semi-processed data, or completely processed data, depending on the requirements of the application. The processing speed of the instrument depends on the type or complexity of the operation performed.

Optimizing Data Processing Time — involves using faster algorithms and distributed processing.

These suggestions will help you optimize data processing time:

- Evaluate your choice of algorithms to ensure that the most efficient operations are used for your application and system configuration.

- Use implied array operations instead of for/next loops in your controller programs. This allows numeric operations to be performed much faster. The implied array operation creates temporary arrays to perform the implicit operation (for example, add a scalar to the array) rather than an element-by-element operation.

- Carefully select the data type for your controller programs. Try to group integer, short floating-point, and long floating-point operations. It takes less time to process each as a group than to do mixed data type operations that require conversion from one format to another.

- Evaluate your measurement needs to identify the most effective device for each data processing task. For example, would the instrument best perform a given function on a waveform, or would your controller perform that function more quickly?

Human Interaction Time

The fifth major component of system performance is determined by operator intervention required to enter test parameters or to make adjustments to a device under test (DUT).

This component can easily become the largest part of the total operating time for a system. Direct measurement of this component is the best way to determine its effect on system performance.

Optimizing Human Interaction Time — can be difficult. The best advice is to avoid the need for human interaction with the system as much as possible.

Follow these suggestions to optimize human interaction time:

- Use programmable interfaces and switches to route signal connections wherever possible. These include programmable relay scanners, multi-function interfaces, and signal multiplexers.

- Keep the user interface simple. The instrument is designed especially for this purpose. User menus are quick and easy to use, so you can make changes quickly.

Appendix B:
Reserved Words

Reserved words represent the entire set of CSA 803C and 11801C prede-fined command words, including headers, links, and arguments.

In this section, reserved words appear in mixed case, with the required minimum leading substring in uppercase. Any leading substring of the full reserved word will be accepted as input, so long as the minimum string is given.

A
ABBwfmpre
ABOrt
ABSLevel
ABSolute
ABSTouch
AC
ACCumulate
ACQNum
ACQuisition
ACTive
ADJtrace
ALL
ALLFps
ALLSTD
ALLSTO
ALLTrace
ALL_wavfrm
ALTErnate
ALTinkjet
AMPlitude
ASCii
ATTenuation
AUTO
AUTOSet
AVG
AVG.env

B
BASeline
BAUd
BCOrrection
BINary
BINHex
BIT/nr

BITMap
BLACkman
BLHarris
BLOwby
BN.fmt
BOTh
BYPassed
BYT/nr
BYT.or

C
CALibrate
CC.1.5mb
CC.32mb
CC.34mb
CC.44mB
CC.8mb
CC.97mb
CC.CO.2mb
CC.CO.6mb
CC.ONe.139mb
CC.SY.2mb
CC.SY.6mb
CC.ZEro.139mb
CDElay
CDSync
CENter
CENTRonics
CEPT
CH
CHIdent
CHKsm0
CLEar
CMDerr
COLor
COMpare
CONDacq

CONFig
CONTinuous
COPy
COUnt
CR
CREate
CRLf
CROss
CRVChk
CRVId
CSTore
CUR1
CURRent
CURSor
CURVe
C.Points
C.WINBottom
C.WINLeft
C.WINRight
C.WINTop

D
DADj
DAFiltering
DAInt
DAMeasref
DATA
DATACompress
DATAFormat
DATAType
DATE
DATRdy
DB
DBM
DBVPeak
DBVRms
DC

DCOmp
DEBug
DEFine
DEFAult
DEGrees
DELAy
DELete
DEScription
DIAg
DIRection
DISPlay
DISPNum
DISPErsion
DISPLAY_modes
DISTal
DIThered
DIV2
DIVS
DOT1Abs
DOT1Rel
DOT2Abs
DOT2Rel
DOTs
DRAft
DREcall
DS1
DS1C
DS1.Old
DS2
DS3
DS4NA
DS4XNA
DSYmenu
DSYS
DUAl
DUTy
D.Points

D.WINBottom
D.WINLeft
D.WINRight
D.WINTop

E
ECHo
EDGe
EMPty
ENCdg
ENH_accuracy
ENV
EOL
EQ
ER
ERMethod
EVEN
EVENT
EXErr
EXTAttenuation
EXTCoupling
EXTernal
EXWarn
EYEPattern

F
FAlled
FALltime
FASt
FEEt
FEOi
FDDI
FFT
FILtering
FLAgging
FMOde
FORCe

FORMat
FPAnel
FPNext
FPS
FPSList
FPSNum
FPUpdate
FREq

G
GPIb
GRADed
GRADFirst
GRADScale
GRAticule
GRLocation
GT

H
H1Bar
H2Bar
HAMming
HANning
HARd
HARMonic
HBArs
HERez
HIFreq
HIPrec
HIRes
HISTogram
HIST.pt
HISTScaling
HMAg
HMOde
HNUmber
HOLdoff
HORiz
HP560C
HPGl
HPOsition
HREfpt
HUE
HW

I
ID
IDLe
IEEE
INActive
INChes

INErr
INFinite
INIt
INPut
INTENsity
INTernal
INWarn

J
JITter
JITTLOcation
JITTLEvel
JITT.histpt

K
KILI

L
LABAbs
LABel
LABRel
LEFt
LENgth
LEVel
LF
LFCr
LIGhtness
LINear
LMOde
LMZone
LOCate
LOG10
LONgform
LOOpgain
LOWer
LSB
LT

M
MAIn
MAINPos
MANual
MASK
MASKDefine
MASKStat
MAX
MAXTranum
MEAN
MEAS
MESial
METers

MFIltering
MID
MIN
MINUs
MLEvelmode
MMOde
MODe
MPAram
MSB
MSLIst
MSLOpe
MSNum
MSYs
MTRack
MTRANs

N
NAVg
NCOunt
NCUrrent
NENv
NF
NGRAded
NHISt.pt
NMAsk
NOIse
NOISLocation
NOIS.histpt
NONe
NORmal
NOTrg
NR.pt
NT
NULl
NVRam
NWAVfrm
NWFm

O
OC1
OC12
OC18
OC24
OC3
OC36
OC48
OC9

ODD
OFF
OFFSet
OHMS
ON
ONCE

ONUll
OPCmpl
OPTional
OQUal
OR
OUTput
OVErshoot

P
PAIred
PANzoom
PARity
PASsed
PCTg
PDElay
PEAK
PERiod
PERSistence
PHAse
PINdex
PIN8
PIN24
PLUs
PORt
POS
POWeron
PP
PREscaler
PRINter
PINTIng
PROPvelocity
PROXimal
PT.fmt

R
RATe
REAdout
RECall
RECOrd
RECTangular
REDuced
REFAmplitude
REFBaseline
REFErence
REFFiltering
REFLEvel
REFLMzone
REFMesial
REFREsh
REFRmzone
REFSEt
REFSLope

REFSNratio
REFTOpline
REFTRace
REFXsition
RELative
REMAining
REMove
RESet
RHO
RHOFactor
RHOPos
RHOZero
RI
RIGht
RISetime
RMS
RMSDev
RMZone
RQS
RS232
RUN

S
SAMid
SATuration
SCReen
SEConds
SELect
SENsitivity
SET
SETSeq
SETZero
SFRequency
SHIft
SIGMA1
SIGMA2
SIGMA3
SINgle
SLOpe
SMAgnitudeSMOde
SMOFfadj
SMOothing
SNRatio
SOFt
SONet
SOUrce
SPEaker
SPLit
SPOoling
SRQmask
STARt

STAT
STATHist
STATIstics
STATUs
STByte
STD
STDDev
STM1
STO
STOList
STONum
STOP
STOPBits
STORe
STORE_recall
STRing
STS1
STS3
STSX3
SW

T
TBCalmode
TBMain
TBWin
TDRAmplitude
TDRDelay
TDRPolarity
TDRState
TESt
TEXt
TF
THD
TIFf
TIMe
TO
TOFfset
TOPBase
TOPline
TOTal
TOTRAce
TRAce
TRACK
TRAList
TRANS
TRANUm
TRG
TRIAngular
TRIgger
TRSep
TYPe

U
UID
UN
UNDEF
UNDershoot
UNDO
UPPer
UPTime
URecall
USEr
USEREye
USERMask
UTILITY1
UTILITY2

V
V1Bar
V2Bar
VARiable
VBArs
VECtors
VERBose
VERt
VOLts
VPCurve
VPEak
VPOsition
VSIze

W
WAIT
WAVfrm
WFId
WFMCalc
WFMPre
WFMScaling
WHOle
WIDth
WIN
WINDow
WINList
WINNum

X
X1
X10
X2
X5
XCOord
XDIv
XINcr

XMUlt
XQUal
XSize
XTNd
XUNit
XY
XZEro

Y
YCOord
YDIv
YMUlt
YQUal
YSize
YTEnergy
YTMns_area
YTPls_area
YUNit
YZEro

Z
ZEROPoint

Appendix C:
Character Sets

The character sets include standard ASCII characters and a special set of characters that include math, Greek, European, Spanish, and graphic symbols.

The special "escape" characters are formed by putting an ASCII escape character (decimal 27) in front of another ASCII character. For example, to place an integral math symbol (∫) on the instrument display, enter an escape character (represented by <ESC>) followed by the letter **d**.

```
TEXT STRING:"<ESC>d"
```

For more information on placing characters on the display, see the TEXt command entry in the *Command Set* chapter.

The character-set tables begin on the following page.

ASCII Character Set

	0	1	2	3	4	5	6	7
0	0 NUL	16 DLE	32	48 0	64 @	80 P	96 `	112 p
1	1 SOH	17 DC1	33 !	49 1	65 A	81 Q	97 a	113 q
2	2 STX	18 DC2	34 "	50 2	66 B	82 R	98 b	114 r
3	3 ETX	19 DC3	35 #	51 3	67 C	83 S	99 c	115 s
4	4 EOT	20 DC4	36 $	52 4	68 D	84 T	100 d	116 t
5	5 ENQ	21 NAK	37 %	53 5	69 E	85 U	101 e	117 u
6	6 ACK	22 SYN	38 &	54 6	70 F	86 V	102 f	118 v
7	7 BEL	23 ETB	39 '	55 7	71 G	87 W	103 g	119 w
8	8 BS	24 CAN	40 (	56 8	72 H	88 X	104 h	120 x
9	9 HT	25 EM	41)	57 9	73 I	89 Y	105 i	121 y
A	10 LF	26 SUB	42 *	58 :	74 J	90 Z	106 j	122 z
B	11 VT	27 ESC	43 +	59 ;	75 K	91 [	107 k	123 {
C	12 FF	28 FS	44 ,	60 <	76 L	92 \	108 l	124 |
D	13 CR	29 GS	45 −	61 =	77 M	93]	109 m	125 }
E	14 SO	30 RS	46 .	62 >	78 N	94 ^	110 n	126 ~
F	15 SI	31 US	47 /	63 ?	79 O	95 _	111 o	127 ⌑

Escape Character Set

	0	1	2	3	4	5	6	7
0	0 Ä	16 Ñ	32 *Color 1	48	64 Π	80 π	96 ↓	112 ⋯
1	1 ä	17 ñ	33 *Color 2	49	65 α	81 θ	97 ↑	113 Ä
2	2 Ö	18 ˌ	34 *Color 3	50	66 γ	82 ρ	98 →	114 Ⓔ
3	3 ö	19 ì	35 *Color 4	51	67 δ	83 Σ	99 ←	115 Ⓡ
4	4 Ü	20 Ã	36 *Color 5	52	68 Δ	84 τ	100 ∫	116 ⊓
5	5 ü	21 ã	37 *Color 6	53	69 ε	85 υ	101 ÷	117 ⌐
6	6 à	22 Â	38 *Color 7	54	70 φ	86 ν	102 °	118 ¬
7	7 è	23 õ	39	55	71 Γ	87 ω	103 √	119 ⌐
8	8 á	24 õ	40	56	72 θ	88 χ	104 ⌐	120 ∟
9	9 é	25 É	41	57	73 ι	89 ξ	105 ±	121 ┼
A	10 Â	26 Ø	42	58	74 ψ	90 ζ	106 ≠	122 —
B	11 å	27 ø	43	59	75 κ	91 Φ	107 ≤	123 ├
C	12 Æ	28 Œ	44	60	76 λ	92 Λ	108 ≥	124 ┤
D	13 æ	29 œ	45	61	77 μ	93 Ψ	109 ©	125 ⊥
E	14 ç	30 Ç	46	62	78 η	94 σ	110 ®	126 ┬
F	15 β	31 ∞	47	63 x̄	79 Ω	95 Ξ	111 ≈	127	

*These color indices can be used to produce colored text on the display with the `TEXt` command. The color indices correspond to the display colors defined with the `COLor<ui>` command. See the Color Indexes table in the `COLor<ui>` discussion in the *Command Set* chapter.

To produce colored text, preceed text with one of these escape characters. The `TEXt` command allows multiple color changes per line of text. Refer to the *Command Set* chapter.

Appendix D:
Utility Programs

These are some common operations that can be performed from the external interface:

- Taking measurements

- Binary waveform transfer into an array

- Storing and recalling front panel settings

- Handling SRQs (instrument service requests)

- String transfer to the instrument display

The following programs demonstrate these utilities on popular instrument controllers.

Setup

These applications are for use with IBM PC-compatible computers configured with a National Instruments GPIB-PC Interface Card. A compatible computer with a similar GPIB interface card can also be used. These programs are written in Microsoft QuickBASIC, Version 4.0.

We also show Hewlett-Packard 200 and 300 Series controller versions of these programs. These programs are written in HP BASIC, Versions 2.1 through 4.0.

Interface Configuration

Set up the GPIB parameters of the instrument as shown in Table D-1.

Table D-1: CSA 803C or 11801C Interface Configuration

GPIB Function	Selection
Mode	TalkListen
Address	1
Terminator	EOI/LF
Debug	Off

Computer Interface Configurations

The following information describes how to set up your GPIB driver system for using these programs.

IBM PC-Compatible Computers — require you to invoke the configuration program for your GPIB interface. For example, for the National Instruments GPIB-PC Interface Card, invoke the ibconfig.exe file and follow the instructions.

The following illustrations show the appropriate configuration for using these utility programs.

Figure D-1 shows how your GPIB driver board characteristics should be set, and Figure D-2 shows how your device (instrument) characteristics should be set.

```
 National Instruments │  Board Characteristics  │     IBM PC-AT

 Board: GPIB0                          SELECT (use right/left arrow keys):

 Primary GPIB Address .......... 0
 Secondary GPIB Address ........ NONE
 Timeout setting ............... T10s
 EOS byte ...................... 00H
 Terminate Read on EOS ......... no
 Set EOI with EOS on Write ..... no
 Type of compare on EOS ........ 7-bit
 Set EOI w/last byte of Write .. yes
 GPIB-PC Model            ↔ PC2A        PC2 or PC2A
 Board is System Controller .... yes
 Local Lockout on all devices .. no
 Disable Auto Serial Polling ... yes
 High-speed timing ............. yes
 Interrupt jumper setting ...... 7
 Base I/O Address .............. 02E1H
 DMA channel ................... 1


 F1: Help      F2: Explain Field     F6: Reset Value      F9: Return to Map
```

Figure D-1: GPIB Driver-Board (GPIB) Settings

```
┌─────────────────────┬─────────────────────────┬──────────────────────┐
│ National Instruments │   Device Characteristics │      IBM PC-AT       │
├─────────────────────┴─────────────────────────┼──────────────────────┤
│ Device: TEK11K         Access: GPIB0           │ SELECT (use right/left arrow keys): │
├────────────────────────────────────────────────┼──────────────────────┤
│ Primary GPIB Address              * 1          │ 0 to 30              │
│ Secondary GPIB Address ........ NONE           │                      │
│ Timeout setting .............. T10s            │                      │
│ EOS byte ..................... 00H             │                      │
│ Terminate Read on EOS ......... no             │                      │
│ Set EOI with EOS on Write ..... no             │                      │
│ Type of compare on EOS ........ 7-bit          │                      │
│ Set EOI w/last byte of Write .. yes            │                      │
│                                                │                      │
│                                                │                      │
│ F1: Help      F2: Explain Field    F6: Reset Value    F9: Return to Map │
└────────────────────────────────────────────────┴──────────────────────┘
```

Figure D-2: GPIB Driver-Device (CSA 803C or 11801C) Settings

HP 200/300 Series Controllers — These programs require you to load the accompanying "I/O" file for your controller.

NOTE

Refer to your HP 200 or 300 Series controller programming manual for configuration details.

NOTE

In these examples, it is assumed that the "@Br" and "BR%" variables identify the instrument assigned to the GPIB port of your controller.

IBM PC-Compatible Controller

The following five program examples are for IBM controllers.

Taking Measurements

```
CALL IBFIND("tek11k", bd%)
CALL IBWRT(bd%, "MSLIST1 PER,FREQ,MAX,PP,RISE,
 FALL;MEAS1?")
msg$ = SPACE$(200)
CALL IBRD(bd%, msg$)
PRINT msg$
END
```

Transferring a Binary Waveform into an Array

```
REM WFM I/O for the 11k scope using Microsoft
   QuickBASIC 4.0 & BC 6.0
CALL ibfind("tek11k", bd%)
CALL ibwrt(bd%, "LONGFORM ON;SELECT?")
msg$ = SPACE$(80): CALL ibrd(bd%, msg$)
CALL ibwrt(bd%, "ENCDG WAV:BIN;BYT.OR LSB;
 OUTPUT   " + MID$(msg$, 8, 6))

CALL ibwrt(bd%, "CURVE?")
CALL ilrd(bd%, msg$, 20)
 hbyte$ = " ": lbyte$ = " "
CALL ilrd(bd%, hbyte$, 1): CALL ilrd(bd%, lbyte$, 1)
bytes = ASC(hbyte$) * 256 + ASC(lbyte$)
nr.pt = (bytes - 1) / 2
DIM wfm%(nr.pt)
CALL ibrdi(bd%, wfm%(), bytes)
CALL ilrd(bd%, msg$, 1)

SCREEN 2:   WINDOW (0, -32767)-(nr.pt, 32767)
PSET (0, wfm%(0))
FOR i = 0 TO nr.pt - 1: PSET (i, wfm%(i)): NEXT
END
```

Storing and Recalling Front Panel Settings

```
CALL ibfind("tek11k", bd%)
CALL ibwrt(bd%, "ENCDG SET:BINARY;SET?")
msg$ = SPACE$(5000)
CALL ibrd(bd%, msg$)
INPUT "Press Enter to send the setup back to the
scope", A$
CALL ibwrt(bd%, msg$)
END
```

Handling SRQs

```
CALL ibfind("tek11k", bd%)
CALL ibwrt(bd%, "SRQMASK USER:ON;RQS ON")
PRINT "Press the RQS icon on the CSA 803C/11801C (Esc to
  exit)"
WHILE INKEY$<>CHR$(27)
    GOSUB POLL
WEND
END

POLL:
 msg$ = SPACE$(80)
 stbyte%=0
 call ibrsp(bd%, stbyte%)
 IF stbyte%<>0 THEN
   CALL ibwrt(bd%, "EVENT?")
   CALL ibrd(bd%, msg$)
   PRINT "Status byte:";stbyte%
   PRINT msg$ : PRINT
 END IF
RETURN
```

Transferring a String to the CSA 803C/11801C Display

```
CALL ibfind("tek11k", bd%)
x = 5: REM x: {0 to 49}
y = 5: REM y: {0 to 31}
text$ = "'hello there world'"
msg$ = "text x:" + STR$(x) + ",y:" + STR$(y) +
",string:" + text$
CALL ibwrt(bd%, msg$)
END
```

HP 200 & 300 Series Controllers

The following five program examples are for HP controllers.

Taking Measurements

```
10 DIM Meas$[200]
20 ASSIGN @Br TO 701;EOL CHR$(10) END
30 OUTPUT @Br;"MSLIST1 PER,FRE,MAX,PP,RISE,FALL"
40 OUTPUT @Br;"MEAS1?"
50 ENTER @Br;Meas$
60 PRINT Meas$
70 END
```

Transferring a Binary Waveform into an Array

```
 10 ASSIGN @Br TO 701;EOL CHR$(10) END
 20 ASSIGN @Brbin TO 701;FORMAT OFF
 30 OUTPUT @Br;"LONGFORM ON"
 40 OUTPUT @Br "SELECT?"
 50 ENTER @Br;Trace$
 60 OUTPUT @Br;"ENCDG WAVFRM:BIN;BYT.OR MSB;
      OUTPUT "&Trace$[8]
 70 OUTPUT @Br;"CURVE?"
 80 ENTER @Br USING "#,20A,W";Header$,Bytcnt
 90 Nr_pt = (Bytcnt-1)/2
100 ALLOCATE INTEGER Curve(1:Nr_pt)
110 ENTER @Brbin;Curve(*)
120 ENTER @Br USING "B";Cksum
130 PRINT Curve(*)
140 DEALLOCATE Curve(*)
150 END
```

Storing and Recalling Front Panel Settings

```
10 DIM Setting$[5000]
20 ASSIGN @Br TO 701;EOL CHR$(10) END
30 OUTPUT @Br;"ENCDG SET:BINARY;SET?"
40 ENTER @Br USING "-K";Setting$
50 DISP "press CONTINUE to reset the front panel" 60 PAUSE
70 OUTPUT @Br;Setting$
80 END
```

Handling SRQs

```
 10 DIM Event$[100]
 20 ASSIGN @Br TO 701;EOL CHR$(10) END
 30 ON INTR 7 GOSUB Poll
 40 ENABLE INTR 7;2
 50 OUTPUT @Br;"SRQMASK USER:ON;RQS ON"
 60 DISP "press RQS icon on CSA 803C/11801C"
 70 GOTO 70
 80 POLL: Stat = SPOLL(701)
 90   OUTPUT @Br;"EVENT?"
100   ENTER @Br;Event$
110   PRINT Stat,Event$
120   ENABLE INTR 7
130 RETURN
140 END
```

Transferring a String to the CSA 803C/11801C Display

```
10 DIM Text$[100]
20 ASSIGN @Br TO 701;EOL CHR$(10) END
30 INPUT "TEXT: ",Text$,"LOCATION: ",X,Y
40 OUTPUT @Br;"TEXT X:";X;",Y:";Y;",
    STRING:'"&Text$&"'"
50 END
```

Appendix E:
GPIB Interface Functions

GPIB Interface Functions Implemented

Table E-1 lists the GPIB interface function and electrical function subsets supported by the CSA 803C and 11801C with a brief description of each.

Table E-1: GPIB Functions

Interface Function	Subset	Meaning
Acceptor Handshake	AH1	The instrument can receive multi-line messages across the GPIB from other devices.
Controller	C0	No Controller capability; the instrument cannot control other devices.
Device Clear	DC1	The instrument can respond both to the DCL (Device Clear) interface message, and to the Selected Device Clear (SDC) interface message when the instrument is listen-addressed.
Device Trigger	DT0	No Device Trigger capability; the instrument does not respond to the GET (Group Execute Trigger) interface message.
Electrical	E2	The instrument uses tri-state buffers, which are optimal for high-speed data transfer.
Listener	L4	The instrument becomes a listener when it detects its listen address being sent over the bus with the ATN line asserted. The instrument ceases to be a listener and becomes a talker when it detects its talk address being sent over the bus with the ATN line asserted.
Parallel Poll	PP0	No Parallel Poll capability; the instrument does not respond to PPC (Parallel Poll Configure), PPD (Parallel Poll Disable), PPE (Parallel Poll Enable), or PPU (Parallel Poll Unconfigure) interface messages, nor does it send out a status message when the ATN and EOI lines are asserted simultaneously.
Remote/Local	RL1	The instrument can respond to both the GTL (Go To Local) and LLO (Local Lock Out) interface messages.

E-2

Table E-1: GPIB Functions (Cont.)

Interface Function	Subset	Meaning
Service Request	SR1	The instrument can assert the SRQ line to notify the controller-in-charge that it requires service.
Source Handshake	SH1	The instrument can initiate multi-line messages to send across the GPIB to other devices.
Talker	T5	The instrument becomes a talker when it detects its talk address being sent over the bus with the ATN line asserted. The instrument ceases to be a talker and becomes a listener when it detects its listen address being sent over the bus with the ATN line asserted. The instrument also ceases to be a talker when it detects another device's talk address being sent over the data lines with ATN asserted.

Appendix F:
System Event Handling

Status and Event Reporting System

Status bytes and event codes combine to represent common instrument system events. Figure F-1 shows the remote interface status and event reporting system of the instrument and summarizes its major elements. These elements will be discussed in the following pages.
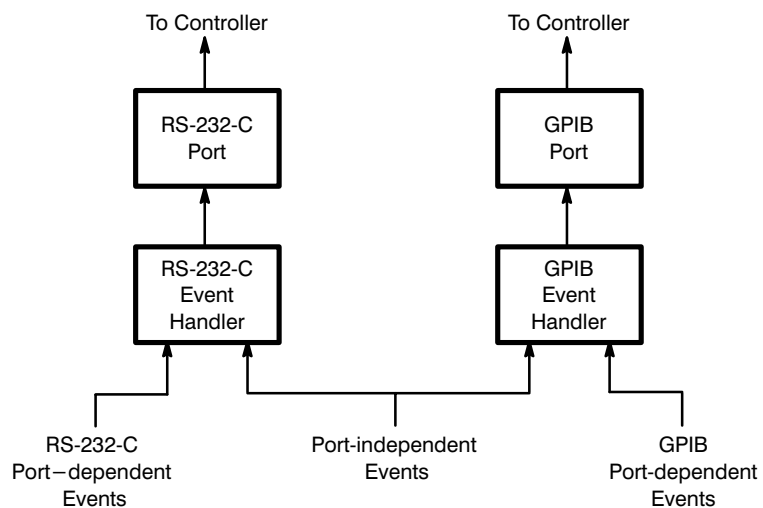
Figure F-1: Remote Interface Status Reporting System Block Diagram

The system events that are generated by the instrument are handled as either port-dependent or port-independent events.

Port-dependent Events

A port-dependent event is generated when one of the following system status conditions occurs:

- Command error

- Execution error

- Execution warning

Port-dependent events are returned only to the port responsible for the event. For example, if the instrument detects a command error in an RS-232-C-only command, the event associated with the error will be returned only to the RS-232-C port.

Port-independent Events

Port-independent events are always returned to both the GPIB and RS-232-C ports. A port-independent event is generated when one of the following system status conditions occurs:

- Internal error

- Internal warning

- Operation complete

- Power-on

- User request

System Event Handling Priorities

Since more than one event may occur before a GPIB or RS-232-C controller can respond to a service request, the instrument uses priorities to report events. See Table F-1.

Table F-1: Event Priorities

Priority	Event Class
1	Power On
2	Command error
	Execution error
	Execution warning
	Internal error
	Internal warning
	Operation complete
	User request
3	No status to report

RS-232-C Event Handling

Figure F-2 is a block diagram of the RS-232-C event handler. The event handler consists of two software registers (SB and EC in the illustration) for the current status byte and current event code, and a LIFO (last-in first-out) buffer.
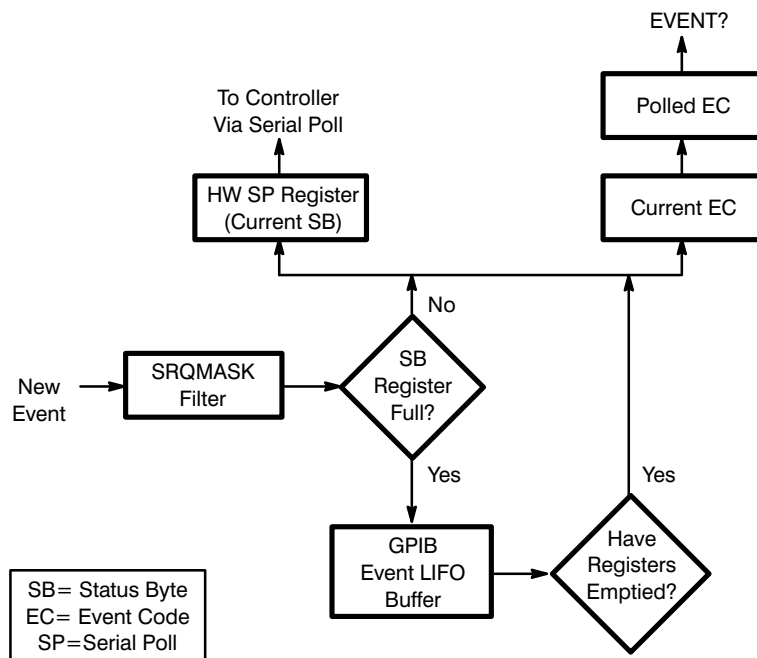
Figure F-2: RS-232-C Event Handling

When a new event is passed to the event handler, the instrument checks the `SRQMask` for that event. If the `SRQMask` is off, the event is discarded. If the `SRQMask` is on, the instrument checks to see if the current status byte register is empty (has "no status to report"). If it is empty, the event handler latches the new status byte and event code into the current status byte and event code registers. Once these registers contain data, all subsequent events are stacked in a 40-event LIFO buffer. Should a new event cause the LIFO buffer to overflow, the oldest event in the buffer is discarded.

Reading the RS-232-C Current Event Registers

An RS232 `STByte?` query returns the contents of the current status byte register. This is a nondestructive read.

An RS232 `EVENT?` query returns the contents of the current event code register and, assuming the LIFO buffer contains event(s), moves the top LIFO event into the current status byte and event code registers. If the buffer is empty, the current status byte is changed to "No Status To Report" and event code 400 is written to the current event-code register. In effect,

`EVENT?` causes the RS-232-C event handler to update its software registers and make the next event (if any) available for subsequent `STByte?` or `EVENT?` queries.

GPIB Event Handling

Figure F-3 is a block diagram of the GPIB event handler. This event handler consists of two software registers (Polled EC and Current EC in the illustration), a LIFO buffer, and the IEEE STD 488 serial poll register (a hardware register).

Operation of the GPIB event handler depends upon whether GPIB `RQS` is set to `ON` or `OFF`.

NOTE

The RS-232-C current-status-byte software register is functionally equivalent to the serial poll hardware register diagram shown in Figure F-3.

Event Reporting When GPIB RQS is Off

When GPIB `RQS` is off, the polled event code register is not used when a new event is passed to the event handler. If the `SRQMask` for an event is off, then the event is discarded. However, if the `SRQMask` for the event is on, the instrument checks to see if the current status byte register is empty or has "no status to report." If it is empty, the event handler latches the new status byte and event code into the hardware serial poll register and current event code register. Once this latched state is entered, all subsequent events are stacked in a 41-event LIFO buffer. Should a new event cause the LIFO buffer to overflow, the oldest event in the buffer is discarded.

Notice that when GPIB `RQS` is off, the GPIB event handler behaves virtually the same as the RS-232-C event handler, with the exception that the current status byte is stored in a hardware register and not in a software register.

Reading the GPIB Current Event Registers (RQS Off)

A GPIB controller uses an IEEE-STD-488 serial poll to read the contents of the hardware serial poll register, which is identical to the current status byte register. This is a nondestructive read. There is no instrument command provided to read the GPIB hardware serial poll register.
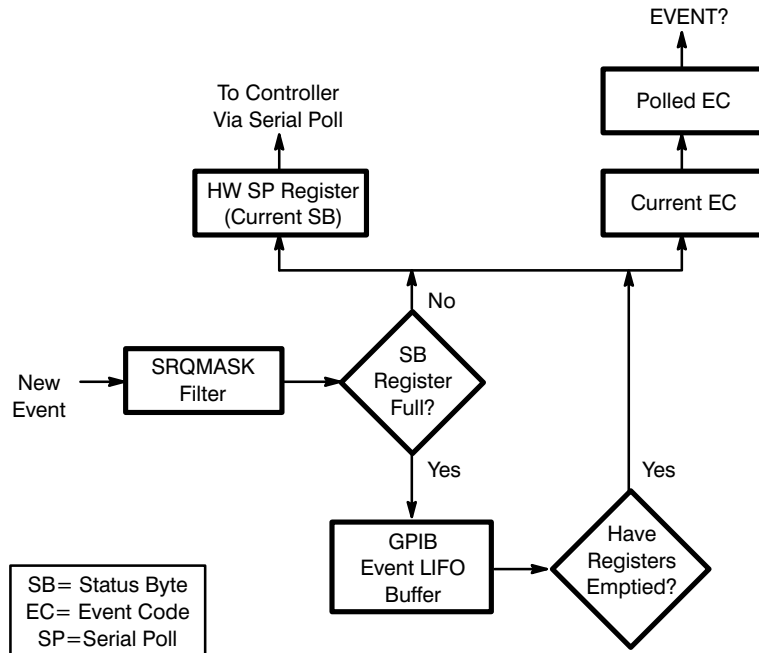
Figure F-3: GPIB Event Handling

A GPIB EVENT? query command returns the contents of the current event code register and, assuming the LIFO buffer contains event(s), moves the top LIFO event into the current status byte and event code registers. If the buffer is empty, the current status byte is changed to "No Status To Report" and event code 400 is written to the current event code register. In effect, EVENT? causes the GPIB event handler to update its hardware and software registers, and make the next event (if any) available for subsequent serial polls or EVENT? queries.

NOTE

When RQS *is off, only the* EVENT? *query updates the event handler's software and hardware registers. Repeated serial polls simply return the same status-byte value.*

Event Reporting When GPIB RQS is On

When a new event is passed to the event handler, the same operations are executed as when GPIB `RQS` is off. The only difference is that bit 7 of the status byte of the new event is set, causing the instrument to assert SRQ after writing the status byte to the serial poll register.

Note that when GPIB `RQS` is on, the polled event code register is significant. At power-on or whenever `RQS` is turned on, this register is initialized with event code 0, which is referred to as the NULL event. The description string of the NULL event is:

```
"RQS is ON...status byte pending"
```

Reading the GPIB Current Event Registers (RQS On)

When GPIB `RQS` is on, it is the IEEE-STD-488 serial poll (not the `EVENT?`) that causes the event handler to update its event registers.

When the instrument asserts SRQ, an external controller must first serially poll the instrument to read the status byte of the system event that just occurred. The instrument responds to the serial poll by moving the current event code register contents into the polled event code register. The instrument next checks for pending events in the LIFO buffer. If found, the instrument moves the status byte of the top event into the hardware serial poll register, thus updating it and causing the instrument to generate another SRQ. At the same time, the event code for top event is moved into the current event code register, thus updating it. However, if no events are pending in the LIFO buffer, the instrument moves a status byte into the hardware serial poll register that indicates No Status To Report, and its corresponding event code 400 is moved into the current event code register. No SRQ is generated under these conditions.

If a controller sends an `EVENT?` following the serial poll, the instrument returns the contents of the polled event code register and initializes it to the NULL event. Then, at the next serial poll, the instrument again moves the contents of the updated current event code register into the polled event code register. This operation ensures that the status byte and the polled event code.

Summary of Important Points When RQS is On

■ The `EVENT?` query returns the contents of the polled event code register.

■ The proper sequence for reading event registers is to first serial poll the instrument and then, if more information is desired, follow up with an `EVENT?` query.

■ When `EVENT?` returns the NULL event, the instrument is signaling that a new event has occurred and its status byte must first be polled before its event code can be queried.

■ If more than one event is pending and the instrument is serially polled twice with no intervening `EVENT?`, the event code associated with the first polled status byte is lost.

Turning On the RQS Icon with SRQMASK USER

The `SRQMask USEr` command allows you to make a Request for Service (RQS) from the front panel. When `SRQMask USEr` is on at either the GPIB or RS-232-C port, the instrument displays an RQS icon on its front panel. See Figure F-4. When initially displayed, the RQS icon is not highlighted and is not selected. When you touch the RQS icon, the icon is highlighted and an event 403 (`Front panel RQS icon selected`) is reported to the ASCII port. When `SRQMask USEr` is off at both ports, the icon is not displayed. Since both `USEr` masks are off by default at power-on, the RQS icon is not visible at that time.

The **RQS** → RQS
Icon

Cursors   Window   Def Tra

1.3443V

200mV
/div
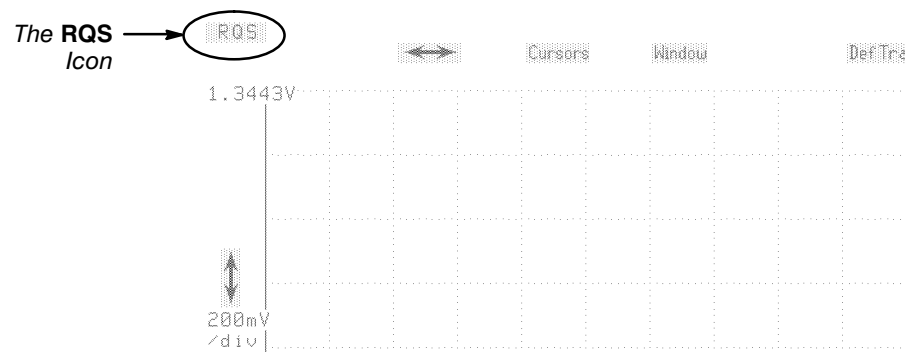
Figure F-4: RQS Icon on the Front Panel Display

The RQS icon changes from selected to not selected under any one of these circumstances:

- Both GPIB `SRQMask USEr` and `RQS` are on and a GPIB controller serially polls (and thereby clears) the status byte associated with event code 403.

- The GPIB `SRQMask USEr` is on and `RQS` is off and a GPIB controller uses `EVENT?` to read (and thereby clear) event code 403 from the GPIB event stack.

- The RS-232-C `SRQMask USEr` is on and an RS-232-C controller uses `EVENT?` to read event code 403.

- The GPIB `SRQMask USEr` is on and DCL (Device Clear) or SDC (Selected Device Clear) is received at the GPIB port. In this situation, all pending events (including event 403) are discarded. RS-232-C DCL has the same effect (assuming the RS-232-C `SRQMask USEr` is on).

- The GPIB `SRQMask USEr` is on and event code 403 is discarded from the GPIB stack. This situation arises when a GPIB controller does not query the GPIB event stack and subsequent instrument events cause the stack to overflow. When event code 403 is discarded, the message **Request for external service ignored** appears on the screen.

If the RS-232-C `SRQMask USEr` is on and the above condition appears at the RS-232-C port, the instrument takes the same actions as it did for the GPIB interface.

Events Reported at Instrument Power-On

When the instrument is powered on, diagnostic tests automatically execute (unless bypassed with hardware straps). When the diagnostics complete, nondiagnostic firmware in the instrument takes over and the remote interfaces are activated. The instrument then reports power-on status: event 401 (power on) if diagnostics passed or were bypassed, or event 394 if diagnostics failed. Specific information about diagnostic failure can be obtained with the `DIAg?` query-only command.

Following the power-on status report, the integrity of the instrument nonvolatile RAM (NVRAM) is checked and, if found to be unsatisfactory, one of the following events is reported:

- Event 657 — NVRAM was completely initialized and all stored settings (if any) were discarded. This event is typically reported when the NVRAM battery fails.

- Event 658 — This is the same as event 657, except that the following conditions are not initialized from the factory settings: mainframe link of the `UID?` command, the number of times the instrument has been powered on, and the length of time the instrument has been powered on.

Event 658 is typically reported when bad settings are passed to the instrument from an external interface, causing a software crash. In this case, event 658 is reported when the instrument is subsequently powered off and then on.

Glossary

Acquisition

The process of repeatedly sampling the signals coming through input channels, and accumulating the samples into traces.

ASCII

Acronym for American Standard Code for Information Interchange. ASCII is a standard eight-bit code used by many computers and data terminals.

Asynchronous

Relating to data transmissions which are not synchronized through a system clock. Also, errors which are not synchronized with a command.

Autoset

A means of letting the instrument set itself to provide a stable and meaningful display of a given trace.

Averaging

Displaying a trace that is the combined result of several acquisitions, thereby reducing random noise.

Baseline Correction

The process of maintaining the displayed vertical placement of a trace, correcting for changes in the signal levels that would ordinarily move the trace up or down.

Binary Block

Tektronix-specified format for binary data transmissions: %,*<byte count><data value><data value>...<data value> <checksum>*.

BNF

Acronym for Backus-Naur Form, which is a formal language structure for syntax definition.

Channel

A place to connect a signal or attach a network or transmission line to sampling heads. Also, the smallest component of a trace expression.

Channel Number

The number assigned to a specific signal input connector. The top channel of the left-most sampling head compartment of the mainframe is always mainframe channel 1, regardless of any repositioning or omission of sampling heads.

Checksum

Checksum comparison is a serial communication operation used to verify data accuracy by comparing the sum of data received against a previously computed sum (checksum).

Color Graded

A display mode in which the instrument displays regions of a trace in different colors according to the densities of displayed points (based on multiple acquisitions of the trace) in that region.

Complex trace

A trace with a trace expression beyond a single channel specification. Any trace using a numeric value, a function, a reference to a stored trace, or an arithmetic operator is a complex trace. However, using the average function does not make a trace complex.

Concatenate

To link commands together.

Cursor

Any of four styles of paired markers that you position with the knobs or `CURSor` commands. The instrument displays the positions of the cursors and the distance between them, in axis units.

DCE

Acronym for Data Communications Equipment. The instrument is configured as a DCE device as defined in the EIA standard RS-232-C.

Debug Mode

Copies input data from either the GPIB or the RS-232-C interface to the front panel display for program troubleshooting.

Default Measurement Parameter

A value from the default set of measurement parameters. The operator can change the default values. Whenever a trace is created, the measurement parameters are copied from the default set.

Device-Dependent Message

Messages initiated by a controller that can only be understood by a specific device. The entire command set are the primary device-dependent messages for the instrument.

Distal

The point farthest (most distant) from a reference point. As used in the instrument, the ending measurement point for some timing measurements.

DMA

Acronym for Direct Memory Access. DMA capability is a feature available in some controllers that transfers data directly into memory by bypassing the central processing unit (CPU). The instrument comes standard with a GPIB-compatible DMA.

DTE

Acronym for Data Terminal Equipment which is a computer or terminal as defined in EIA standard RS-232-C.

DUT

Acronym for Device Under Test.

EIA

Acronym for Electronics Industries Association.

Enveloping

Displaying a trace that shows the extremes of variation of several acquisitions.

Escape Character Set

An alternate character set that is accessed by including an ASCII escape character (decimal 27) in front of the appropriate ASCII character.

Floating Point Value

A type of numeric argument (*<NR2>* or *<NR3>*) that includes a decimal point and may include an exponent.

GPIB (General Purpose Interface Bus)

The GPIB interface is an eight-bit parallel bus that allows remote computer control of the instrument and other synchronous devices. GPIB characteristics are specified in IEEE STD 488 1978.

Hardware Measurement

An automated measurement that is captured by special circuitry that monitors signals directly, as opposed to software measurements that are derived from acquired trace samples.

Histogram (Display)

A statistical function, selected from the Display Modes major menu, that displays a graph of the statistical distribution of acquired sample points. The histogram graph may be calculated for either the horizontal values or vertical values of the sample data. The histogram is performed on the selected trace within a window defined by the user with the the Histogram Limits.

Histogram (Measurements)

The instrument performs a histogram algorithm on a trace to determine some measurement parameters, such as topline and baseline. This measurement histogram is not directly controllable by the user.

Horizontal Reference Point

The point about which the trace is expanded or contracted when horizontal size adjustments are made. The horizontal reference point remains anchored as the rest of the trace grows or shrinks around it.

IEEE STD 488

The Institute of Electrical and Electronic Engineers specification for the GPIB interface.

Infinite Persistence Mode

A mode of operation where the instrument displays newly acquired trace data points and keeps the previously acquired data points on the screen.

Initialize

Setting the instrument to a completely known set of default conditions.

Internal Clock

A trigger source that is synchronized with the Calibrator signal.

LIFO

Acronym for the Last-In First-Out method used to process I/O buffer contents.

Loop Gain

Adjusts the capability of the sampling head to respond to large changes between samples. Excessive loop gain causes the sampling head to overshoot large transitions, while insufficient loop gain causes the sampling head to undershoot transitions.

LSB

Acronym for Least Significant Bit or Least Significant Byte.

Main

Refers to the primary time base used for acquiring data. See Window.

Major Menu

The menu that is displayed at the bottom of the screen alongside the Knob menu. One of the several major menus is always displayed.

Mask

A displayed, user defined polygon with up to 50 vertices used for Mask Testing.

Mask Testing

Testing of the selected trace against one to ten user defined masks. Sample points falling within the masks are added to a count which may be used to control acquisition.

Measurement

An automated numeric readout that the instrument provides directly from the displayed trace in real time, without operator intervention.

Measurement Parameter

One of several control/command parameters that the instrument operator can exercise over the automated measurement process.

Measurement Statistics

The accumulation of a history of individual measurement readouts, showing the mean and standard deviation of a selected number of samples.

Measurement Tracking

The process of automatically adjusting the measurement parameters to reflect changes in the trace.

Mesial

The middle point of a range of points. As used in the instrument, the middle measurement point between proximal and distal points for timing measurements.

MSB

Acronym for Most-Significant Bit or Most-Significant Byte.

Nonvolatile RAM (NVRAM)

Random access memory (RAM) with a battery backup system to prevent the loss of data in case of power failure.

Offset Nulling

Adjusts the DC voltage accuracy of a sampling head by nulling offset errors in the sampling head so that the DC output voltage equals the DC input voltage.

Parse

To decode or analyze data according to a syntax.

Persistence

The amount of time a data point remains displayed. There are four persistence modes available in the CSA 803C and 11801C: Normal, Variable, Infinite, and Color Grading.

Pixel

A visible point on the display. The display is 551 pixels wide and 704 pixels high. Each pixel may be set to one of eight predefined colors.

Proximal

The point nearest (in closest proximity) to a reference point. As used in the instrument, the beginning measurement point for timing measurements.

Quoted String

An element of instrument command syntax (`<qstring>`). A quoted string is required by some command arguments and returned as responses to specific queries. The `<qstring>` element is enclosed by quotes and can be any of the characters defined in the instrument character set.

Record Length

The number of samples (data points) that make up a trace.

RS-232-C

An interface that allows remote operation of the instrument via a controller or terminal. Serial asynchronous data can be transmitted between the instrument and another device as defined in EIA standard RS-232-C.

Sample Interval

The time interval between successive samples in a trace record.

Sampling Head

A high-performance amplifier unit that captures the incoming signal and reports the sampled data to the mainframe. Sampling head units can be selected for specifications that best suit your application.

Scalar

A specific quantity that has magnitude but not direction (a real number, not a vector).

Selected Trace

The highlighted (brightest) trace of a multi-trace display. The selected trace is the trace that is acted on by the knobs, menu selectors, and commands.

Setting

The state of the front panel and system at some given time.

Signed Integer Value

A type of numeric argument (`<NR1>`) which is an integer with a leading sign.

Smoothing

Processing applied by the sampling head prior to the digitization of a trace, to reduce apparent noise. With smoothing, the sampling head samples the signal 8 times instead of once, and the average of the samples is then used by hardware measurements and the digitizing circuitry.

Software Measurement

An automated measurement that is derived from acquired trace samples, as opposed to hardware measurements that are captured by special circuitry that monitors the signals directly.

Stored trace

A trace record with attributes that is saved in a dedicated area of memory.

Synchronous

Data transmission in which timing is provided by a clock in the sending unit.

TDR

Acronym for Time-Domain Reflectometry. TDR is a method of characterizing a transmission line or network by sending a signal from one end and monitoring the electrical reflections. The CSA 803C and 11801C are compatible with TDR technology when used with a TDR-capable sampling head.

TDR Amplitude

A sampling head adjustment for the amplitude of the TDR pulse.

Tektronix Codes and Formats

An shortform title for the Tektronix GPIB Codes, Formats, Conventions, and Features internal standard. The instrument syntax and commands comply with this internal Tektronix standard.

Time Base

The time-dependent specifications that control the acquisition of a trace. The time base determines when and for how long to acquire and digitize signal data points.

Trace

The visible representation of an input signal or combination of signals. Synonymous with waveform.

Trace Expression

The definition of a displayed trace. It can include one or more channels combined arithmetically and modified by functions.

Trace Number

A number assigned by the instrument to identify a trace. Display traces are numbered 1 through 8. A new trace is always given the lowest available number.

Tracking

The process of automatically adjusting the measurement parameters or window position to reflect changes in the trace.

Trigger

An electrical event that initiates acquisition of a trace record and to which time attributes and measurements are referenced.

Truncate

To delete less significant digits from a number. Truncation reduces precision.

Twos-Complement

A representation of negative numbers used by digital computer systems to facilitate arithmetic processing.

Uptime

The number of hours the instrument has been powered on.

Unsigned Integer

A type of numeric argument (`<ui>`) which is an integer without a leading sign (for example, `TRACE <ui>` or `TRACE3`).

Variable Persistence Mode

A mode of operation where the instrument displays newly acquired trace data points and keeps the previously acquired data points on the screen for a specified duration.

Waveform

The visible representation of an input signal or combination of signals. Synonymous with trace.

Waveform Preamble

A response returned from the `WFMPre?` query that contains the scaling information for the trace. A trace preamble consists of the WFMPRE header followed by preamble arguments. All preamble data are ASCII encoded.

Window

Data acquired using a secondary time base with a higher sample rate (and therefore higher resolution) than the Main time base. (See Main.) A trace that represents a horizontally expanded portion of another trace.

XY trace

A trace where both the horizontal and vertical position of the data points reflect signal data.

Yt trace

A trace where the vertical position of the trace data points reflects signal data, and the horizontal position of the trace data points reflects time.

Index